Université de Nancy I

Centre de Recherche en
Informatique de Nancy
Inria Lorraine

La composante lexicale dans les systèmes de dialogue oral homme-machine du CRIN

THÈSE

présentée et soutenue publiquement le **19 septembre 1988**

pour l'obtention du **doctorat de l'Université de Nancy I**

(Mention Informatique)

par

Bernard Mangeol

Composition du jury :

Président : Jean-Paul HATON

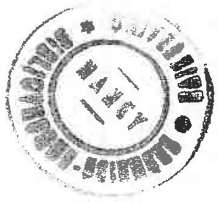
Rapporteurs : Guy PÉRENNOU
Marion CRÉHANGE

Examinateurs : Roger MOHR
Jean-Marie PIERREL

Université de Nancy I

Centre de Recherche en
Informatique de Nancy
Inria Lorraine

La composante lexicale dans les systèmes de dialogue oral homme-machine du CRIN



THÈSE

présentée et soutenue publiquement le **19 septembre 1988**

pour l'obtention du **doctorat de l'Université de Nancy I**
(Mention Informatique)

par

Bernard Mangeol

Composition du jury :

Président :	Jean-Paul HATON
Rapporteurs :	Guy PÉRENNOU
	Marion CRÉHANGE
Examinateurs :	Roger MOHR
	Jean-Marie PIERREL

Table des matières

A Claude,

à Boris et Régis.

Je tiens à exprimer mes vifs remerciements :

à Jean-Paul HATON, professeur à l'Université de Nancy 1, qui me fait l'honneur de présider ce jury,

à Jean-Marie PIERREL, professeur à l'Université de Nancy 1, qui a dirigé ce travail, pour m'avoir accueilli dans ce groupe de travail, pour ses conseils et pour l'appui amical qu'il m'a toujours manisfesté,

à Marion CREHANGE, pour ses conseils très utiles –et très amicaux– pour améliorer ce travail,

à Guy PERENNOU, pour avoir bien voulu juger ce travail, malgré l'importance des nombreuses tâches qu'il assume.

à Roger MOHR, d'une part pour avoir accepté de siéger à ce jury, et d'autre part pour m'avoir soutenu dans mes premières expériences d'enseignement. Sa franchise et ses conseils amicaux m'ont souvent beaucoup aidé.

Je ne voudrais pas oublier tous ceux qui ont participé aux projets qui ont abouti à ce travail, particulièrement :
Jean-Jacques Bonin, Noëlle Carbonell, Christophe Cérin, Dominique Fohr, Yves Laprie, Philippe Morin, Pierre Mousel, Pascal Richard, Laurent Romary et Azim Roussanaly

et surtout Alain Quéré, qui m'a supporté (dans les deux sens du terme!) pendant plus de deux ans dans son bureau, en répondant avec une patience extraordinaire à mes questions les plus diverses, et en aidant le jeune enseignant-chercheur que j'étais dans tous les domaines : administratifs, pédagogiques, informatiques ou matériels. Je ne peux pas quitter cet établissement sans me sentir infiniment redevable pour tout ce qu'il m'a apporté.

INTRODUCTION

Nous vivons en cette fin de siècle une véritable révolution technologique. Si l'informatique a déjà plus de quarante ans aujourd'hui, la véritable révolution ne réside pas dans cette science, mais dans les nouveaux usages qui en sont faits. Le temps où l'informatique et l'usage d'ordinateurs étaient réservés à quelques privilégiés est définitivement révolu. La micro-informatique et la télématique permettent à de nombreux non-spécialistes d'utiliser des systèmes automatisés, dans leur travail, leurs loisirs ou plus simplement leur vie familiale.

Cette ouverture aux non-spécialistes impose aux concepteurs de systèmes d'en augmenter la convivialité, de simplifier les interfaces homme-machine afin de les rendre les plus naturelles possibles vis-à-vis d'un usager occasionnel. De nouveaux concepts sont donc apparus récemment, accompagnés de nouveaux produits : les écrans graphiques couleurs, avec souris, des écrans tactiles, des tablettes graphiques et des synthétiseurs de parole ont déjà envahi le marché.

Cependant, la parole représente certainement pour l'homme le moyen de communication le plus spontané et le plus naturel. Adjoindre à une machine des moyens de perception supplémentaires (traitement de la parole, traitement d'image) et une interprétation des messages reçus, constitue donc les objectifs de demain. Il n'est donc pas étonnant qu'une partie importante des recherches en informatique portent sur le développement de systèmes automatiques de reconnaissance de la parole. Si des réalisations existent déjà, elles se limitent encore à la reconnaissance de mots isolés, parfois connectés, et ce sont le plus souvent des systèmes mono-locuteurs, acceptant un lexique très restreint.

Cette restriction au niveau lexical est due essentiellement à la méthode de reconnaissance de mots adoptée : une comparaison aveugle du mot prononcé avec tous les mots que le système connaît fournit —en principe— ce mot.

Pour permettre une communication en langue quasi-naturelle, bien que toujours limitée à un domaine d'expertise, le mécanisme de reconnaissance des mots prononcés devient beaucoup plus complexe. Dans les systèmes de reconnaissance sophistiqués, ce travail est confié à un module spécialisé. Notre objectif

est de présenter ce module qui joue un rôle fondamental dans un système de compréhension, à cause du rôle pivot joué par les mots dans tout discours.

Notre premier chapitre s'attachera donc à montrer l'importance et le rôle du lexique, dans les systèmes automatiques de compréhension d'une part, mais aussi chez l'homme, à l'aide de modèles fournis par des psycho-linguistes.

Un deuxième chapitre nous permettra de recenser d'une part les informations nécessaires à la compréhension d'énoncés de parole, dans le cadre d'une application finalisée, et d'autre part les traitements associés à chacune de ces informations.

Dans le chapitre III, nous étudierons de façon détaillée trois systèmes en cours de développement dans notre laboratoire, en vue de préciser les spécifications que nous souhaitons pour le niveau lexical d'un système de reconnaissance automatique de la parole.

Les quatrième et cinquième chapitres décrirons les développements faits dans le cadre de ces trois systèmes, au niveau lexical, et les résultats obtenus. Nous proposons des modifications au niveau du décodage acoustico-phonétique, pour permettre une validation d'hypothèses phonétiques, émises par le niveau lexical. Chaque composante du système aurait alors le même fonctionnement, pour être à la fois producteur d'informations (hypothèses) d'une part, et consommateur (validations) d'autre part. Ce type de fonctionnement au niveau du décodage acoustico-phonétique devrait permettre de vérifier beaucoup plus finement les mots. En effet, si un décodage parfait, c'est-à-dire où tous les phonèmes sont correctement identifiés, est impossible à faire de façon ascendante, une validation a posteriori des phonèmes manquant ou erronés constitue une nouvelle approche très prometteuse.

Chapitre 1

Importance du niveau lexical.

1 Rôle du lexique

1.1 Introduction

Dans tous les systèmes de compréhension automatique de la parole, le mot et le niveau lexical qui lui est associé ont une importance primordiale pour le décodage et la compréhension d'un énoncé. Le passage par le mot est une étape obligée entre ce qui est souvent appelé les traitements de "haut niveau" et ceux de "bas niveau".

Les traitements de "bas niveau" sont propres à l'oral : ce sont eux qui permettent de passer du signal acoustique à des mots candidats. Dans un système de reconnaissance analytique, où la reconnaissance d'un mot se fait soit à l'aide de traits phonétiques, soit par un passage au niveau du phonème, le niveau lexical devra donc comporter une composante phonétique et phonologique, capable de fournir les mots correspondant au treillis phonétique obtenu ou à l'ensemble des traits décelés sur une partie de signal.

Les traitements de "haut niveau" peuvent reprendre des résultats et des méthodes utilisés pour le traitement du langage écrit. Le rôle du lexique est alors de fournir pour chaque mot à traiter une liste d'informations utiles à une analyse syntaxique, sémantique et pragmatique. On devra donc d'abord définir les informations à fournir, puis faire un choix d'implémentation pour ces informations et pour les procédures d'accès (consultation, modification...).

La quantité d'information à associer à chaque mot sera bien sûr dépendante de la complexité du système mis en oeuvre. Pour un système à vocabulaire limité (jusqu'à deux cents ou trois cents mots), ces informations peuvent se limiter à une représentation phonétique et phonologique et à des traits syntaxiques permettant une analyse descendante, guidée par la syntaxe [PIERREL 75]. Si l'application de-

vient plus ambitieuse, une telle approche échoue. Des modèles linguistiques plus élaborés sont nécessaires, à la fois pour la création de la base de donnée lexicale, —c'est à dire permettant de collecter puis coder toutes les informations statiques pouvant servir à un tel système—, et pour la consultation ultérieure de cette base de données. La prise en compte d'informations sémantiques fera augmenter sensiblement le nombre des informations associées à chaque entité lexicale et permettra une focalisation beaucoup plus fine des mécanismes de sélection de mots, permettant ainsi d'éviter ou de retarder une explosion combinatoire du nombre d'hypothèses à traiter, quand la taille du lexique augmente.

Dans tous les cas, la structure d'un système "classique" de reconnaissance pourra se schématiser comme l'indique la figure 1.1 que nous propose Jean-Paul Haton dans [HATON 85]

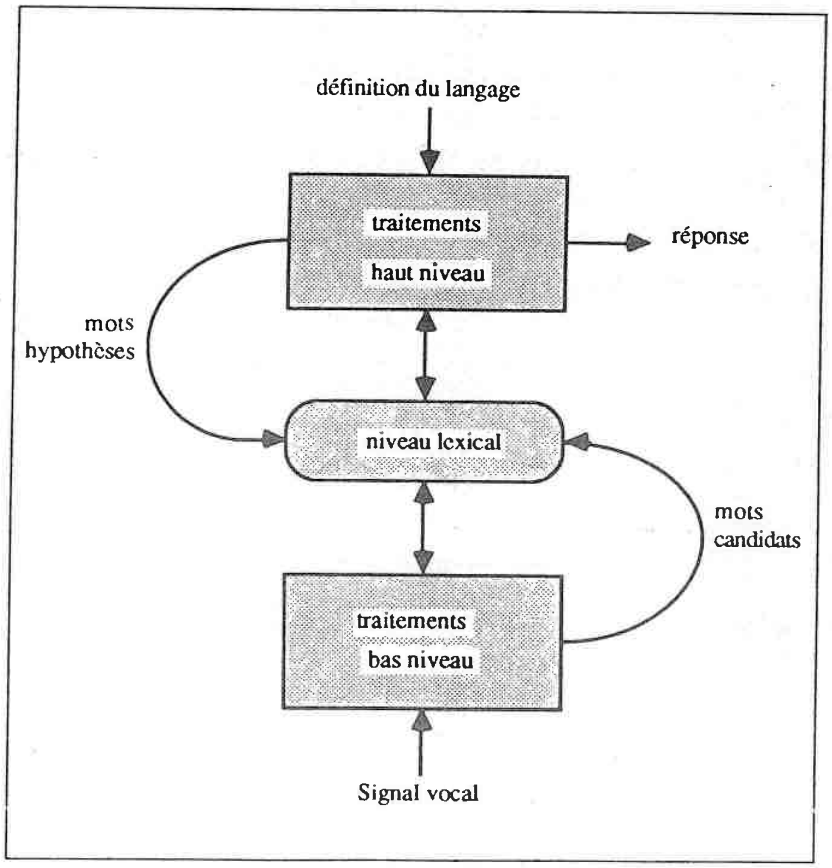


Figure 1.1: Système classique de reconnaissance d'après [HATON85]

Examinons les différentes architectures proposées dans les divers systèmes de

reconnaissance développés et la place prise par le niveau lexical dans ces architectures.

1.2 Systèmes guidés par la syntaxe

Certaines applications de dialogues oraux ne nécessitent que des langages artificiels, restreints, qui seront entièrement décrits par leur syntaxe (par exemple commandes d'un microscope électronique ou d'un système industriel simple). De tels langages pourront donc être définis par un grammaire à contexte libre ou par un automate. Aucun traitement sémantique n'est nécessaire, car il existe une correspondance immédiate entre une structure syntaxique donnée et la commande à exécuter. Le lexique de tels systèmes se réduira donc à des informations phonétiques et phonologiques pour les niveaux bas, les seules autres informations étant syntaxiques. Deux approches duales conduisent à deux architectures linéaires, l'une descendante, guidée par la syntaxe, l'autre ascendante, prenant d'abord en compte les mots de meilleurs scores fournis dans le treillis lexical.

Le projet MYRTILLE I [PIERREL 75] développé par Jean-Marie Pierrel à Nancy était centré sur l'analyse syntaxique, qui émettait des hypothèses de type mots à vérifier sur le treillis phonétique. L'ordre de la vérification lexicale de chacune de ces hypothèses dépend alors de la stratégie adoptée. Le schéma général de ce système nous est donné par la figure 1.2 [1].

Le système KEAL développé par le CNET (Centre National d'étude des Télécommunications) [MERCIER 77] à Lannion illustre bien l'autre stratégie, qui consiste à activer de façon séquentielle et ascendante les diverses étapes du processus de compréhension :

1. Construction d'un treillis lexical complet à partir du treillis phonétique.
2. Recherche dans ce treillis de la suite optimale de mots correspondants à une analyse syntaxique correcte.

L'architecture de ce système correspond donc à la description donnée par la figure 1.3

1.3 Systèmes à vocabulaire plus étendu

Là encore, le rôle et la place du lexique dans le système va dépendre de la philosophie du concepteur. Nous distinguerons essentiellement deux modèles

[1] D'après le livre de Jean-Marie Pierrel [PIERREL 87] et qu'il m'a gracieusement laissé reproduire

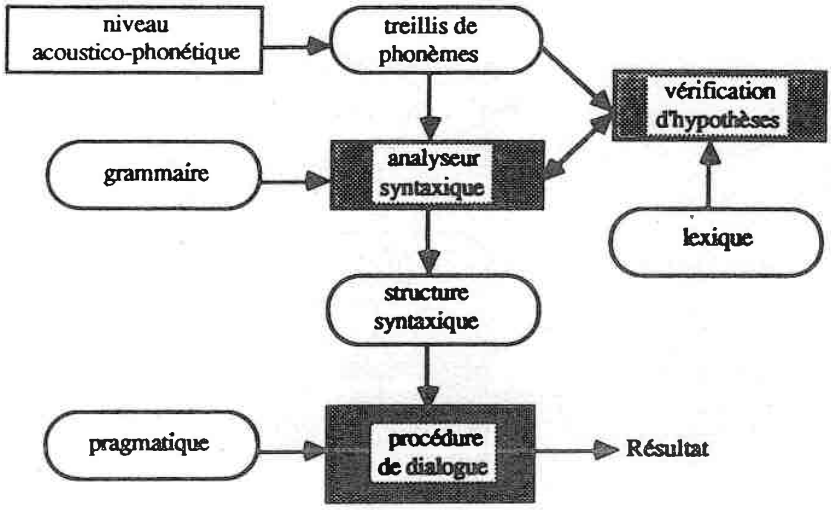


Figure 1.2: Schéma général du système MYRTILLE I d'après [PIERREL 87].

d'organisations pour ces systèmes plus évolués que ceux que nous venons d'étudier :

1. **Modèle non hiérarchique**, centré sur sur une base de données externe :

   Ce modèle, appelé plus communément "Blackboard" a été proposé pour la première fois par Erman [ERMAN 77] dans HEARSAY II.

   Chaque changement dans le blackboard constitue un événement et met fin aux traitements en cours. Le blackboard réévalue alors sa configuration après la prise en compte de cet événement et active une nouvelle base de connaissance (ou KS pour Knowledge Source). Cette activation va obligatoirement conduire à une nouvelle modification du blackboard : un ou plusieurs éléments de solution seront créés, modifiés ou complétés.

   A l'arrêt du processus, qui correspond à une base de connaissances KS particulière –STOP–, une interprétation correspondant à la meilleure structure de phrase trouvée sera engendrée.

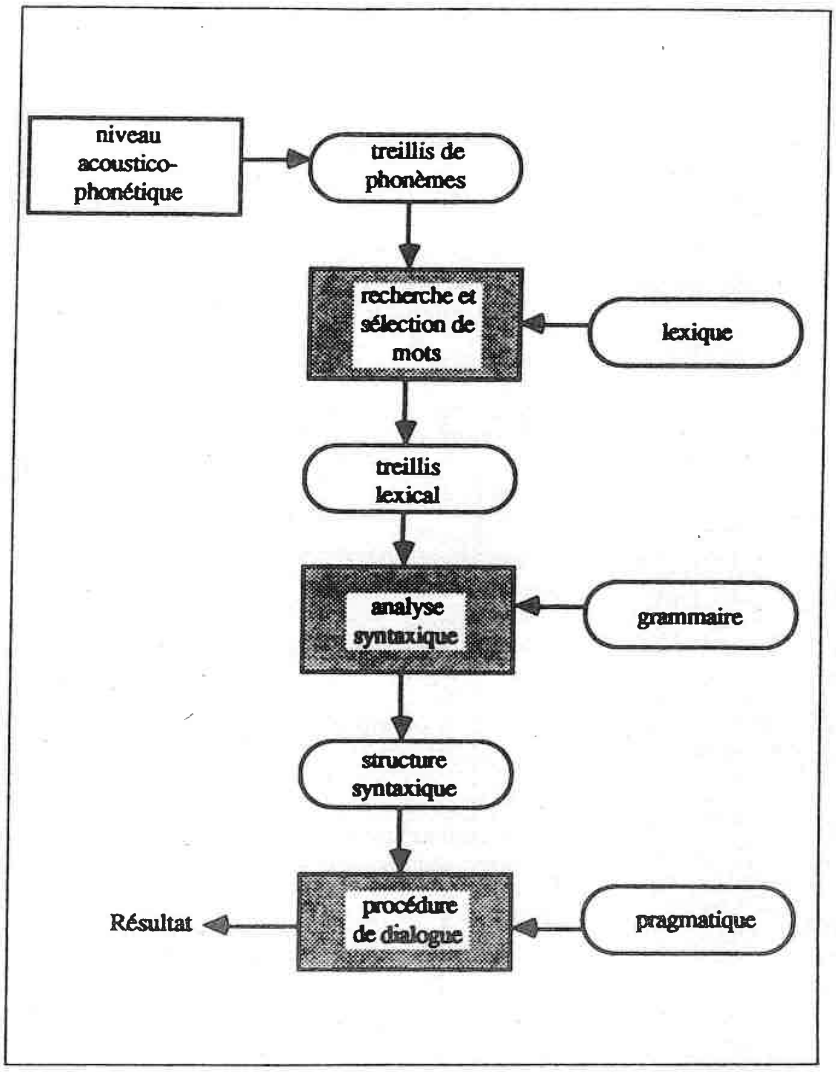


Figure 1.3: Architecture du sytème KEAL du CNET

2. **Modèle hiérarchique**, où l'un des modules joue le rôle de superviseur.

   Dans de tels systèmes, la progression de la reconnaissance est contrôlée par l'un des modules. Si le vocabulaire est étendu, il est aberrant d'essayer de vérifier tous les mots sur toutes les plages de signal. L'idée générale est donc de limiter l'explosion combinatoire : des modules d'émissions d'hypothèses et de restriction du champ des recherches vont d'abord être activés. Quand l'espace de recherche sera suffisamment restreint, on pourra activer les modules de validation. Un module spécial , le "*superviseur* ", sera donc chargé de gérer tous les autres, en décidant de leur activation et de leur arrêt. Des

systèmes comme HWIM de BBN [WOLF 80] ou **MYRTILLE** II du CRIN [PIERREL 81] correspondaient à ce modèle. Toutefois, cette architecture différente n'empêche pas une mise en oeuvre dans un ordre assez similaire des différents modules ou bases de connaissances.

L'architecture du système HWIM nous est fourni par la figure 1.4.

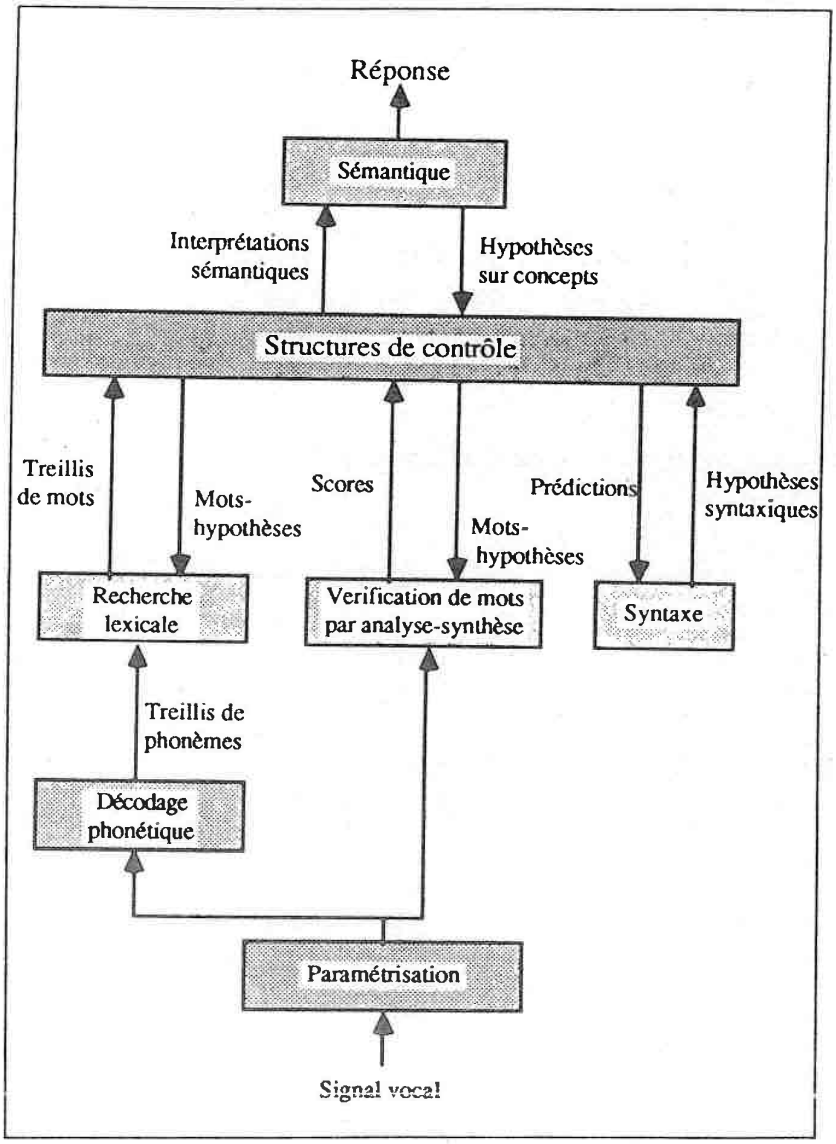


Figure 1.4: Architecture du système HWIM

3. Modèle proposé pour DIAL

Nous venons de voir que le modèle du système utilisé influait peu sur l'ordre de mise en oeuvre des différents modules, seules les informations en cours de traitement étant réellement importantes. Dans le système DIAL développé au CRIN, nous proposons donc une structure où le contrôle est décentralisé, les modules dialoguant deux à deux, selon un modèle producteur consommateur, l'un des agents produisant des hypothèses, son partenaire les consommant. Chacun des modules pourra être à son tour producteur et consommateur, selon le degré d'avancement des traitements.

1.4 Systèmes de dialogue

La sélection de mots sur le treillis fourni par le décodage acoustico-phonétique reste entachée d'erreurs, malgré les récents progrès enregistrés par les niveaux infra-lexicaux. De plus, il n'est pas exclu que le locuteur fasse d'autres erreurs, d'élocution ou de syntaxe. Une analyse syntaxique trop brutale, à la manière d'un compilateur, est donc inadéquate pour des systèmes de reconnaissance automatique de parole, car elle conduirait la plupart du temps à une erreur. La tolérance aux erreurs doit être triple :

1. Élision autorisée d'une partie de l'énoncé par le locuteur :
   -Descends la pince.
   -Prends le cube.
   -Remonte. (Pour "remonte la pince")

   Cette élision peut être autorisée par la grammaire, cette liberté d'expression ayant pour but de rendre le dialogue plus naturel entre l'utilisateur et le système de compréhension.

2. Élision de mots courts, non signifiants :
   -Descends pince.

   Le système ne pourra jamais distinguer le cas où l'élision a été faite par le locuteur ou par le système lui-même, des mots courts de ce type restant difficiles à détecter correctement.

3. Élision de mots signifiants ou remplacement par un mot synonyme, non connu du système :
   -Descends la pince.
   -Prends le cube.
   -Remonte l'ensemble.

Le système ne pourra reconnaître "l'ensemble", si ce mot ne fait pas partie de son lexique. Toutefois, si "Remonte" est parfaitement reconnu, on pourra émettre des hypothèses sur l'objet qui doit subir cette opération. Un mot parasite sera donc remplacé par un mot du langage, si ses contextes gauche et droit sont bien reconnus. Le mot le plus plausible à priori sera retenu si plusieurs candidats sont présents (fréquence d'apparition la plus élevée, longueur phonétique compatible avec la plage de signal non appariée). Les traitements mis en oeuvre seront les mêmes que pour un mot très mal décodé et les deux se confondront pour l'interprétation ultérieure de la commande.

Lorsque le système fait lui-même une hypothèse, il faut valider celle-ci en utilisant une source d'informations différente de celle qui l'a engendrée. En cas d'impossibilité, une confirmation en provenance de l'usager est nécessaire.

Si plusieurs stratégies peuvent être mises en oeuvre, toutes reposent sur la possibilité d'un vrai dialogue avec l'utilisateur, soit pour une validation explicite des INFÉRENCES faites par le système, soit pour une validation implicite (pas de contestation détectée dans le discours de l'usager) , ou une remise en cause (l'utilisateur proteste).

En résumé, dans tous ces types de systèmes, le lexique a un rôle de pourvoyeur de mots à partir d'un treillis phonétique. Il peut aussi fournir d'autres informations, par exemple des longueurs phonétiques moyennes de mots absents ou non reconnus sur le signal, en vue d'un remplacement par les analyseurs ou plus simplement pour évaluer de façon plus fine la pénalité correspondant à leur élision, celle-ci pouvant être proportionnelle à la longueur du mot absent. En effet, si un mot outil, très court, peut passer inaperçu, le terme "*carte d'identité*" ne pourra être élidé complètement et même si ce terme n'est pas reconnu, une plage de signal importante sera nécessaire entre deux mots reconnus, si l'on veut l'insérer à cet endroit.

2 L'accès lexical chez l'homme : modèles des psycho-linguistes

2.1 Introduction

L'homme est capable de comprendre un énoncé de parole en temps réel, c'est-à-dire au fur et à mesure de sa réception. Une partie importante de son travail de compréhension est donc faite en parallèle avec l'audition des mots qui composent l'énoncé. L'étude des mécanismes impliqués dans l'identification des

mots a déjà fait l'objet de travaux conséquents de la part de psycho-linguistes. Il nous a donc semblé opportun d'examiner les résultats de ces recherches, certains modèles proposés pouvant être facilement implémentables. Juan Segui a particulièrement étudié les procédures d'accès lexical chez l'auditeur humain en essayant de répondre à trois questions [SEGUI 80] :

1. Quelles sont la nature et l'organisation du lexique?
2. Quel est le format des représentations lexicales?
3. Quelles sortes de procédures rendent possible l'accès à ces représentations?

2.2 L'accès lexical : expériences des psycho-linguistes.

Juan Segui distingue d'une part les procédures d'accès au lexique et d'autre part l'utilisation des informations recueillies à la suite de cet accès. Il donne la définition suivante de l'accès lexical :

- Accès : Résultat des opérations qui permettent d'associer une représentation sensorielle à une représentation mentale correspondant à un mot de la langue.

Cet accès pourra être de deux types :

1. Pour "l'accès direct", le code sensoriel sera comparé de façon simultanée aux différentes représentations lexicales.
2. Pour une "recherche active dans le lexique", le code sensoriel est utilisé pour un parcours séquentiel du lexique jusqu'à ce qu'une correspondance soit trouvée entre une représentation lexicale et ce code.

Pour le type d'informations utilisées au cours de cet accès, les différents auteurs n'ont pas encore réussi à trouver un consensus. Là encore, Juan Segui fait une distinction entre deux types de modèles théoriques :

- Les modèles "autonomes" n'utilisent que des informations très spécifiques, par exemple des informations infra-lexicales, issues du traitement du signal. Ces seules informations (ainsi que l'organisation du lexique elle-même, bien sûr) interviennent dans la détermination de l'accès.
- Les modèles "interactifs" semblent plus réalistes : ils supposent que les procédures d'accès sont sensibles non seulement aux informations provenant des niveaux infra-lexicaux, mais aussi à toutes celles issues des autres sources

de traitement, comme par exemple les informations syntaxiques et sémantiques.

Le parallèle entre ces modèles d'accès chez l'homme et les modèles informatiques que nous essayons de réaliser est très frappant : les modèles les plus simples correspondent tout à fait à la démarche suivie par les chercheurs dès 1975 pour mettre en oeuvre de petites applications de reconnaissance. Si le vocabulaire grandit, ces modèles s'avèrent insuffisants et leur amélioration conduit tout naturellement au deuxième type de modèles que nous proposent les psycho-linguistes.

Certains auteurs [SEGUI 88] contestent d'ailleurs la validité des modèles autonomes, et justifient leur modèle interactif en faisant remarquer qu'un mot est plus facilement identifiable dans certains contextes. Par contre, les auteurs qui distinguent l'accès lexical à la représentation d'un mot de la phase d'identification de ce mot ne considèrent pas cet argument comme une preuve décisive. Il semble évident que le contexte joue un rôle dans le traitement perceptif, mais le problème est de connaître quand et comment ce contexte est pris en compte : avant l'identification pour sélectionner un sous-lexique, en parallèle pour aider à cette identification ou a posteriori pour valider la reconnaissance ou lever les ambiguïtés.

Ces modèles théoriques sont étudiés et validés en psycho-linguistique par des tâches expérimentales très diverses :

- détection de cibles : l'usager doit réagir le plus rapidement possible dès que sa cible apparaît dans le signal de parole. Celle-ci peut être de divers types :
  - phone : détection de phonèmes,
  - syllabe : détection des syllabes,
  - mot : détection de mots,
  - mot d'une catégorie donnée : détection catégorielle,
  - mot qui rime avec un modèle fourni : détection de rime.

  La qualité des résultats obtenus avec chacune de ces cibles permet de valider ou non l'utilisation d'informations d'un tel niveau et d'évaluer si ces informations sont disponibles de façon immédiate ou non.

- procédure de décision lexicale faisant appel à un seuil d'identification : on présente un mot à partir de ses seuls segments initiaux, en progressant par tranches de 30 millisecondes par exemple, jusqu'à identification correcte

du mot par l'auditeur. Cette technique permet donc d'évaluer la quantité d'information acoustique nécessaire à un auditeur pour identifier un mot, et ses variations en fonction des propriétés intrinsèques de chaque mot (fréquence, longueur) et en fonction du contexte de présentation.

La quantité d'information de "*bas-niveau* " nécessaire à l'identification va diminuer et ceci va à la fois permettre une identification plus rapide et rendre possible une reconnaissance d'un mot non identifiable sans un tel contexte. La prise en compte de ce type de remarques nous semble donc indispensable lorsque l'on désire implémenter un système plus ambitieux, acceptant un vocabulaire étendu, dépassant le millier d'unités lexicales.

2.3 Modèles d'accès proposés en psycho-linguistique

Les résultats de ces expériences ont été interprétés par leurs auteurs, qui ont ensuite construit des modèles théoriques de fonctionnement du lexique chez l'auditeur humain ou chez le lecteur et l'auditeur. Examinons un certain nombre des modèles proposés :

1. Le modèle de Marslen-Wilson [MARSLENWILSON 78].

   Celui-ci nous semble le plus intéressant à plusieurs titres :

   - Ce modèle ne cherche à rendre compte que de la perception du langage parlé et ne fait pas de référence à des expériences portant sur l'accès lexical pour un sujet lisant un texte. Il prend donc en compte la spécificité des informations traitées par un auditeur. Le modèle proposé fait appel à la notion de cohortes, compatible avec les propriétés de base d'un signal de parole. Celui est en effet perçu de façon séquentielle, la masse d'informations recueillies augmentant en fonction du temps.
   - Le modèle des cohortes voit l'ensemble des mots possibles comme un continuum de possibilités de choix parmi un sous-lexique, qui va diminuer en fonction des informations déjà recueillies et traitées. Le nombre des mots candidats va donc diminuer au cours de l'audition, pour se réduire à un seul item lexical à partir d'un point "t", point à partir duquel ce mot est le seul compatible avec la séquence de sons déjà reçue. En général, ce point d'identification précède la fin du mot. Par exemple, le mot "*identification* " lui-même sera reconnu à partir du son "s" de sa dernière syllabe, mais le mot "*identificateur* " devra attendre un phonème supplémentaire pour se distinguer de son féminin *identificatrice*.

- Ce modèle est interactif, le contexte est utilisé à tout moment pour contribuer au processus de sélection des mots. L'homme comme la machine est en effet bien démuni pour interpréter correctement un mot isolé s'il n'est pas parfaitement articulé. L'actualité nous en fournit la preuve : une petite fille de trois ans, abandonnée et à qui l'on demande son prénom le donne. Un appel est lancé pour retrouver les parents d'une petite Vanessa, agée de trois ans. Après identification par la grand-mère, son prénom exact était Anissa. La recherche du prénom féminin en fonction de la réponse de la fillette s'est arrêtée dès qu'un prénom convenable a été trouvé. Sa proximité avec un autre prénom peut-être moins fréquent a produit l'erreur que l'on sait. Les informations en provenance du signal étaient insuffisantes pour lever l'ambiguïté pour un auditeur humain comme elles l'auraient été pour un machine de traitement de parole. Une compréhension de suites de mots sans lien serait tout aussi difficile pour un homme que pour une machine. Seule la prise en compte d'autres informations permet à l'auditeur humain un compréhension en temps réel d'énoncés de parole.
- Le modèle des cohortes de Marslen-Wilson prend parfaitement en compte ce genre de remarques : l'information acoustique y est vue comme une information de taille importante, jamais disponible globalement. Les sons se succèdent linéairement dans le temps, leur traitement se fera donc de façon séquentielle. Le maximum d'information devra être extrait du lexique de façon pertinente pendant la réception du signal acoustique, en parallèle avec la réception.

Ce modèle d'accès n'est proposé que pour la reconnaissance. On peut également s'interroger sur son usage pour la production de parole. Je me suis longtemps interrogé sur la grande difficulté pour un non-initié à prononcer : "*Je suis un original qui ne se* désoriginalisera *jamais*". Une explication plausible est l'absence de mots ayant comme début "désorig", car l'accrochage se produit à ce niveau. Un petit entraînement crée sans doute cette entrée lexicale chez le sujet, et une diction fluide apparaît rapidement.

Le bon fonctionnement de ce modèle exige une mise en correspondance permanente de l'ensemble des mots du lexique mental avec le signal acoustique, la cohorte étant constituée de l'ensemble des mots compatibles à un instant donné.

Si l'on veut étendre ce modèle à une comparaison floue, admettant des erreurs ou des imprécisions dans la comparaison, le modèle d'élimination sur

un facteur gauche différent n'est pas parfait. Ce modèle est aussi incapable d'expliquer les phénomènes lié à la fréquence d'emploi d'un mot particulier (exemple : la confusion Vanessa–Anissa). Hormis cette imperfection et donc sa fragilité relative aux erreurs, le mode de fonctionnement du processus conduisant à l'identification d'un mot est à retenir : une cohorte initiale correspondant à un sous-lexique actif est initialisée. Cette initialisation est suivie d'une sélection par élimination de tous les candidats incompatibles avec la suite des informations sensorielles perçues. Par exemple, après avoir reconnu "iden", la cohorte se réduit à tous les mots compatibles avec ce début. Tous les mots incompatibles avec ce début s'éliminent d'eux mêmes. Le modèle de Marlsen-Wilson précise qu'il n'y a pas de contrôleur central. Chaque mot actif essaie de détecter une incompatibilité entre ses propriétés et les informations recueuillies. Celles-ci sont d'abord acoustiques, mais peuvent aussi provenir des niveaux supérieurs de traitement. Le processus est entièrement interactif. L'information contextuelle peut accélérer le processus de sélection, en éliminant tous les mots ne correspondant pas au contexte. Le point d'identification est alors plus proche du début, le sous-lexique des mots candidats étant nettement plus réduit. Toutefois, cette présélection ne se fait pas à priori, des informations acoustiques doivent être prises en compte pour sélectionner une première cohorte, qui sera ensuite réduite par confrontation contextuelle dans une deuxième étape.

Enfin, dans le cas d'une mauvaise compréhension initiale, l'échec est total et seul un processus complexe de correction d'erreurs va permettre de rétablir l'information perdue.

2. Le modèle d'Elman et Mac Clelland [ELMAN 84]

   Ce modèle constitue un essai d'implémentation informatique du modèle théorique proposé ci-dessus. Le lexique de base est constitué des mots les plus fréquemment employés en anglais. Le système est de type connexionniste et à chaque mot est associé un noeud interne qui pourra être excité par la détection d'un trait phonétique (22 traits distinctifs sont différenciés) ou par la détection d'un phonème (37 phonèmes sont distingués). Chaque trait ou phonème correspond aussi à un noeud interne, mais de structure plus simple que celui d'un mot.

   La grande originalité de ce système réside dans la transmission bi-directionnelle des signaux :

   - La reconnaissance d'un mot est *classique* :
     Chaque trait identifié va envoyer un signal excitateur à tous les phonè-

mes qui le possèdent. Chaque phonème reconnu aura le même comportement vis-à-vis des mots qui le contiennent.

- Chaque noeud possède un seuil d'excitation à partir duquel il devient actif. Il envoie alors des signaux inhibiteurs aux autres mots de même type que lui et il affecte également les autres unités d'un niveau inférieur, en excitant les phonèmes qui le composent, qui eux-mêmes vont retransmettre cette excitation à leurs traits caractéristiques. Ces traitements bi-directionnels ont été instaurés pour essayer de simuler des données expérimentales qui laissent supposer la présence d'une influence directe du lexique sur le traitement phonétique.

  Le phénomène de fréquence pourra facilement être traduit par un abaissement du seuil de mise en activité, donc la quantité d'informations nécessaire sera d'autant plus faible que la fréquence d'un mot est élevée.

  Sa seule faiblesse réside dans l'absence de modèle pour représenter les niveaux syntaxico-sémantiques.

3. Le modèle de Morton [MORTON 79]

   Ce modèle est assez proche des précédents. L'identification d'un mot se fera grâce à l'activation passive de son détecteur ou "logogène". Ce niveau d'activation va s'accroître avec l'accumulation des indices jusqu'à une valeur suffisante pour permettre l'identification du mot. Ces logogènes constituent une interface entre des niveaux bas (les analyseurs sensoriels) et des niveaux hauts (le système cognitif). Leur niveau d'activité peut-être modifié indifféremment par des informations en provenance de ces deux origines. Morton a démontré que la quantité d'informations nécessaire pour identifier un mot hors contexte était nettement supérieure. Il a effectué des mesures avec le mot "chien", présenté isolement, puis après le début de phrase : "le soir, la dame promène son ..." Son modèle convient parfaitement pour rendre compte de cet effet de focalisation apporté par le contexte. Il est par contre incapable de modéliser les inhibitions observées quand le mot finalement présenté n'est pas un de ceux attendus. L'identification réelle est alors encore plus difficile que l'identification d'un mot présenté de façon isolé.

   Morton a noté l'existence d'un effet d'apprentissage ou de répétition d'un même mot. La reconnaissance d'un mot venant d'apparaître est facilitée lors d'une nouvelle apparition. Ce phénomène est d'autant plus important que les deux présentations successives sont rapprochées. Le taux d'activité d'un mot détecté ne retombe pas de façon brutale à zéro, mais de manière progressive. La quantité d'informations nécessaire pour l'identifier lors d'une

deuxième présentation en est réduite d'autant, même si les informations de "*bas-niveau* " sont physiquement différentes. (exemple donné pour l'écrit : *"bateau"* et "BATEAU").

4. Le modèle autonome de Forster[FORSTER 76]

   Forster nie l'existence d'une interaction des "niveaux hauts" sur le processus d'accès lexical. Celui-ci serait autonome et seules des informations issues des traitements de "bas niveau" peuvent intervenir dans la localisation des entrées lexicales. Il fait donc une distinction nette entre l'accès lexical et le lexique lui-même ou lexique central. Pour Forster, c'est dans ce lexique central que se trouvent toutes les connaissances linguistiques d'un individu, tandis que l'accès lexical se fait par des structures qui permettent d'accèder à ces connaissances. Si le lexique central est unique, les accès sont divers et spécifiques. Il distingue un accès pour une représentation visuelle et orthographique et un autre pour une représentation auditive et phonétique. Chacun de ces accès est factorisé en fonction de la représentation : les mots commençant par une même séquence sont regroupés dans un même secteur, délimitant ainsi les entrées à consulter. Les entrées les plus fréquentes sont placées de façon privilégiée, en tête de leur voie d'accès, pour permettre une consultation plus rapide en cas de recherche. L'accès lexical proprement dit, c'est-à-dire la récupération des informations associées à une entrée ne se fera que dans un deuxième temps, quand le mot sera localisé après appariement avec l'information sensorielle recueillie. On ne pourra donc en aucun cas utiliser des informations contextuelles, syntaxiques ou sémantiques avant d'avoir effectué cet accès. À ce niveau, toutes les entrées lexicales plausibles seraient sélectionnées de manière aveugle, et les ambiguïtés ne seraient levées que plus tard, après consultation des données linguistiques.

   Ce modèle semble moins intéressant, et Juan Ségui qualifie les explications fournies par Forster de *"peu convaincantes"* pour ce qui est des facilitations de compréhension liées au contexte.

3 Position du lexique par rapport aux autres niveaux dans les systèmes de compréhension

Nous venons de voir que les psycho-linguistes distinguent des modèles autonomes, où seules des informations en provenance des niveaux infra-lexicaux (traitement du signal au sens large) sont utilisées dans la détermination des accès au lexique et des modèles interactifs, qui supposent une coopération entre ces

mêmes niveaux infra-lexicaux et des niveaux supra-lexicaux (ou de haut niveau telle syntaxe, sémantique, contexte de dialogue). Notre compétence ne nous permet pas de juger ces différents modèles psycho-linguistiques, mais ils vont nous servir à illustrer les différentes approches des informaticiens pour traiter ces problèmes.

3.1 Le lexique et ses niveaux infra-lexicaux

Essayons tout d'abord de définir ces niveaux infra-lexicaux. Les auteurs en retiennent le plus souvent quatre, mais tous ne seront pas mis en oeuvre de façon systématique. Ce sont :

- la morphologie : Elle vise à définir chaque catégorie ou classe de mots par un ensemble de caractéristiques formelles, autres que le sens ou la sémantique.
- la phonétique : Toute description phonétique s'appuie sur l'établissement d'un système de phonèmes. La figure 1.5 illustre celui que nous utilisons tout au long de ce manuscrit.
- le phonologie : étudie les phonèmes du point de vue de leur fonction linguistique. Elle nous servira à modéliser des déformations (ou altérations) de la prononciation théorique d'un mot.
- la prosodie : C'est l'ensemble des phénomènes linguistiques mélodiques (intonations, accents), et des règles qui régissent ces phénomènes.

Cette dernière n'a pas encore été très utilisée dans les systèmes de reconnaissance automatique de la parole. Nous y reviendrons en temps voulu quand nous parlerons des projets en cours de développement au CRIN.

La partie purement phonétique ne pose guère de problèmes particuliers : tous s'accordent pour dire qu'un lexique devra contenir une représentation phonétique standard des mots qui le composent pour faire de la reconnaissance automatique de la parole.

la composante morpho-lexicale

Le problème posé est simple, mais n'a pas encore de réponse à ce jour : Doit-on coder en machine toutes les formes possibles d'un mot ou bien au contraire n'y mettre que son radical, et déduire à l'aide de procédures de construction automatique les différentes dérivations de ce radical, en y ajoutant les désinences, préfixes

VOYELLES			CONSONNES		
Phon.	exemples	classe	Phon.	exemples	classe
a	patte, bas, pâte	Orales	p	pas	Plosives sourdes
i	il		t	tas	
y	nu		k	cas	
ɔ	bol		b	bon	Plosives sonores
o	eau		d	dans	
ə	le, peu, peur		g	gars	
e	blé		v	vie	Fricatives sonores
ɛ	merci		z	zéro	
u	ou		ʒ	je	
ã	an	Nasales	f	feu	Fricatives sourdes
õ	on		s	sous	
ɛ̃	lin		ʃ	chat	
œ̃	un		n	nous	Nasales
SEMI-VOYELLES			m	ma	
ɥ	huit		ɲ	agneau	
w	oui		l	la	Liquides
j	yeux, baille		r	rue	

Figure 1.5: Codes Phonétiques du Français

et suffixes admissibles. Plus l'application est importante, et plus une solution de "calcul" des mots du langage à partir de morphèmes de base et d'un générateur morphématique, qui permettra de limiter la taille du lexique à mémoriser, devient intéressante. Un algorithme de génération automatique de formes dérivées à partir de radicaux et de désinences n'est pas encore envisageable à l'heure actuelle. De plus, certains groupes de mots, formant des expressions figées de la langue ont une contruction difficilement modélisable et doivent être considérées comme des

entrées lexicales à part entière.

C'est le cas des expressions, parfois familières, de la liste suivante :
– oeil-de-boeuf, pied-de-biche, main courante, pied-à-terre, reine-claude, trompe-l'oeil, avoir un coeur en or, manger son pain blanc, boire du petit lait , casser sa pipe ...

Une approche "globale" de l'unité lexicale conduit à considérer de 40 à 50 000 formes de base. Une approche analytique, décomposant chaque mot en morphèmes, serait beaucoup plus économique pour coder tous les mots, car il suffirait de relever quelques centaines de morphèmes liés par des règles combinatoires dans le cadre d'un analyseur morphématique [GRUAZ 86]. Le modèle de Gruaz comprend deux phases. Le mot est d'abord découpé en unités de base, les morphes, qui seront regroupés en deuxième lieu en morphèmes, unités minimales associant un contenu à une expression. Ces morphèmes peuvent être lexicaux (préfixaux, radicaux, suffixaux) ou grammaticaux (genre, nombre, conjugaison)

Pour illustrer son concept de morphe, Gruaz s'appuie sur les deux listes de mots suivantes :
– excuser, accuser, récuser et abroger, arroger, interroger, proroger

Il en déduit une liste de constituants de base ou morphèmes :
– ex, ac, ré, cus, ab, ar, inter, pro, rog, er.

Ces morphes servent dans un deuxième temps à décomposer les mots :

Par exemple reposer donnera les trois morphes re- -pos- -er

Ils pourront se structurer en morphèmes de deux manières :

1. reposer : repos- -er : "se détendre"

2. reposer : re- -pos- -er : "poser à nouveau"

Les deux morphèmes radicaux (repos- et -pos-) correspondent bien à deux substantifs en français (le repos et la pose). Les deux structurations de cette liste de morphes en listes de morphèmes conduisent à une même orthographe, mais correspondent bien à deux entrées lexicales distinctes.

À ce problème déjà très ardu du regroupement des morphes en morphèmes pour des mots existants, s'ajoute des règles très fines pour déterminer les possibilités et les conditions d'adjonction de préfixaux et suffixaux. Le modèle fourni semble donc satisfaisant pour modéliser la décomposition des mots, mais il n'est pas encore assez complet et robuste pour permettre l'implémentation d'un algorithme de formation des mots à partir de morphes ou de morphèmes.

Voici deux listes de mots obtenus à partir des radicaux -serv- et -plor-, les

préfixes étant déjà choisis (ob- et ex-) :
- *observer, observé, observable, inobservabilité, observatoire, observation, observateur, observance, ...*
- *explorer, exploré, explorable, inexplorabilité, exploratoire, exploration, explorateur, ...*

"*explorance*" n'est pas un mot autorisé par l'académie, mais dans ce cas comment peut-on modéliser cet interdit sans en faire un cas particulier ou doit-on l'accepter et autoriser ainsi un éventuel algorithme de génération à inventer de nouveaux mots, "sur mesure", à l'image des enfants?

La composante phonologique

La représentation phonétique théorique de chaque mot s'avère être une information insuffisante pour une application dès que son vocabulaire dépasse une centaine de mots. On part alors d'une représentation phonologique du mot, qui va se transformer par l'application de règles phonologiques en une forme phonétique particulière.

Il faut traiter les altérations phonologiques à l'intérieur d'un mot, celles dues aux flexions (pluriel, féminin, conjugaison des verbes ...) ou celles liées aux phénomènes de co-articulation entre deux ou plusieurs mots. Nous allons illustrer chacune des ces trois possibilités par quelques exemples simples.

1. Altération à l'intérieur d'un mot

   Nous les classerons à nouveau en deux sous-types :

   - phénomènes de co-articulation proprement dits :
     - assimilation : Un trait phonétique particulier est transmis au(x) phonème(s) voisin(s). C'est en général un trait lié à un mécanisme articulatoire lent comme la nasalité (le voile est lent à se mettre en place par rapport aux autres outils articulatoires) ou le voisement (la mise en vibration ou l'arrêt des cordes vocales ne sont pas instantanés).
       Exemples :
       * rondement :/r/õ/d/∂/m/ã/ → /r/õ/n/m/ã/
       * absolu :/a/b/s/o/l/y/ → /a/p/s/o/l/y/
       * anecdote :/a/n/ϵ/k/d/o/t/ → /a/n/e/g/d/o/t/
       * longuement :/l/õ/g/∂/m/ã/ → /l/õ/ŋj/m/ã/

- fusion : Deux consonnes identiques fusionnent, la résultante étant toutefois de longueur supérieure à la normale. Une autre fusion se produit dans la séquence /i/l/j/ qui devient /i/j/
  Exemples :
  * million :/m/i/l/j/õ/ → /m/i/j/õ/
  * compte-tour :/k/õ/t/t/u/r/ → /k/õ/n/t/u/r/

- Élision du ∂ muet :
  Exemples :
  - petite : /p/∂/t/i/t/ → /p/t/i/t/
  - demande : /d/∂/m/ã/d/ → /d/m/ã/d/

2. Accents régionaux

   Certains locuteurs ont des accents ou des prononciations particulières qui produisent des insertions, des substitutions ou des disparitions de phonèmes. Si certaines de ces altérations sont modélisables pour chaque locuteur, des cas particuliers ou des apparitions épisodiques du phénomène sur un sous-ensemble restreint du vocabulaire peuvent poser des problèmes

   Exemples :

   - septembre :/s/ϵ/p/t/ã/b/r/ → /s/ϵ/t/ã/b/r/
   - candidat : /k/ã/d/i/d/a/ → /k/ã/n/d/i/d/a/
   - quelque : /k/ϵ/l/k/∂/ → /k/ϵ/k/∂/

3. Altérations des finales de mot :

   Ces altérations sont liées aux flexions de déclinaison et de conjugaison des verbes. Elles affectent le plus souvent plusieurs phonèmes. Dès que le lexique devient important, il devient difficile de conserver l'ensemble des formes fléchies pour chaque mot. L'ensemble des radicaux et l'ensemble des désinences associé est tout à fait suffisant pour générer toutes ces formes fléchies.

   En particulier, cette solution devra être mise en oeuvre pour modéliser la conjugaison des verbes réguliers. Si le pluriel d'un nom se résume le plus souvent à l'ajout d'une lettre finale "s", pouvant provoquer une liaison, il ne faudra pas omettre non plus les exceptions du type :

   - "al" → "aux"

   ni les exceptions plus ponctuelles :

   Exemples :

- oeil → yeux
- oeuf : /oe/f/ → oeufs : /ø(s)/ → /ø/
  En effet, la liaison avec oeufs est peu naturelle.

4. Altérations à la jonction de deux mots :

Ces altérations sont de deux types : liaisons et altérations phonologiques proprement dites. Les règles régissant les liaisons sont nombreuses en français, et les phénomènes de coarticulation sont encore plus nombreux qu'à l'intérieur d'un mot :

- liaisons :

  Exemple des chiffres six et dix : trois prononciations sont possibles en fonction du contexte :

  - six enfants → /s/i/z/ã/f/ã/
  - six garçons → /s/i/g/a/r/s/õ/
  - ils sont six → /i/l/s/õ/s/i/s/

  Certaines liaisons sont obligatoires :

  - mes amis → /m/e/z/a/m/i/
  - les enfants → /l/e/z/ã/f/ã/

  Enfin, certaines permettent de lever des ambiguïtés :

  - un savant anglais → /œ̃/s/a/v/ã/ã/g/l/ϵ/
  - un savant anglais → /œ̃/s/a/v/ã/t/ã/g/l/ϵ/

  auront deux significations différentes : le premier sera un savant de nationalité anglaise, le second sera un citoyen anglais, érudit. L'ambiguïté est levée par l'usage : la liaison entre l'adjectif et le nom qu'il qualifie est naturelle, tandis que la liaison d'un nom avec son adjectif qui le suit est rare.

  Ainsi, certaines liaisons sont peu recommandées :

  - l'enfant adopté → /l/ã/f/ã/a/d/o/p/t/e/

  Un autre type d'ambiguïté peut être levé par la présence ou l'absence de liaison : on fait rarement une liaison qui franchit une limite de la structure de surface de la phrase :

  - fais les entrées → /f/ϵ/l/e/z/ã/t/r/e/
  - fais les entrer → /f/ϵ/l/e/ã/t/r/e/

La première liaison est très probable, car "les entrées" constitue un groupe nominal, tandis que dans le second cas une liaison est peu courante.

Ainsi, au pluriel, la séquence (adjectif)–(nom commençant par une voyelle) provoque une introduction très probable du phonème /z/ :

– les petits enfants → /l/e/p/t/i/z/ã/f/ã/

sauf si l'adjectif comporte une consonne finale exempte de troncation , comme direct, exact, neuf, sec, net, vif, tardif, ...

Les cas suivants correspondent à des introductions optionnelles de /z/

– de secs accrochages → /d/∂/s/ϵ/k(/z/)/a/k/r/o/ch/a/gh/
– de tardifs encouragements
→ /d/∂/t/a/r/d/i/f(/z/)/ã/k/u/r/a/gh/∂/m/ã
– de nets îlots → /d/∂/n/ϵ/t/(/z/)i/l/o/

- altérations liées à la co-articulation de deux ou plusieurs mots :

On retrouve les phénomènes déjà observés à l'intérieur d'un mot : voisement, nasalisation, dénasalisation ... avec en plus des phénomènes d'insertion ou d'élisions de phonèmes, en général pour éviter des hiatus ou pour faciliter la prononciation d'une suite de phonèmes incompatibles entre eux.

Exemples :

– insertion d'un e muet (∂)
ours blanc → /u/r/s/ + /b/l/ã/ → /u/r/s/∂/b/l/ã/
– élision de phonèmes
pauvre type → /p/o/v/r/∂/ + /t/i/p/ → /p/o/v/t/i/p/
– dévoisement (et élision)
carnet de chèques → /k/a/r/n/ϵ/ + /d/∂/ + /ch/ϵ/k
→ /k/a/r/n/ϵ/t/ch/ϵ/k/
– nasalisation :
ma grande-tante → /m/a/g/r/ã/d/∂/ + /t/ã/t/
→ /m/a/g/r/ã/n/t/ã/t/
heure et demie → /oe/r/(∂)/ + /ϵ/ + /d/∂/m/i/
→ /oe/r/ϵ/n/m/i/
– dénasalisation :
divin espoir → /d/i/v/$\tilde{i}$/ + /ϵ/s/p/w/a/r/
→ /d/i/v/i/n/ϵ/s/p/w/a/r/

- voisement : ce phénomène semble plus rare et dépend des habitudes et de la région d'origine du locuteur.
  dimanche dernier → /di/m/ã/ch/ + /d/ϵ/r/n/j/ϵ/
  → /d/i/m/ã/gh/d/ϵ/r/n/j/ϵ/

Ces nombreuses altérations phonologiques, malgré leur complexité, ne constituent pas la fin des difficultés du concepteur d'un logiciel de reconnaissance automatique. Elles se combineront ensuite avec les erreurs de décodage acoustico-phonétique pour donner un treillis phonétique réel parfois éloigné du treillis théorique souhaité. Une approche globale, prenant en compte pour la comparaison les phénomènes liés à ces deux causes, reste souhaitable, ces deux phénomènes étant loin d'être totalement disjoints : un locuteur qui parle vite et articule insuffisamment provoquera beaucoup d'erreurs de décodage, tandis que quelqu'un qui "détache" ses mots fera peu d'altérations dans son énoncé d'une part, et verra le décodage de son énoncé facilité d'autre part.

3.2 Le lexique et ses niveaux supra-lexicaux

En parallèle avec ces interactions entre le lexique et des niveaux infra-lexicaux, il existe également de fortes interactions entre le lexique et des niveaux supra-lexicaux, tels que la syntaxe et la sémantique. Ces deux niveaux existent dans la plupart des systèmes de reconnaissance automatique de parole, mais la limite entre ces deux bases de connaissance n'est pas toujours bien connue.

1. La syntaxe : Elle regroupe l'ensemble des informations contenues dans la grammaire et elle s'exprime sous forme de règles de construction de phrases. Cependant, selon la finesse des règles contenues dans la grammaire, cela peut conduire à des représentations très diverses :

   (a) `<Ph aff> ::= <groupe sujet><groupe verbal><complement>`

   (b) `<Q_obt_pasp_8> ::= <ou><faut-il s'adresser><pour obtenir> <un passeport>`

   Jusqu'à un passé récent, la reconnaissance de phrase s'appuyait presque exclusivement sur une telle description syntaxique du langage à reconnaître. La non tolérance aux erreurs de ces systèmes a conduit à réduire l'importance de cette base de connaissance et à rééquilibrer les nouveaux systèmes autour de trois bases de connaissances, l'une syntaxique, une deuxième sémantique et une dernière pragmatique. La syntaxe est surtout utilisée pour limiter à

chaque étape de l'analyse le nombre d'hypothèses à examiner pour les processus descendants ou pour sélectionner dans le treillis de mots fourni par le niveau lexical les phrases syntaxiquement correctes. De plus, la structure syntaxique ainsi obtenue va servir au processus d'interprétation.

L'obtention du modèle syntaxique n'est pas immédiate dans le cadre d'un langage naturel. Pour un langage artificiel, toutes les phrases correspondant à une grammaire définie a priori seront réputées correctes. La même méthode peut fournir des résultats pour un langage naturel : on définit de façon empirique une grammaire, qui va engendrer un sur-langage du langage réel à traiter, en s'appuyant par exemple sur la syntaxe du fançais de base. On essayera ensuite d'éliminer les phrases *correctes* pour cette syntaxe, mais incorrectes de fait car n'ayant aucun sens, en utilisant d'autres connaissances, par exemple sémantiques ou pragmatiques.

2. La sémantique est définie par les linguistes comme la relation entre le signifiant et le signifié. Elle est plus souvent définie de façon négative, comme regroupant toutes les informations non-syntaxiques et non-lexicales. Elle présuppose donc l'existence de ces deux sources d'informations et regroupe le plus souvent l'ensemble des informations liées au sens et interdisant certaines constructions de phrases pourtant syntaxiquement possibles. Toutefois, ces interdictions sont très dépendantes de l'application traitée.

   La plupart des systèmes informatiques existants comme KEAL ou MYRTILLE I [MERCIER 77,PIERREL 75] se contentent d'une syntaxe minimale, mais figée, pour la reconnaissance. Cela est en effet suffisant pour des applications dont le vocabulaire est restreint, ne dépassant guère la centaine de mots ou racines, tels que la commande de systèmes automatisés. Toutefois, une telle approche se révèle très vite insuffisante si l'on veut traiter des vocabulaires plus étendus, de l'ordre du millier de mots par exemple pour une application finalisée du type "renseignements administratifs". La sémantique va jouer un double rôle dans ce type d'application : il faudra faire un filtrage lexical pour restreindre le nombre d'hypothèses à examiner en cours d'analyse, mais aussi permettre la construction d'une représentation sémantique, correspondant à la structure profonde d'un énoncé et directement interprétable par un module chargé de prendre en charge la demande ainsi formulée par l'utilisateur. Trois modèles permettent la prise en compte de telles informations sémantiques dans les systèmes actuels de reconnaissance : les grammaires sémantiques, les réseaux sémantiques et enfin les modèles hiérarchisés de traits sémantiques associés aux mots du lexique.

[DEWEZE 81,FILLMORE 68,KAYSER 84].

Les différentes bases de connaissances que nous venons d'examiner sont suffisantes pour des applications simples, à vocabulaire limité. Toutefois, les systèmes de reconnaissance engendrés à la suite d'une telle approche sont contraignants et soit peu robustes aux erreurs, soit fastidieux d'usage. Or, le but principal visé dans l'ajout d'une interface vocale était d'augmenter la convivialité du système vis-à-vis d'usagers peu expérimentés. Si de tels systèmes sont utiles dans certaines applications bien précises –commandes d'un microscope électronique pour des opérations en micro-chirurgie par exemple– , ils offrent peu d'intérêt dans les autres cas.

Une nouvelle approche permettant un usage plus souple et plus naturel de l'interface vocale sera centrée autour d'un module convivial de dialogue. Celui-ci sera chargé de gérer un échange en langage quasi-naturel, à la fois souple et efficace et acceptant des sujets non entrainés. Il devra donc être capable à la fois de contrôler le canal de communication, d'utiliser l'historique du dialogue pour résoudre les ellipses et les anaphores et de gérer la tâche et le dialogue lui-même pour répondre aux questions posées et satisfaire ainsi les requêtes en provenance de l'usager. Cette approche, plus ambitieuse et à plus long terme, est faite au CRIN et un système de dialogue est en cours d'élabaration dans notre équipe. Nous le décrirons de façon plus détaillée dans le chapitre III. Il est clair que l'être humain en situation de dialogue utilise à la fois la nature du dialogue et son historique pour aider à l'interprétation (contextuelle) des énoncés qu'il entend. Une nouvelle composante "*dialogue*" nous semble donc indispensable dans un système de compréhension et d'interprétation de parole et l'architecture que nous proposons [CARBONELL 85] devrait faciliter la mise en oeuvre de systèmes plus conviviaux, permettant l'accès à des non-spécialistes avec des interfaces orales, en langage quasi-naturel.

Chapitre 2

Informations nécessaires au niveau lexical. Traitements associés.

1 Introduction

Le lexique est sans doute la partie la plus évocatrice d'un système de compréhension de parole, pour une personne non-spécialiste et qui se documente sur le sujet. Dès l'école élémentaire, chaque enfant est familiarisé avec la notion de mot et cette notion apparaît avant celle de grammaire. Si dans un système de taille réduite, le lexique peut se réduire à la liste des mots présents dans le langage, un grand nombre d'informations supplémentaires sont nécessaires dès que le nombre de mots compréhensibles par le système atteint le millier.

Dans un tel système, on continuera d'appeller lexique l'ensemble des mots connus du système et l'ensemble des informations relatives à ces mots.

Le passage par le mot restera priviligié dans tout système de reconnaissance : l'entité lexicale servira une première fois de façon ascendante, pour proposer des mots reconnus a priori, sans autre information que les résultats du décodage acoustico-phonétique et ceux du module de détection d'indices prosodiques; il servira une seconde fois de façon descendante, quand les analyseurs voudront tester la présence de mots hypothèses, ces hypothèses étant émises à l'aide d'une grammaire locale et du contexte déjà reconnu.

Comment peut-on savoir a priori quel est l'ensemble des mots que le système devra connaître pour mener à bien sa tâche? Cet ensemble de mots sera bien sûr dépendant de cette tâche. Mais existe-t-il un sous-lexique constant, dont les mots sont communs à toute une classe de tâches? Nous allons essayer de répondre à ces questions dans ce chapitre.

2 Définition du vocabulaire

Plusieurs approches permettent de définir le langage autorisé pour communiquer avec un système de compréhension. Comme un système informatique ne peut pas contenir l'ensemble des informations linguistiques utilisées par l'homme lorsqu'il parle de façon entièrement naturelle, il semble exclu de comprendre automatiquement un dialogue entièrement naturel, sans but prédéfini. Nous nous limiterons donc à la compréhension d'énoncés émis dans un cadre de dialogue finalisé, dont le but est de mener à bien une tâche déterminée.

2.1 Grammaire a priori

Les langages les plus simples à reconnaître pour un système informatisé correspondent aux langages entièrement définis à l'aide d'une grammaire qui fixe de façon brutale la syntaxe admise pour s'adresser au système, ainsi que l'ensemble des mots ou terminaux autorisés. C'est le premier niveau de langage que les systèmes automatiques de compréhension ont permis à leurs usagers. Un tel langage sera qualifié d'artificiel, même s'il est à consonnance naturelle et à chaque application sera associée la définition d'un langage artificiel qui lui sera spécifique.

Ce type de langage paraissant trop contraignant, les concepteurs de systèmes automatisés se sont attachés à définir des langages plus souples. En particulier, pour être réellement conviviaux, ces langages doivent permettre à un usager de s'exprimer selon ses habitudes. Si le langage autorisé reste un langage construit à partir d'une grammaire descriptive, on autorise le locuteur à faire des références anaphoriques, à élider une partie d'un énoncé et à produire des phrases dont la syntaxe n'est pas rigoureusement correcte. Dès 1975, des solutions permettant d'interpréter certains formes elliptiques ont été proposées dans le système SRI (Standford Research Institute) [WALKER 78].

Exemples de dialogue permis dans ce système :

Usager1 : *Est-ce que les Britanniques possèdent tous les sous-marins?*

Système1 : ....

Usager2 : *et les porte-avions?*

Système2 : ....

et

Usager1 : *Est-ce que les Britaniques possèdent les porte-avions?*

Système1 : ....

Usager2 : *Et les États-Unis?*

Système2 : ....

Pour résoudre ces ellipses, le système renferme une représentation de l'univers de l'application, sous forme de réseau sémantique [COULON 86]. Chaque objet référencé en cours de dialogue devient un noeud de ce réseau, qui va servir à compléter l'énoncé suivant s'il est elliptique.

Deux approches pour définir un langage plus naturel sont étudiées actuellement.

La première est théorique et donne la définition d'un langage quasi-naturel en énonçant une liste de propriétés que doit vérifier un tel langage.

Ainsi Jean-Pierre Desclés [DESCLES 82] nous cite entre autres :

- c'est un langage artificiel, donc construit
- c'est une restriction d'un langage naturel, de manière à être facilement descriptible de façon formelle.
- De plus, c'est une sous-langue de la langue naturelle et il doit donc en vérifier les propriétés essentielles.

  Pour Jean-Pierre Desclés, les propriétés essentielles des langages naturels sont entre autres :

  1. Existence d'une fonction personnelle. Elle pourra s'exprimer soit par des pronoms personnels, soit par d'autres mécanismes, incluant des procédés lexicaux.
  2. Faculté de pouvoir rapporter des énonciations, de façon directe ou indirecte.
  3. Possibilité d'acquérir et représenter le temps par des procédés grammaticaux.
  4. Existence de particules déictiques.
  5. Existence de procédés anaphoriques.

Les propriétés essentielles retenues par Jean-Pierre Desclés ne semblent pas être celles que nous attendons d'un langage permettant un dialogue simple, dont le seul but sera par exemple l'obtention d'un renseignement figurant dans une base de données. En particulier, il est inutile de rapporter des énonciations dans

un dialogue oral finalisé. Cette définition des langages quasi-naturels correspond donc plus à un outil théorique qu'à un modèle de description d'un langage oral de communication homme-machine.

La seconde approche consiste à définir le lexique en déterminant de façon pratique l'ensemble des mots succeptibles d'être utilisés dans le cadre de l'application visée. La manière la plus simple pour déterminer cet ensemble est de constituer un corpus de dialogues portant sur l'application, et d'en déduire les mots utiles à cette application. Cette approche fait l'objet du paragraphe suivant.

2.2 Définition à partir de corpus

La seconde approche pour définir l'ensemble des mots à introduire dans le lexique est plus pragmatique : pour déterminer le sous-langage utilisé dans un système de dialogue finalisé, il suffit de constituer un corpus suffisamment important, en plaçant le locuteur dans la situation désirée. Comme le grand public reste peu ou mal informé des possibilités réelles des systèmes de reconnaissance actuels (la science fiction contribue elle aussi à rendre crédible une machine qui comprendrait un dialogue naturel en temps réel), un simple vocodeur qui déformera la voix d'un compère suffira à persuader la plupart des sujets testés qu'ils sont en communication avec une machine. Cela permet de recueillir des corpus offrant le plus grand intérêt pour nos recherches.

En effet, chaque application conduira à des productions d'énoncés de type spécifique : un même individu ne parle pas du tout de la même façon selon qu'il demande un renseignement par téléphone, ou qu'il commente le match sportif de la veille. A chaque application va être ainsi associé de façon expérimentale un sous-lexique bien spécifique. L'analyse statistique des mots utilisés au cours d'un nombre suffisant de transactions émises par des usagers et enregistrées dans ce but fournira donc le lexique de l'application. Différentes études de corpus ainsi réalisées ont mis en évidence plusieurs propriétés de tels lexiques :

- Même si l'application peut sembler très ambitieuse et complexe en première approximation, le lexique utilisé est très typé. Des comptages de mots (ou plutôt de formes orthographiques, qui devront être vérifiées en cas d'ambiguïté comme *porte*) dans un corpus donné font apparaître deux sous-lexiques :
  - Un sous-lexique de mots outils, dont la fréquence en pourcentage (rapport du nombre d'occurrences de ce mot / nombre total de mots prononcés) va rapidement se stabiliser.
  - Un sous-lexique de mots propres à l'application, très techniques, appa-

raissant de façon très épisodique. Ce sous-lexique va grandir en taille avec le nombre de transactions recueillies. Souvent, une personne aura son sous-lexique propre, qui variera selon son niveau de langage et selon l'expertise qu'elle possède, si la conversation qu'elle poursuit est à but technique (obtention d'un renseignement administratif par exemple).

Par exemple, dans le corpus recueilli au CRIN [ROUSSANALY 86] concernant des renseignements administratifs du même type que ceux fournis dans les pages roses d'un annuaire, on peut noter les éléments suivants :

* Nombre de transactions : 42
* Nombre d'énoncés total : 1153
* Nombre moyen d'énoncés par transaction : 27
* Nombre total de mots : 13136
* Nombre de mots du lexique ainsi constitué : 1377

Pour le corpus constitué par Guy Deville à la cellule action-travail du ministère de l'emploi et du travail de Belgique [DEVILLE 87a] et concernant là aussi des demandes de renseignements administratifs, les chiffres sont les suivants :

* Nombre de transactions : 100
  (en réalité sélection des 100 transactions les plus *intéressantes*)
* Nombre d'énoncés total : 4181
* Nombre moyen d'énoncés par transaction : 41
* Nombre total de mots : 40896
* Nombre de mots du lexique ainsi constitué : 2316

Etude de quelques mots outils :

* le mot *dire* :
  22 occurrences dans le corpus CRIN sur 13136 mots prononcés :
  fréquence : 0,17 %
  70 occurrences dans le corpus de Namur sur 40896 mots prononcés :
  fréquence : 0,17 %
* le mot *voulez* :
  13 occurrences dans le corpus CRIN sur 13136 mots prononcés :
  fréquence : 0.1 %
  37 occurrences dans le corpus de Namur sur 40896 mots prononcés :
  fréquence : 0.09 %

(NB : 12 occurrences pour le CRIN donneraient un taux identique : 0.09 %)

* le mot *quelle* :
  4 occurrences dans le corpus CRIN sur 13136 mots prononcés :
  fréquence : 0.03 %
  11 occurrences dans le corpus de Namur sur 40896 mots prononcés :
  fréquence : 0.03 %
  (NB : Un troisième chiffre ne serait pas du tout significatif, une seule occurrence en plus ou en moins impliquant une variation de 0.01 % pour le corpus du CRIN. Le deuxième chiffre donné est donc lui-même sujet à caution)
  Toutefois des mots comme *oui* et *non* n'ont pas les mêmes fréquences d'apparition dans les deux corpus. Il y en a presque deux fois plus dans le corpus réel (Namur) que dans le corpus obtenu par simulation au CRIN :
  Pour le mot *non* par exemple :
  71 occurrences dans le corpus CRIN sur 13136 mots prononcés :
  fréquence : 0.54 %
  319 occurrences dans le corpus de Namur sur 40896 mots prononcés :
  fréquence : 0. 78 %

Les questions posées à Namur étaient sans doute plus fréquemment fermées :

* *Ai-je droit à quelque chose?*
* *Y-a-t-il un préavis?*
* *On ne peut rien toucher dans ce cas, n'est-ce-pas?*

Ce type de résultats fournit donc à la fois l'ensemble des mots utiles pour une application donnée, mais aussi la fréquence moyenne d'apparition des mots outils dans une telle application.

Ces deux corpus ont été constitués en utilisant des transactions réelles, c'est-à-dire que les locuteurs parlaient à une autre personne. Est-ce que des utilisateurs vont se comporter de manière nettement différente en face d'une machine? La troisième expérience que nous allons étudier répond en partie à ce type de questions.

- Un troisième test du même type a donc été réalisé [AMALBERTI 86] pour tester le comportement des utilisateurs d'un tel système. Plusieurs scénarii ont été testés pour vérifier certaines hypothèses. Ce test est double : il

comporte des transactions naturelles, telles que celles des deux premiers corpus, et des transactions pour lesquelles les locuteurs sont informés qu'une machine automatique de traitement va essayer de les renseigner. Les sujets testés dans le cadre *"machine"* affirment avoir essayé de limiter leur vocabulaire et leurs reprises au milieu d'un énoncé. Cependant les résultats portant sur 89 dialogues sont formels : au lieu d'appauvrir le vocabulaire utilisé, ces 9 personnes ont au contraire utilisé plus de mots que leur homologue du premier groupe : ils ont utilisé en moyenne 396 mots pour leur première session, contre 341 pour les sujets s'adressant à une opératrice (ou plutôt le sachant).

L'explication fournie dans [AMALBERTI 86] semble plus que crédible :

Devant l'incompréhension d'un auditeur (étranger par exemple), le locuteur ne va pas répéter sa phrase dans les mêmes termes, mais imputer l'incompréhension constatée au choix du vocabulaire *"Cette attitude conduit, devant une demande de répétition, à ne pas répéter la même phrase (plus lentement), mais à reformuler celle-ci"*.

Par contre, le nombre d'interjections et de reprises est nettement inférieur chez les sujets du groupe *machine*. Ceux-ci ont donc bien modifié leur comportement.

Cette remarque coïncide avec le comportement et les résultats obtenus en reconnaissance par les locuteurs devant notre machine, à la fois pour des démonstrations et pour les tests de nos programmes.

- Pour un locuteur donné et utilisant de façon habituelle le système, ses résultats vont s'améliorer au point de vue décodage au cours du temps, sans modifier le programme ou l'adapter au locuteur, car c'est au contraire le locuteur qui s'adapte à la machine. Cette adaptation sera d'autant plus rapide si cette personne est habituée à parler à un large auditoire, donc à articuler, ou habituée à parler dans un microphone.
- Un locuteur ne répondant à aucune de ces conditions aura souvent des taux de reconnaissance catastrophiques pour ses premiers énoncés.
- Une telle restriction serait bien sûr à prendre en compte à l'avenir avant la diffusion au grand public de tels systèmes.

Nous n'avons pas l'ambition de tirer toutes les conclusions possibles de ces tests, nos compétences en linguistique ou en psycholinguistique étant très limitées. L'étude linguistique des deux premiers corpus est en cours et fera l'objet de la

thèse de Guy Deville. Toutefois, ce genre de résultats va influencer énormément notre démarche, à la fois pour constituer automatiquement un lexique, sous le double aspect mots présents et informations attachées à ces mots, mais aussi structuration de ces informations. Nous allons essayer de définir les informations et leur structuration d'après les premiers résultats fournis par Guy Deville [DEVILLE 86,DEVILLE 87b].

3 Informations indispensables

Les informations minimales dans tout lexique en vue d'une compréhension orale sont celles de type phonétique et phonologique. En effet, pour chaque mot, on doit être capable de comparer une transcription phonétique théorique , avec un morceau de treillis pour évaluer une plausibilité de présence de ce mot.

La description phonétique exacte devient toutefois très vite insuffisante si la taille du vocabulaire s'agrandit, ou n'est pas toujours facile à définir Plusieurs solutions se présentent au concepteur du système à ce niveau :

- Si la taille du vocabulaire reste raisonnable (jusqu'à trois cents mots peut-être), on peut prévoir a priori toutes les variations phonologiques que vont subir les mots du vocabulaire, en fonction de leurs différents contextes admissibles et de règles de coarticulation et d'assimilation bien connues des phonéticiens. A chaque mot sera alors associée une liste des diverses réalisations phonétiques que le système admettra.

  Exemple :

  calamar → /k/a/l/a/m/a/r/

  calmar → /k/a/l/m/a/r/

  Pour le concepteur non phonéticien, cette technique présente un gros avantage : il pourra opérer par apprentissage, en utilisant le corpus ayant servi à constituer le lexique : il suffit de le considérer comme une suite d'énoncés à reconnaître, et d'associer à chaque mot les différents décodages fournis par le système lui-même. Cela permet de précompiler simultanément deux types d'informations différentes :

  - Les informations phonologiques
  - Les erreurs du système de décodage. En effet, celles-ci ne sont pas toujours explicables de façon immédiate, mais cela n'empêche en aucune façon leur apparition répétitive dans un même contexte, pour un locuteur donné. Le décodage semble très stable si le contexte l'est : ainsi, le

mot *morin* en fin d'énoncé, (par exemple "*copie le fichier essai dans la directory morin* ") pour un des locuteurs habitué au système reçoit très souvent comme étiquettes à l'issue du décodage acoustico-phonétique : */m/õ/b/õ/*

- Une telle approche est inacceptable dès que la taille du vocabulaire devient trop importante, pas tant à cause de la place occupée que du temps nécessaire à l'élaboration de cette représentation.

  De plus, une modification du niveau de décodage entraînera une altération grave du modèle : il faudra en principe recommencer la phase d'apprentissage. Tous les avantages apportés par la conception modulaire du système seraient alors perdus.

  Une deuxième approche peut elle aussi conduire à ce type de représentation, mais sans que celle-ci tienne compte du niveau de décodage. Toutes les altérations phonologiques prévisibles sont représentées explicitement comme autant de représentations phonétiques d'un mot donné. La recherche se fait alors de façon exacte sur le treillis proposé par le niveau bas, et cela permet l'utilisation des algorithmes classiques de pattern matching, très performants.

  Toutefois, une telle approche est trop rigide, de petits écarts par rapport à toutes ces prévisions d'altérations existeront toujours. Pour prendre en compte ces petits écarts, un algorithme de recherche admettant des différences entre la chaîne cherchée et la représentation trouvée devient obligatoire.

Deux méthodes pour écrire ces algorithmes approchés vont nous conduire à deux nouveaux types de représentations des connaissances phonologiques, selon le but poursuivi :

- Si l'on recherche des mots avec un bon degré de confiance, on va garder une structure du mot en terme de macro-classes ou patrons phonétiques, qui sera caractéristique d'une classe de mots. La recherche se fera alors à l'image de cette représentation, en deux temps :

  - Une première passe très rapide recherchera le patron phonétique de façon exacte. Ici encore tous les algorithmes très performants de pattern matching vont s'appliquer.
  - Quand le patron d'un mot est localisé, on va calculer un score, soit de ressemblance, soit de dissemblance, entre la représentation phonétique

exacte de ce mot et le treillis proposé par le module de décodage acoustico-phonétique sur cette plage de signal. Comme cette comparaison est limitée dans l'espace à la plage trouvée, tous les essais d'altérations phonologiques, de substitution, d'omission et d'insertion pourront être étudiés.

Cette approche semble prometteuse, mais la grande difficulté réside dans le choix de structuration des phonèmes du mot pour lui associer un patron phonétique stable pour le niveau de décodage actuel. De légers progrès de ce niveau, surtout au niveau de la segmentation, rendraient ce choix plus facile. Nous reviendrons sur ce point dans le chapitre 4, qui présente les résultats obtenus pour ce genre d'approche.

- La deuxième façon d'aborder les choses est plus réaliste à ce jour. Sur certaines plages de signal, il est en effet impossible de trouver un invariant dans le décodage associé à un mot donné. Toute recherche exacte s'avère donc infructueuse. La solution la plus simple dans ce cas est d'utiliser une représentation phonétique parfaite du mot et de calculer une *pseudo-distance* entre cette représentation théorique et la représention à identifier, toutes les déformations possibles , qu'elles soient phonologiques ou liées aux limites actuelles du décodage (erreur de segmentation ou d'étiquetage), étant prises en compte de façon algorithmique.

Ces algorithmes pourront utiliser des résultats statistiques concernant la fréquence d'une erreur et donc sa gravité relative. Par exemple :

- Les *"l"* sont des phonèmes brefs, souvent omis par le décodage. La pénalité d'une omission sera limitée

- Les noyaux vocaliques sont détectés de façon remarquable par l'algorithme de segmentation. Une élision sera à pénaliser lourdement.

- Les trois plosives voisées sont encore trop souvent confondues entre elles (*/b/, /d/, /g/*). Une telle confusion sera bénigne en cours de comparaison.

Pour toutes ces informations assez peu contextuelles, une représentation simple reste très efficace :

- Pour les substitutions : une matrice de confusion permet de traiter ce problème.

- Pour les élisions : Une table fournissant la pénalité à associer à un phonème manquant constitue déjà une bonne approximation. Toutefois, un affinage consistant à prendre en compte le contexte peut s'avérer payant.
- Pour les insertions, la principale information à prendre en compte, outre la nature du phonème inséré, est sa longueur : un *"l"* très court est souvent le résidu d'une voyelle d'avant (i, é ou ai) coupée de façon trop brutale. Ce même phonème sera toutefois à prendre au sérieux si sa longueur est proche de la durée vocalique moyenne de l'énoncé en cours d'analyse.

4 Informations complémentaires

4.1 Informations syntaxico-sémantiques

Dès que la taille du vocabulaire augmente, il convient de limiter l'explosion combinatoire au niveau des analyseurs syntaxiques, liée au nombre de mots candidats à un moment donné. Les seules informations phonétiques et phonologiques s'avèrent donc rapidement insuffisantes pour sélectionner de façon satisfaisante les entrées lexicales convenables.

Deux difficultés se présentent au concepteur du niveau lexical : le lexique devra être vu de façon différente par chacun des modules qui fait référence à une unité *mot*. Toutefois, un même mot référencé à plusieurs reprises de façon indépendante (par exemple émis comme hypothèse d'une part au niveau lexical, d'autre part au niveau sémantique), ne devra être associé qu'à un seul score, mais celui-ci devra se renforcer à chaque nouvel apport d'information, si cette information est obtenue de manière indépendante par rapport à toutes celles déjà prises en compte. C'est cette indépendance qui reste la partie la plus difficile à tester. Si pour le niveau *bas*, le test est facile à effectuer (l'information est booléenne : pour chaque phonème, il est facile de regarder s'il est déjà utilisé dans un mot pris en compte par le niveau supérieur dans l'analyse en cours, et, pour un mot, il est encore plus simple de contrôler si l'on a tenu compte de la vérification phonétique dans son score), pour les niveaux syntaxiques et sémantiques, il faudra vérifier qu'une même information n'aboutit pas à une même entrée lexicale par deux chemins différents. A l'origine de ces chemins se trouvent des informations élémentaires qui peuvent être de deux types :

- Des traits syntaxiques : catégorie grammaticale, genre, nombre...
- Des traits sémantiques, hiérarchisés, reposant sur une grammaire de cas. L'ensemble de ces traits sémantiques sont associés à des primitives en nom-

bre limité mises en évidence par [DEVILLE 87b]. A chaque primitive sont associés un certain nombre de traits de base.

Sept traits principaux semblent suffisants pour décrire la plupart des concepts manipulés dans notre sous-langage, relatif aux demandes de renseignements administratifs :

- ANIMÉ (ani+ ou ani-) : Cette primitive est associée à une intervention d'un être animé
- CONTRÔLE (ctr+ ou ctr-) : L'action sera contrôlée par l'agent
- DYNAMIQUE (dyn+ ou dyn-) : Cette primitive impose un changement de situation
- ESPACE (spa+ ou spa-) : Cette primitive est associée à l'expression d'une dimension spatiale
- MOUVEMENT (mvt+ ou mvt-) : La primitive d'action concernée exprime un mouvement (ou non)
- PRODUCTION (pro+ ou pro-) : La primitive sous-entend le transfert d'un objet d'un donneur vers un récepteur
- TRANSITIVITÉ (trs+ ou trs-) : La primitive admet aux moins deux arguments, dont l'un est l'objet de surface du prédicat dérivé de cette primitive (ou pour trs- : la primitive ne peut pas avoir un argument objet de surface du prédicat dérivé).

- Ces traits de bases vont servir à classifier des prédicats de type verbe en primitives de 16 genres différents.

Les verbes sont d'abord divisés en trois types de base :

1. Les verbes exprimant une **Action**
2. Les verbes décrivant un **État**
3. Les verbes traduisant l'exécution d'un **Processus**

Pour les prédicats de type **action**, tous possèdent les trois traits de base ctr+, dyn+ et ani+. Les quatre autres traits vont nous permettre de dégager sept primitives :

- Les deux premières actions sont des mouvements, caractérisés par le trait mvt+

- Mouvement1 sera intransitif (trs-)

  Exemples : *aller, se rendre, courir*

- Mouvement2 sera transitif (trs+)

  Exemple *apporter*

- Deux autres prédicats d'actions se différencient des précédents par l'absence de mouvement (mvt-) et sont tous deux transitifs (trs+) :

  - Actobjet

    Exemples : *signer, oblitérer*

  - Actanimé

    Exemple *vacciner*

- Les deux prédicats d'actions suivants font référence à des échanges d'objets :

  - Échange-production modélise une action qui aboutit au transfert (physique ou mental) d'un *objet*, d'une première entité animée qui possède le contrôle de l'action, vers un récipiandaire, animé lui aussi.

    Exemple : *donner*

  - Échange-obtention est très proche de la primitive précédente : Seul le contrôle de l'action est déplacé du donneur au récepteur.

    Exemple : *recevoir*

- Les autres actions faites par des sujets animés et qui ne sont donc ni transitives (trs-) ni des actions exprimant un mouvement (mvt-) seront qualifiées comme **atransitives**

  Exemple *voter*

Le deuxième type de primitives concerne les **états** :

- Le premier sous-type sera **Location.**

  On retiendra deux sous-ensembles :

  - Location1 correspondra à des verbes non transitifs (trs-), n'exprimant pas de mouvement (mvt-), non contrôlé et non dynamique (ctr-, dyn-). Elle exprime une position d'une entité quelconque dans l'espace (spa+) :

    Exemples : *se trouver, se situer*

- Location2 correspond à des verbes transitifs (trs+), qui n'expriment pas de mouvement (mvt-), mais expriment un état contrôlé par une entité animée (ctr+, ani+), cet état ayant une dimension spatiale (spa+)
  Exemple : *garder*

- Une autre primitive exprimera l'**extension**. Sa seule différence avec Location2 est qu'aucune dimension spatiale n'apparaît (spa-) :
  Exemple : *savoir*

- Deux autres primitives expriment un statut. Nous les différencions sur l'objet ayant ce statut (animé [ani+] ou inanimé [ani-])

  - Statut1 aura donc les traits trs-, mvt-, ctr-, dyn- et spa-, mais ani+. Il exprime une propriété d'un animé :
    Exemples : *Être belge, être majeur*
  - Statut2 exprime une propriété d'un inanimé (ani-)
    Exemple : *être valide*

- Deux primitives encore expriment des mesures. La distinction porte encore sur le qualifié, animé on non :

  - Mesure1 fait référence à un verbe intransitif, n'exprimant pas de mouvement mais un état incontrôlé (trs-, mvt-, ctr-, dyn-), mais qualifiant un animé (ani+) et n'exprimant pas une localisation spatiale (spa-)
    Exemple : *Être agé de*
  - Mesure2 ne se différencie que par le trait (ani-) : il qualifie un inanimé :
    Exemple : *coûter*

Le dernier grand type correspond à des processus spontanés. Les deux primitives de ce type seront là encore différenciées sur le trait animé ou non-animé :

- Process1 correspond à des verbes transitifs (trs+), n'exprimant pas de mouvement (mvt-), mais un processus (ctr-, dyn+) s'appliquant à un animé (ani+)
  Exemples : *changer* de nationalité, *devenir* majeur, *naître*

- Processus2 a le même profil, mais s'applique à des objets inanimés (ani-)
  Exemples : le prix *augmente* tous les ans, ma carte d'identité *a brulé*

4.2 Grammaire de cas induite

Toutes ces primitives verbales, classifiées en fonction des traits de base précédemment définis, vont être à la base de la définition d'une grammaire de cas adaptée à nos besoins. Les différents cas distingués dans notre système vont donc découler des primitives de type verbe que nous venons de recenser.

En voici une description succincte, illustrée à chaque fois par un exemple.

- **Agent** (agt) Exemple : JE vais à la préfecture
- **Patient** (pat) Exemple : JE suis français
- **Objet** (obj) Exemple : J'ai perdu MON PASSEPORT
- **Bénéficiaire** (bén) Exemple : Je TE donne l'autorisation
- **Source** (src) Exemple : Je reviens de PARIS
- **Location** (loc) Exemple : J'habite À VILLERS-LES-NANCY
- **Temporel** (tmp) Exemple : Je l'ai fait HIER
- **Instrument** (ins) Exemple : L'arbre a été cassé PAR LE VENT
- **Moyen** (moy) Exemple : Je t'envoie cela PAR LE PREMIER COURRIER
- **Mesure** (mes) Exemple : Le timbre fiscal vaut DEUX CENT CINQUANTE FRANCS
- **But** (but) Exemple : Je voudrais un visa POUR VISITER MOSCOU
- **Condition** (cond) Exemple : Que faut-il de plus, SI L'ON EST MINEUR?
- **Cause** (caus) Exemple : Je voudrais une nouvelle carte d'identité PARCE QU'ELLE EST PÉRIMÉE

L'ensemble de ces cas permet de définir les primitives verbales précédentes en fonction de leur présence obligatoire (+), optionnelle (o), ou interdite (-).

Cela nous donne le tableau descriptif suivant :

P. Verb.	agt	pat	obj	bén	src	des	loc	tmp	ins	mes	moy	but	cond	caus
Mvmt1	+	-	-	-	+	+	-	o	-	-	o	o	o	o
Mvmt2	+	-	+	-	+	+	-	o	-	-	o	o	o	o
Actanimé	+	-	-	+	-	-	o	o	o	o	o	o	o	o
Actobjet	+	-	+	o	-	-	o	o	o	-	o	o	o	o
Echprod	+	-	+	+	-	-	o	o	-	-	o	o	o	o
Echobt	o	-	+	+	-	-	o	o	-	-	o	o	o	o
Atrans	+	-	-	o	-	-	o	o	-	-	o	o	o	o
Location1	-	+/-	-/+	-	-	-	+	o	-	-	-	-	o	o
Location2	-	+	+	o	-	-	o	o	-	-	-	-	o	o
Extension	-	+	+	-	-	-	-	o	-	-	-	-	o	o
Statut1	-	+	-	-	-	-	-	o	-	-	-	-	o	o
Statut2	-	-	+	-	-	-	o	o	-	o	-	-	o	o
Mesure1	-	+	-	-	-	-	-	-	-	+	-	-	o	o
Mesure2	-	-	+	o	-	-	o	-	-	+	-	-	o	o
Process1	-	+	o	-	-	-	o	o	-	-	-	-	o	o
Process2	-	-	+	-	-	-	o	o	o	-	-	-	o	o

Pour les primitives non verbales, tels les noms ou les pronoms, elles vont être caractérisées par d'autres traits sémantiques. Par exemple, pour une application "Renseignements administratifs", Guy Deville [DEVILLE 87b] préconise l'utilisation des huits traits suivants : *Animé, Humain, Docum (L'objet est un document administratif), Actanimé (le référencé est un acte administratif), Temps, Lieu, Age, Coût.*

Les traits *Animé* et *Humain* peuvent recevoir deux valeurs ("+" ou "-"), les autres ne pourront être que des qualifiants positifs.

Le trait *Animé* a aussi un statut priviligié et sa valeur va déterminer deux sous-ensembles parmi les autres traits, la validité de chaque sous-ensemble étant associée pour le premier à +Animé et à -Animé pour le second :

Le premier sous-ensemble est le singleton {*Humain*}, le second contient les six traits { *Docum, Actadmi, Temps, Lieu, Age* et *Coût*}

Exemples : (Toujours pour une application de type "Renseignements administratifs")

+Animé +Humain	:	vous, père, tuteur, commissaire
+Animé -Humain	:	mairie, secrétariat, tribunal, préfecture
-Animé	:	problème
-Animé +Docum	:	carte d'identité, certificat
-Animé +Actadmi	:	procédure, examen, délivrance
-Animé +Temps	:	1988, jour, mois
-Animé +Lieu	:	Villers-les-Nancy, commissariat, préfecture
-Animé +Age	:	majeur, an
-Animé +Coût	:	Francs

Pour cette application, la hiérarchisation est pauvre, car l'ensemble des noms autres que les mots outils de la langue sont techniques et propres au domaine.

Ces traits sémantiques et la structure casuelle associée aux différentes primitives verbales vont constituer l'essentiel des informations sémantiques.

En cours d'analyse, ces informations seront renforcées par des informations syntaxiques. Toutefois, il est exclu dans un système de dialogue ORAL de se limiter à des énoncés syntaxiquement parfaits. Les règles syntaxiques utilisées devront être souples et leur usage principal sera de faire des vérifications locales. En effet, il suffit d'une hésitation, d'une reprise/reformulation en cours d'énoncé pour mettre en échec un analyseur syntaxique trop rigide. Comme ces phénomènes sont fréquents en parole, les analyseurs syntaxico-sémantiques doivent en tenir compte : en cas d'impossibilité de mener à son terme l'analyse syntaxique, on pourra continuer la tentative de compréhension de l'énoncé courant si les informations sémantiques restent cohérentes.

4.3 Représentation syntaxico-sémantique choisie

De nombreux modèles de représentation pour des grammaires et des analyseurs syntaxiques adaptés à la reconnaissance de la parole ont déjà été proposés. Leur but principal était de réduire l'indéterminisme induit par l'incertitude des terminaux reconnus (un terminal sera le plus souvent un mot du langage, ou un groupe de mots constituant une locution figée :*Je_voudrais_savoir_si, Pourriez_vous_me_dire*) , chaque choix de règle de réécriture au cours d'une analyse étant doublement aveugle :

- cette règle n'est pas la bonne car la grammaire du langage est indéterministe
- cette règle n'est pas la bonne car le terminal utilisé n'était pas correct

Le choix que nous avons retenu permet de modéliser simultanément la grammaire du langage à reconnaître et les traitements associés. Il a été défini dès 1981 par Jean-Marie Pierrel pour le système MYRTILLE II [PIERREL 81]. Ce sont

les Réseaux à Noeuds Procéduraux ou RNP, qui sont en réalité peu éloignés du concept des ATN de Woods [WOODS 70]. La différence essentielle réside dans le rôle des procédures associées à chaque noeud.

Pour Woods, celles-ci se chargeaient de construire la représentation syntaxique de la phrase, tandis que dans les RNP, elles servent à ordonner dynamiquement les hypothèses envisageables à partir de l'analyse partielle déjà effectuée et en prenant en compte d'une part le contexte déjà traité (test sur la représentation syntaxique de la partie d'énoncé déjà traitée) et d'autre part des informations phonétiques simples permettant de différencier les mots correspondant à des sorties différentes (tests sur le signal de parole).

Les RNP ont été développés pour faire des traitements syntaxiques adaptés à la compréhension de la parole et ils restent aujourd'hui l'outil permettant de résoudre au mieux les problèmes posés.

Chaque contrainte syntaxique locale sera traduite par un sous-réseau. L'entrée dans un tel sous-réseau se fera à la suite de l'activation d'une procédure d'entrée. Les déplacements sur les arcs correspondront soit au traitement d'un terminal du lexique (le mot correspondant sera soit détecté sur la plage de signal appropriée, soit supposé élidé), soit au traitement d'un non terminal (franchir cette transition consistera à faire un appel [parfois récursif] au sous-réseau correspondant).

Les traitements se feront dans les noeuds procéduraux et consisteront surtout à recueillir un maximum d'informations permettant d'ordonner les différentes sorties envisageables, voire d'en éliminer.

4.4 Informations pragmatiques

Ce sont toutes les informations liées à l'application et au sous-langage utilisé dans cette application.

Une des premières informations pragmatiques concernant une application est son vocabulaire. Nous avons déjà fait remarquer qu'un sous-lexique technique est propre à l'application, tandis que des mots outils apparaissent avec une fréquence relativement stable. L'analyse d'un corpus permettra de recueillir ce type d'informations. Le stockage de la fréquence a priori de chaque mot peut servir à initialiser des scores de plausibilité de présence –a priori– des mots, pour permettre un examen préférentiel des entités lexicales rencontrées le plus fréquemment. De plus, cette approche semble compatible avec les modèles des psycho-linguistes.

C'est aussi à ce niveau qu'interviennent des modèles de la tâche, de l'utilisateur, du dialogue : lorsqu'un locuteur dit : "C'est pour une carte d'identité ", l'inter-

prétation qui en sera faite dépend du contexte.

Les différentes réponses qu'une telle affirmation peut susciter seront très variables en fonction de la personne (ou plutôt de la fonction de la personne) à qui cet énoncé s'adresse.

- L'hôtesse de la mairie : "Adressez-vous guichet C en bas de l'escalier à droite"
- L'hôtesse d'un centre de renseignements administratifs : "Pour obtenir une carte d'identité, adressez-vous soit à la mairie, soit au commissariat, avec votre livret de famille ou un fiche d'état civil et de nationalité française".
- Dans un magasin : "Ah, c'est vous qui l'avez oubliée hier!"

Toutes ces personnes ont appliqué le scénario le plus classique pour compléter les informations dont elles disposent pour interpréter au mieux la demande. Comme un nombre minimal d'informations est donné, toutes celles qui manquent se voient affecter une valeur par défaut.

La mairie délivre les cartes d'identité, la personne qui se présente en ces termes veut en obtenir une, ou vient déjà la retirer. Il suffit donc de l'envoyer au bon guichet.

L'hôtesse du centre de renseignement a supposé que cette personne voulait obtenir une carte d'identité car ce scénario est plus fréquent qu'une perte. Elle a répondu implicitement à une question du type "Quelles sont les démarches à accomplir pour obtenir une carte d'identité?"

Si la personne désirait savoir ce qu'il fallait faire en cas de perte de ce même objet, elle aurait explicité d'avantage son énoncé —si elle partage les valeurs par défaut de l'hôtesse—. Sinon, cette personne devra lever l'ambiguïté par une contestation ultérieure.

Plus l'application est technique et restreinte et plus le nombre de valeurs par défaut augmente et plus ces valeurs sont partagées par un plus grand nombre de personnes : ainsi, "le prix pour Paris?" demandé au guichet d'une gare sera le plus souvent suivi d'une réponse, correspondant au prix d'un billet SNCF en deuxième classe, à plein tarif, d'un aller simple au départ de la gare où la question est posée, vers la gare de Paris correspondante. Par exemple à Nancy, vous aurez le prix Nancy–Paris-Est.

A chaque mot du lexique devra donc être associé un **prédicat** qui permettra son interprétation ultérieure. Ces prédicats peuvent être ceux de la grammaire de cas.

Exemples:

- Ech-obt(obtenir)
- mvt1(aller)
- mvt1(se rendre)

On peut également relier des mots du lexique à un concept plus général dont ils sont l'un des représentants :

Exemples :

- administration(hôtel-de-ville)
- administration(mairie)
- administration(préfecture)
- administration(commissariat)
- concept-mairie(mairie)
- concept-mairie(hôtel-de-ville)

Toutes ces informations seront nécessaires pour l'interprétation correcte de l'énoncé dans l'univers de la tâche, de façon à rendre le dialogue avec le système le plus naturel possible, c'est-à-dire le plus proche possible d'un dialogue avec un partenaire humain. Nous y reviendrons dans le chapitre suivant quand nous traiterons la partie "DIAL".

Chapitre 3

Systèmes en cours de développement

1 Introduction

En parallèle avec notre réflexion plus formelle sur les systèmes de dialogue oral homme-machine, nous développons sur les différentes machines du laboratoire, en particulier sur un Masscomp 5600 et sur des micro-ordinateurs, différents systèmes de dialogue, à but finalisé.

Les systèmes développés répondent à divers besoins : le premier, "DIAL", est entièrement du domaine de la recherche, et vise surtout à montrer la faisabilité d'un tel système de dialogue. Ce système est donc très complexe, et le premier travail de notre groupe de recherche a été de définir une architecture permettant d'intégrer toutes les connaissances pouvant aider à une bonne compréhension du dialogue, ou à une reconnaissance plus rapide d'un énoncé [CARBONELL 84a, CARBONELL 86]. Cette définition est le fruit d'un travail collectif, mené par l'ensemble des participants au projet.

Une deuxième famille de systèmes, d'ambition plus limitée, correspondent à des maquettes pré-industrielles, et situent mieux l'état de l'art du domaine. Leur but est de montrer ce que nous savons faire en 1988, ou plutôt ce qu'il est possible de faire à moyen terme (le moyen terme se situant de trois à cinq ans). Dans ces systèmes, aucun des modules n'est simulé, et les résultats correspondent à une réalité. Si les aspects temps réel et pourcentage de réussite restent à améliorer pour un usage industriel, les limites et les possibilités sont clairement montrées, et c'était justement le but poursuivi.

Au chapitre précédent , nous avons recensé l'ensemble des informations utilisables par un système de compréhension. Ces informations sont trop complexes pour être utilisées ou même gérées de façon globale. A chaque type d'informations

correspondra donc une structure propre, et des procédures de traitement propres. Notre système sera donc modulaire, les différents modules étant les plus indépendants possible, chaque module ayant une base de connaissance spécifique. Les modules devant s'échanger des informations devront définir un protocole de communication, et une représentation commune des informations à échanger. Cette approche a plusieurs avantages :

- La séparation d'un module et de la base de connaissances qu'il utilise facilite les modifications à apporter à une partie donnée, ainsi que la mise au point des programmes avec une base de connaissances incomplète.

- L'indépendance entre les différents modules permet de réaliser séparément chacun d'entre eux, et d'effectuer des tests en générant les informations devant transiter par les différentes interfaces d'un module.

- Les modules étant conçus de façon indépendante, et en l'absence de blocage sur une donnée manquante, le système global ainsi construit sera naturellement parallèle, chaque module pouvant très bien s'exécuter sur une machine distincte, et être écrit dans un langage propre. La seule contrainte est la possibilité de dialoguer avec les autres parties du système, chaque module devra donc avoir une interface de communication.

- Chaque module étant vu par tous les autres par son interface, on pourra très bien changer une partie ou la totalité des traitements internes sans avoir à modifier les autres parties du système.

2 Le système "DIAL"

2.1 Ses différentes composantes

Nous avons divisé notre système "DIAL" en cinq composantes [CARBONELL 85] : le décodage phonétique –APHON–, un module d'extraction d'indices prosodiques –PROSO–, un module de traitements lexicaux –LEX–, un module chargé des analyses syntaxico-sémantiques –SYNSEM–, et enfin un gestionnaire de dialogue –DIALOG–.

Nous allons détailler les interactions entre chacun de ces cinq modules, ainsi que les résultats produits par chacun d'entre eux.

2.2 Le module APHON

Ce module est chargé de tous les traitements, depuis l'acquisition du signal de parole émis par le locuteur jusqu'à la livraison du treillis de phonèmes correspondant, ainsi que d'autres informations pouvant être utiles aux autres processeurs.

Après l'acquisition du signal et les deux étapes de prétraitement et de traitement, plusieurs types d'informations sont disponibles :

- Début et fin de parole effectifs
- Durée vocalique moyenne (durée moyenne d'une voyelle dans l'énoncé courant)
- Segmentation du signal en classes de phonèmes et étiquetage de ces classes (une étiquette par segment)
- Treillis phonétique associé :

  À chaque segment produit par le prétraitement est associée une liste de phonèmes possibles. Chaque occurrence de phonème envisagé n'est valide que dans un contexte donné, celui qui a permis son "hypothétisation". Un même phonème pourra apparaître plusieurs fois sur la plage, mais avec des scores différents pour chacun des contextes envisagés. En effet, les valeurs des indices ou les indices eux-mêmes associés à un phonème sont dépendants du contexte. La véracité d'un même indice sera fortement influencée par le contexte, il pourra être caractéristique dans un cas et être assez peu signifiant dans un autre.

  Exemple d'interprétation d'une même mesure d'un indice calculable pour identifier un segment fricatif sourd :

  - limite_inf_friction = 3300 et contexte_labial alors "*s,0.6*"
  - limite_inf_friction = 3300 et contexte_non_labial alors "*s,0.2*"
  - limite_inf_friction = 3300 et contexte_labial alors "*ch,0.2*"
  - limite_inf_friction = 3300 et contexte_non_labial alors "*ch,0.5*"

- Chaque hypothèse sera donc un quadruplet (label, score, contexte_gauche, contexte_droit), où contexte_gauche et contexte_droit seront des liens de chaînage vers d'autres hypothèses, correspondant respectivement au segment précédent et au segment suivant. Si chaque segment en cours d'analyse connaît son contexte à gauche, car l'analyse a déjà été faite, tout contexte droit potentiel devra être envisagé. Si une hypothèse à ce stade ne se confirmait pas, on détruit alors le morceau d'arborescence construit en fonction

de cette supposition erronée. Le rejet d'une hypothèse conduit donc le programme d'étiquetage à parcourir ses chaînages arrières et à détruire le noeud courant si le facteur droit dont on vient était le dernier valide. On se contentera d'ôter ce facteur droit s'il avait d'autres frères.

C'est l'arbre ainsi élagué qui constituera le treillis phonétique passé au niveau lexical.

1. **Acquisition du signal d'un énoncé. Traitements acoustico-phonétiques et prosodiques**

   L'acquisition du signal est très liée au matériel dont nous disposons. Après une période sans acquisition de parole véritable pour tester nos programmes, nous disposons depuis décembre 1986 d'une machine spécialisée (un MASSCOMP 5600) en traitement de signal. En particulier, il comporte une carte permettant de faire des acquisitions et des restitutions de parole. Cette carte permet de digitaliser le signal issu d'un microphone standard, et inversement de la re-synthétiser. Le signal brut est codé sur douze bits, et seize convertisseurs peuvent travailler en parallèle. Le tableau correspondant à ce signal brut digitalisé est présent en mémoire centrale, et donc accessible par tous les niveaux de traitement.

   Jusqu'à présent, les contraintes techniques empêchaient la présence simultanée de ce signal avec tous les modules de traitement. Les architectures proposées ou réellement implantées étaient donc de façon naturelle uniquement ascendantes, chaque module du niveau bas vers le niveau haut prenait les résultats du précédent comme données, après la fin de l'exécution de celui-ci. Les passages des résultats d'un niveau à l'autre étaient le plus souvent simulés, à la fois à cause de limites techniques (les temps de calcul étaient trop importants au niveau du décodage pour fournir un ensemble de résultats suffisamment étoffé et en un temps raisonnable au niveau lexical par exemple), et aussi de limites qualitatives : il est inutile d'essayer de connecter un système évolué de reconnaissance et d'interprétation de phrases si les résultats fournis par le décodage n'atteignent pas environ 80 % de bons phonèmes.

   Comme tout retour arrière était impossible, le but était de ne pas omettre un résultat possible : ainsi, le décodage acoustico-phonétique préférait faire une sur-segmentation, et fournir pour chaque segment ainsi trouvé une liste de phonèmes suffisamment longue pour que la probabilité d'omission de la bonne étiquette soit la plus faible possible. A l'usage, il apparaît que cette stratégie n'était pas forcémment excellente : par exemple, quand la

segmentation était exacte (par exemple, si ce module a été remplacé par une segmentation manuelle quasi-exacte) l'étiquetage automatique effectué par aphodex fournissait à près de 90 % la bonne étiquette dans les deux premières proposées. La correction d'omission par l'ajout d'une troisième étiquette était marginale (de l'ordre de 1,5 % de gain global). Mais les grosses erreurs du décodeur actuel ne résident pas dans l'étiquetage. Celui-ci est peu affecté par une segmentation automatique, même si celle-ci est peu précise. De plus, un segment trouvé (par exemple i ou é) mais classifié de façon erronée (par exemple liquide) sera étiqueté invariablement "l"; ce type d'erreur représente une grande partie des erreurs faites par le module de détection des noyaux vocaliques, mais il est très facilement corrigé au niveau lexical :
Un décodage du type :

"/p/r/)/s/l/s/y/s/" est presque parfait pour identifier le mot "processus", sa transcription phonétique idéale étant "/p/r/)/s/ϵ/s/y/s/"

mais l'omission complète (courante car le segment est très court) est beaucoup plus grave :

cela donne :

"/p/r/)/s/y/s/"

voire

"/p/)/s/l/s/" et cela rend très difficile une reconnaissance ascendante de ce mot.

Toutefois un test de présence (demande de reconnaissance descendante à la suite d'une hypothèse d'un niveau supérieur) reste possible. L'idéal dans ce cas est de ré-examiner le signal pour remettre en cause une segmentation hative, avec comme critère la longueur exagérée de la fricative, ou la connaissance a priori de la très petite longueur de la voyelle à chercher (souvent 2 ou 3 prélèvements).

Jusqu'à un passé récent, la qualité d'acquisition était doublement limitée par les machines : les systèmes précédemment développés au CRIN étaient basés sur des acquisitions à 10 KHz ou 12 KHz au mieux. Cela permettait de travailler sur des bases de fréquences variant de 50 Hz à 5000 Hz, ce qui était facheux par exemple pour une locutrice ayant une fréquence fondamentale élevée : détecter des plosives au lieu de fricatives "s" dans "*ces serpents sifflent sur...* " devenait possible!

Les progrès techniques permettent aujourd'hui d'acquérir des prélèvements

à des fréquences très élevées (La carte dont nous disposons sur MASSCOMP permet d'acquérir à une fréquence maximale de 333 KHz!). La principale limite réside donc dans la taille des informations à traiter, et du temps nécessaire au traitement de l'information recueillie (il dépend de la taille de l'information bien sûr). Si 40 KHz est considéré comme parfait pour un mélomane, cela nous a semblé excessif. En effet, aucune information que nous utilisions jusqu'alors n'était altérée à 20 Khz. Après divers essais, et pour réduire un peu la taille des fichiers paroles que nous voulions garder, nous nous sommes fixés une fréquence standard d'échantillonnage de 16 KHz. Une valeur fixe était d'autre part nécessaire à l'ajustement des paramètres des modules de prétraitement, si bien que cette valeur n'a plus été remise en cause ensuite. Elle reste toutefois modifiable par l'utilisateur.

A la fin de la procédure d'acquisition, dans la version actuelle de notre système, la totalité de la plage d'acquisition est affichée de façon graphique pendant l'élocution.

Dès la fin du temps imparti au locuteur, le début et la fin absolue de parole sont donnés à l'utilisateur, ainsi que des messages de mise en garde si l'acquisition n'est pas parfaite :

- début d'acquisition tronquée : le locuteur a commencé à parler avant le début de l'acquisition effective.
- fin d'acquisition coupée : le locuteur n'avait pas fini de parler quand le temps maximum qui lui était imparti s'est écoulé
- des saturations ont été détectées dans le signal. Si leur nombre est trop important, l'analyse du signal sera impossible, ou plutôt risque de fournir des résultats amusants, sauf peut-être pour le concepteur du système.
- le signal a été perçu faiblement : aucune valeur n'atteint 25 % de son maximum : la dynamique risque d'être trop faible pour permettre une bonne détection des indices utilisés dans la suite des traitements

Dans tous les cas, un message de mise en garde est émis. Cependant, l'utilisateur reste seul juge de la qualité de sa production, et la suite des traitements peut se dérouler de façon automatique s'il le demande. Il est tout de même souvent préférable de refaire une acquisition dans un tel contexte.

Si tout est correct, l'enchaînement normal consiste à décoder le signal acquis. Tous les paramètres nécessaires à cette nouvelle phase sont fixés de façon

automatique, sauf avis contraire de l'utilisateur

2. **Prétraitement et segmentation**

   Toute cette partie de traitement de signal, qui aboutit à une segmentation et un étiquetage des segments sous forme de classe phonétique est la réimplémentation en C, après réajustement des paramètres, d'une partie du système APHODEX développé par Dominique Fohr au laboratoire [FOHR 86].

   La version initiale avait été écrite en langage Pascal sur un Motorola 68000. Comme cette machine ne possédait pas d'entrée vocale, tous les fichiers tests étaient des sauvegardes provenant d'un système plus ancien, développé sur Mitra. Ce système était le résultat d'une collaboration étroite et très fructeuse avec un phonéticien de l'institut de phonétique de Nancy II, François Lonchamp, expert en lecture de spectrogramme. Cet expert segmente et étiquette des spectrogrammes mutilocuteurs, avec des résultats nettement supérieurs à ceux que peut fournir un système automatique de décodage, sans se servir d'informations lexicales.

   Le premier travail de Dominique Fohr a donc consisté à recueillir l'expertise de cette personne, qui a bien voulu décoder en sa présence, et surtout en explicitant de son mieux tous ses raisonnements, cinquante phrases prononcées par cinq locuteurs différents. Chaque locuteur a ainsi prononcé dix phrases phonétiquement équilibrées de Combescure [COMBESCURE 81].

   Dès qu'il a eu suffisamment d'informations, Dominique Fohr a développé en parallèle un système expert, SYSTEXP, qui simulait le travail de l'expert, de façon à valider les règles retenues [CARBONELL 84b]. Au départ, la plupart des indices requis n'étaient pas calculés par le système lui-même, mais demandés de façon conversationnelle. Après validation, le calcul des indices valides est devenu effectif si leur extraction était possible de façon algorithmique.

   Ce système comportait trois parties :

   - **Trois modules de prétraitement.**

     Ceux-ci ont juste été traduits en langage C et ont vu leurs paramètres légèrement modifiés pour s'adapter à la nouvelle acquisition.

   - **Une base de connaissance**

     Elle contenait les connaissances expertes formalisées sous forme de règles de production du type :

(ensemble de prémisses) $\Longrightarrow$ (ensemble de conclusions valuées)

Chaque prémisse pouvait être soit une conclusion précédemment déduite par le système, soit un paramètre évaluable par des mesures sur le signal ou par une demande explicite à l'utilisateur.

- **Un moteur d'inférence**

  La plupart des règles sont contextuelles, pour modéliser les phénomènes de co-articulation. Comme l'analyse du signal s'effectue de gauche à droite, seul le contexte gauche est connu quand on examine un phonème. Des hypothèses sur la nature du contexte droit sont générées, et elles ne seront remises en cause qu'au moment de l'analyse effective des phonèmes suivants. Pour permettre cette génération spontanée d'hypothèses, puis leur confirmation et infirmation a posteriori, un chaînage mixte était nécessaire : le chaînage avant générait des hypothèses en fonction des observations déjà effectuées, un chaînage arrière mettait à jour en supprimant des hypothèses erronées (le contexte droit s'est avéré différent de celui attendu)

Les trois modules de prétraitement ont donc été réécrits en C dans le système actuel, mais restent très proche des versions de SYSTEXP. Ils réalisent une première segmentation en macro-classes. toutes les informations utilisées à ce niveau sont non contextuelles. En plus du calcul du pitch, trois algorithmes indépendants relèvent des indices permettant d'isoler :

- **les voyelles** (orales ou nasales)

  Nous en avons retenu quatorze pour notre application.

- **les fricatives** (sourdes ou voisées)

  Elles sont six en français : /f/, /s/, /ch /pour les sourdes et /v/, /z/, /gh/ pour les voisées.

- **les plosives**

  Là encore trois sont sourdes : /p/, /t/ et /k/, et trois voisées : /b/, /d/, /g/.

Les plages de signal restantes et suffisamment larges sont considérées comme des liquides ou des semi-voyelles (soit /r/, /l/, /m/, /n/, /nj/, /w/, /ui/, /j/)

A l'issue de ce traitement, la totalité du signal est divisé en segments, et à chaque segment est associé une étiquette de classe.

Six étiquettes sont possibles :

- PS pour une consonne plosive non voisée
- PV pour une consonne plosive voisée
- FS pour une consonne fricative non voisée
- FV pour une consonne fricative voisée
- VV pour une voyelle
- IV pour un phonème n'entrant pas dans les classes précédentes

Remarque :

/r/, /l/, /m/, /n/ étant tous les quatre dans la catégorie autres phonèmes, et la partie division d'un segment en plusieurs sous-segments du système expert n'étant pas à ce jour ré-implantée, deux de ces phonèmes prononcés de façon contigu seront toujours regroupés en un seul segment étiqueté IV. Ils recevront souvent la double étiquette des deux phonèmes présents, mais seront le plus souvent considérés comme un phonème unique par les niveaux supérieurs.

3. **Etiquetage des segments fournis par le prétraitement**

- **plosives sourdes :**

  Plusieurs critères pertinents de discrimination à l'intérieur de cette classe de phonèmes avaient été validés par le système expert. Ces critères sont le résultat d'observations du burst ou barre explosion qui met fin à la partie occlusive du phonème, et des transitions observées sur les formants des voyelles adjacentes.

  Le suivi de formants reste difficile à faire de façon entièrement automatique (détection de formants parasites, saut de formants), le critère principal de discrimination porte sur la nature de la barre d'explosion. Deux cas sont à envisager :

  1) Le burst n'est pas détecté :

  - Si l'on est en début d'énoncé : Il s'agit plutôt d'une pause (#) ou peut-être d'un P.
  - Si l'on est en fin d'énoncé : Il s'agit sûrement d'une pause.
  - Si l'on est en milieu d'énoncé : c'est sans doute un P, ou une pause, ou peut-être K. Ces trois possibilités étant bien sûr ordonnée selon des plausibilités décroissantes.

  2) Un burst est détecté et analysé :

- Si ce burst est fricatif (bruit assez net, en haute fréquence) : C'est un /t/, sans doute suivi d'une voyelle d'avant, de type i ou é
- Si ce burst est double, en basse fréquence, c'est sans doute /k/, peut-être /p/.
- Si ce burst a une fréquence moyenne élevée (centre de gravité de l'énergie supérieur au troisième formant de la voyelle suivante par exemple) , on classera dans l'ordre /t/, /p/ et /k/.
- Si ce burst se situe en basse fréquence, on classera dans l'ordre /k/ et /p/.
- Si ce burst est équi-réparti (bruit faible, mais présent dans une large bande de fréquences) : On classera dans l'ordre /p/, /t/, et /k/.

- **plosives voisées**

  Des critères très proches de ceux retenus pour leurs homologues non-voisées sont utilisés. Toutefois, on s'interdira toujours de considérer un segment ayant reçu le trait voisé comme un silence.

  Une plosive imparfaite (présence de bruit diffus pendant la partie occlusive), et dont le burst n'est pas visible, recevra la double étiquette /b/, /v/.

  Pour différencier de façon plus fine /b/ et /d/ avec une voyelle d'avant comme contexte droit (/i/ ou /é/ par exemple) , des informations utilisées dans le système expert ne pas encore mises en oeuvre dans la version actuelle. Un suivi des deuxième et troisième formants fournit un résultat fiable s'il fonctionne correctement, mais la mise au point reste difficile.

- **fricatives sourdes ou voisées**

  En plus du voisement qui différencie globalement /f/, /s/ et /ch/ de /v/, /z/ et /gh/ (je), deux indices principaux servent à l'étiquetage :

  - L'intensité du bruit de friction :
    Un bruit peu intense et présent dans toutes les bandes de fréquences est souvent caractéristique de /f/ et /v/. Ces deux phonèmes seront mis en avant si un tel critère est détecté.
  - La limite inférieure du bruit de friction :
    La fréquence à laquelle le bruit de friction s'arrête est fondamentale pour différencier /s/ et /z/ d'une part, et /ch/, /gh/ d'autre part. Cette limite est contextuelle, elle est abaissée dans un contexte labial, remontée si la voyelle suivante est /i/ ou /é/.

- **Voyelles**

  L'étiquetage des voyelles nécessite plus de calculs. Le système en connaît un grand nombre : il distingue les quatorze étiquettes suivantes : /a/, /i/, /é/, /ϵ/, /y/, /ou/, /∂/, /oe/, /eu/, /o/, /)/, /ã/, /õ/, /˜ϵ/. Les phonèmes /õe/ et /˜ϵ/ seront considérés comme un seul et même phonème /˜ϵ/

  Le système fournira deux ou trois étiquettes pour chaque segment supposé être une voyelle. L'absence d'étiquette ou un "?" en troisième position remettra en cause la bonne segmentation.

  L'information qui sert à l'étiquetage est fournie par des transformées de Fourrier : chaque voyelle est décrite dans le système par trois valeurs correspondant aux trois premiers extrema locaux de l'énergie dans la transformée rapide. Ces trois valeurs sont appelées "formants" de la voyelle considérée et ces trois fréquences sont caractéristiques d'une voyelle donnée. Les valeurs retenues sont des valeurs moyennes, pour un locuteur français, de sexe masculin. Elles varient bien sûr d'une réalisation à l'autre, même pour un locuteur donné. Nous ne faisons pas encore d'adaptation automatique de ces valeurs, mais celle-ci est à l'étude. Nous ne faisons pas non plus d'apprentissage, notre but étant d'obtenir un système multi-locuteur sans apprentissage, celui-ci étant trop fastidieux s'il veut reposer sur un ensemble de données de taille suffisante. De toutes manières, il semble hors de question d'imposer un apprentissage long et fastidieux à un utilisateur occasionnel ou à un utilisateur de passage. L'avenir réside donc dans une adaptation automatique au locuteur, en cours de dialogue, ou peut-être même en cours d'énoncé si l'on identifie une voyelle de façon formelle.

  L'adaptation se fera en corrigeant les valeurs moyennes en fonction des valeurs observées réellement sur le signal du locuteur.

  Ces valeurs vont donc servir de référence pour l'étiquetage. Pour chaque prélèvement de la voyelle en cours d'analyse, on calcule une transformée de Fourrier. On relève ensuite les trois premiers formants, et on calcule une distance euclidienne entre ces trois valeurs et les moyennes que nous connaissons. Les trois voyelles les moins éloignées reçoivent un renforcement de leur score. Après avoir traité l'ensemble de ces prélèvements, les voyelles sont ainsi classées par scores décroissants. Si un consensus se dégage, deux étiquettes seulement sont données : nous avons en effet observé que dans ce cas, la co-articulation était faible, et si le locuteur est proche du locuteur moyen, l'étiquette est l'une des

deux premières. Sinon trois étiquettes seront attribuées. C'est cette version intermédiaire et minimale qui a servi aux tests du module LEX.

4. Version actuelle

La partie experte du décodage et correspond à SYSTEXP est en phase de ré-implémentation et de test dans notre laboratoire actuellement. Elle est elle aussi développée sur MASSCOMP et sera incorporable dans le système dès qu'elle sera validée. Dans un but d'uniformisation nationale, les tests sont effectués sur le corpus "**La bise et le soleil**" du GRECO communication parlée dont le texte figure en annexe. Dans cette nouvelle version, l'étiquetage se fait de façon contextuelle, à l'aide des règles recueillies auprès de l'expert phonéticien. Celles-ci permettent de prendre en compte les déformations de certains indices quantitatifs (par exemple hauteur de formants, seuils ou limite inférieure du bruit d'une fricative...) liés à la nature des phonèmes voisins, à droite et à gauche.

Si quelques problèmes subsistent, par exemple ceux liés à des phénomènes de co-articulation trop lointaine pour être facilement détectable, les résultats de l'étiquetage sont en net progrès.

Exemple de co-articulation éloignée :

Les deux mots "structure" et "strictement" ont des réalisations du /s/ initial assez différentes : le premier est coarticulé avec une labiale /y/, car les phonèmes /t/ et /r/ sont neutres vis à vis du placement des lèvres. Celles-ci seront donc positionnées dès que possible, donc dès le début du mot. Pour "strictement", le même phénomène se reproduit, et le /s/ initial est coarticulé avec une voyelle d'avant, /i/.

Tous les autres cas moins difficiles sont prévus par le système, et chaque hypothèse faite à une étape donnée du décodage conduit au développement d'un traitement particulier, générant une nouvelle branche dans le treillis fourni en résultat. De plus, lorsqu'une hypothèse est retenue, un ajustage de la segmentation après reconnaissance de la classe des phonèmes à séparer est mis en oeuvre. Il est en effet plus facile de segmenter correctement une séquence fricative–voyelle quand on sait effectivement que ces deux types de phonèmes existent. Cela devrait améliorer d'autant la qualité d'étiquetage de chacun des segments ainsi affinés.

2.3 Le module PROSO

Ce module travaille sur une double donnée : d'une part la valeur du fondamental F_0, d'autre part sur la liste des plages correspondant à un noyau vocalique. Ses résultats seront aussi de deux types :

- Le fondamental F_0 sert à déterminer le mode de l'énoncé `interrogatif` ou `affirmatif`
- L'évolution de la valeur des extrema locaux du fondamental F_0 à l'intérieur des voyelles permet de marquer un certain nombre d'entre elles comme étant les dernières d'un mot ou d'un groupe syntaxique. Le résultat fourni est donc une liste de marques temporelles, toutes situées dans une voyelle.

1. **Traitements prosodiques** Jusqu'à présent, si beaucoup d'auteurs mentionnent la prosodie comme une information nécessaire à la compréhension d'un énoncé continu, y compris pour un auditeur humain, peu de systèmes de reconnaissance ont réellement utilisé les informations prosodiques. Ces informations peuvent être de trois types :

   - La variation de la fréquence fondamentale
     Celle-ci peut permettre d'une part de distinguer un énoncé affirmatif d'un énoncé interrogatif, et d'autre part de reconnaître ou de valider des frontières entre les différents groupes syntaxiques de cet énoncé.
   - L'intensité
   - Le rythme

   Ces deux derniers paramètres sont utiles aussi bien au locuteur qu'à l'auditeur pour mettre en relief une partie de l'énoncé, par exemple après une incompréhension : le locuteur va prononcer la partie sur laquelle une confusion s'était produite de façon différente, en détachant ses mots. Un bon système de dialogue devrait utiliser ce type de connaissances.

   Depuis l'année passée, un chercheur du CRIN s'attache à extraire ce type d'informations sur le signal acoustique. Ses premiers résultats concerne l'étude de la variation de la fréquence fondamentale F_0, dans le but d'extraire des frontières de mots ou de groupes syntaxiques [BONIN 87,CARBONELL 88].

   Ce module de traitement va donc produire des informations ponctuelles de façon presque autonome. Ses seuls paramètres en entrée et dépendants d'un autre niveau concernent la segmentation en macro-classes de phonèmes : il aura besoin des frontières des noyaux vocaliques.

La place assez indépendante de ce module et le rôle un peu auxiliaire qu'il joue au départ ont fait que ce type de traitement avait été laissé un peu de coté au début du projet. Des problèmes d'évaluation (nécessité de travailler sur des informations exactes) ont eux aussi contribué à un développement indépendant bien que parallèle de cette partie. Ainsi, bien que ce module soit opérationnel, il n'est pas encore intégré à ce jour au système global.

D'autres équipes utilisent la prosodie pour rechercher des limites de constituants syntaxiques ou déterminer la structure syntaxique d'un texte lu [PERENNOU 82] en étudiant la position et la longueur des pauses. Un système mis au point à Marseille par Méloni et Guizol [MELONI 82] s'appuie sur la variation de la fréquence fondamentale F_0, sur la durée et l'intensité des noyaux syllabiques, et sur les pauses pour identifier le schéma prosodique global d'une phrase, puis s'en sert pour l'étiquetage des noyaux syllabiques. Tous ces paramètres sont ajustés en fonction du contexte spécifique de chaque énoncé.

Enfin, Jacqueline Vaissière [VAISSIERE 82] considère que les contours de la fréquence fondamentale et la durée relative des segments détectés devraient permettre de corriger certaines erreurs de segmentation.

2. **Calcul du fondamental**

L'estimation d'une valeur suffisamment précise du fondamental F_0 et la décision voisement/non-voisement d'une portion de signal reste délicat. En effet, la périodicité de F_0 n'est pas constante. Il faut donc choisir une fenêtre de calcul suffisamment petite pour que cette fréquence puisse y être considérée comme stable. D'autre part, pour que cette période soit mesurable, la fenêtre doit en contenir au moins une entière. Ces deux conditions sont mutuellement exclusives aux points de grande variation, mais qui sont ceux qui nous intéressent : ce sont les passages d'un segment voisé à un segment non-voisé et vice versa. De plus la présence de bruit de fond sur le signal acquis altère un peu les résultats.

L'algorithme qu'a implémenté Jean-Jacques Bonin est basé sur des études de Rabiner et Scafer [RABINER 78] aprés avoir effectué un center clipping sur le signal de parole, pour ne retenir que les informations pertinentes pour la fonction d'autocorrélation.

C'est une transformation non linéaire qui élimine toute la partie de signal d'amplitude inférieure à un seuil donné.

Une transformation encore plus radicale conduit à une grande simplification des calculs d'autocorrélation :

Si x(n) est le signal brut, on effectue la transformation suivante :

C[x(n)] = + 1 si x(n)> C_l

C[x(n)] = - 1 si x(n)< $-C_l$

c[x(n)] = 0 sinon

où C_l est le niveau de clipping, en général un pourcentage du maximum d'amplitude sur le segment que l'on est en train de traiter. Ce pourcentage a été fixé à 68 %, valeur recommandée par Rabiner et Schafer [RABINER 78].

Une fonction d'autocorrélation fournit deux pics maximum, et une décision voisement/ non-voisement est prise. Pour les parties voisées, une correction a posteriori est parfois nécessaire pour palier par exemple à des doublements de fréquence. Ce programme de correction utilise des contraintes de continuité sur la valeur du pitch, en interdisant de trop brusques variations de celui-ci sur une portion de signal donnée.

A partir de la courbe du fondamental ainsi obtenu, Jean-Jacques Bonin a exploité une remarque concernant les noyaux vocaliques : *la plupart des marqueurs prosodiques (Frontières de mots) se trouvent aux maxima d'une courbe représentant les fréquences maximales des noyaux vocaliques.* Son algorithme va donc détecter un certain nombre de noyaux vocaliques et les marquer comme étant la dernière voyelle définissant la frontière d'un mot.

Quand un mot est trouvé par niveau lexical, si la dernière voyelle qui le compose est marquée de cette façon, et qu'aucune autre ne l'est, on devra renforcer sa plausibilité. Inversement, une voyelle marquée apparaissant au milieu d'un mot diminuera a priori son score.

L'algorithme de marquage a été testé sur trois corpus différents, acquis dans des conditions différentes, et constitués par des énoncés émanants de quarante trois personnes.

La seule restriction pour un usage automatique concerne la segmentation : Cet algorithme a utilisé une segmentation manuelle des noyaux vocaliques, tous ses autres paramètres étant calculés de façon normale.

Les résultats sont plus qu'encourageants : Pour les cinquante phrases du premier corpus, 161 noyaux ont été marqués, dont 153 de façon correcte. Cela représente une information valide à environ 95 %. Dans le second test (corpus "*La bise et le soleil se disputaient...*") prononcé par neuf locuteurs

et locutrices, 457 noyaux ont été marqués, dont 425 corrects. Un locuteur réalise même un presque sans fautes : 48 marques correctes sur 49 !

Le taux de validité est donc du même ordre, mais ce corpus répétitif (9 réalisations différentes d'un même énoncé), a mis en évidence un autre phénomène intéressant : les *erreurs* sont répétitives : certains mots, longs, sont réfractaires au bon marquage :

s'AvançAIt reçoit un double marquage sur sa première et sa dernière syllabe à 6 reprises (sur 9 occurrences !), **vOYagEUr** fait la même chose quatre fois, et les mots en **re-** ou **ré-** sont abonnés au marquage sur leur première syllabe :

- 4 fois pour rEconnAÎtre
- 2 fois pour rÉchauffÉ
- rEgardÉ, rEnoncÉ provoquent une seule erreur chacun

Cette petite liste contient près de 50 % des erreurs.

De plus nombreux tests seraient nécessaires pour établir une règle générale, mais il semble que les mots qui risquent de poser des problèmes soient prévisibles. Une prévision exhaustive du double marquage de ces mots permettrait d'une part d'éliminer une bonne partie des *erreurs*, et d'autre part de posséder une information supplémentaire d'identification de ces mots si l'*erreur* (parfois plus fréquente que le non-marquage) s'est produite.

Il reste à tester ce système en condition de reconnaissance (c'est-à-dire avec une segmentation entièrement automatique), mais comme environ 90 % des noyaux vocaliques sont correctement détectés, ses performances devraient rester à un niveau plus qu'acceptable.

Environ le tiers des mots, par exemple trois par phrases pour le corpus de Combescure, seront ainsi marqués. La modification des scores de mots que ces renseignements induisent devrait accélérer de façon notable la convergence des analyseurs syntaxiques, la plupart des mots ainsi marqués étant souvent situés en fin de groupe. Cette information peut donc servir doublement dans un système de compréhension de dialogue oral.

2.4 Le module SYNTSÉM

Les données prises en compte par ce module viendront, là encore, d'horizons multiples :

- Elles peuvent provenir du niveau lexical : ce seront des mots reconnus de façon spontanée par la partie *hypothétisation*. Ces mots vont servir de noyau pour établir un ilôt de confiance permettant de débuter une analyse.

- Des hypothèses peuvent aussi provenir du gestionnaire de dialogue : Si le dernier message généré par le système à destination de l'usager était une demande de confirmation, ou une question fermée, celui peut émettre des hypothèses très fortes, à la fois sur la structure de la réponse attendue, mais aussi sur le type de vocabulaire le plus plausible. En effet, même une personne ayant beaucoup d'imagination ne répondra, (de façon spontanée, si elle se trouve réellement dans un situation de dialogue finalisé), à une question du type :" *Vous êtes bien majeur?*" que de très peu de manières :

  Exemples :

  - R1 : Oui.
  - R2 : Bien sûr.
  - R3 : Pas encore.
  - R4 : J'ai 19 ans.
  - R5 : Dans 3 mois.

  Il est donc possible après une telle question d'émettre des hypothèses, à la fois

  - structurelles : elles correspondront à une sous-grammaire, celle du sous-langage utilisé pour répondre à une question fermée de ce type :
  - lexicales : les mots pouvant apparaître dans une telle réponse appartiennent à un sous-lexique (ou à l'union de deux sous-lexiques) très restreint. (sous-lexique de confirmation, ou d'apport d'information - temporelle ou de mesure-). Il suffira de sélectionner les mots du lexique qui possèdent les traits sémantiques traduisant ces propriétés.

  Au niveau des analyseurs syntaxico-sémantiques, chacune de ces hypothèses, provenant du lexique ou du gestionnaire de dialogue, donnera naissance à ce que nous appelons une théorie, correspondant à une analyse d'un fragment d'énoncé. On distinguera

  - des théories ascendantes, correspondant à un ou plusieurs mots, le plus souvent contigus, fournis par le niveau lexical. Leur regroupement en une théorie sera à la fois syntaxique et sémantique, les deux analyses s'effectuant de façon parallèle.

- et des théories descendantes, représentant une traduction des hypothèses émises par le gestionnaire de dialogue.

La progression dans l'analyse d'un énoncé consistera à créer de nouvelles théories vides, puis à les faire grossir pour atteindre une couverture la plus complète possible du signal en entrée. Les diverses possibilités du système correspondent aux fonctions suivantes :

- Création d'une hypothèse-mot : si un groupe syntaxique en cours de construction est encore incomplet, on essaie de l'étendre à gauche ou à droite par des hypothèses lexicales Celles-ci devront être validées par le niveau lexical avant la poursuite de l'analyse dans cette voie. Les mots candidats seront d'abord préssentis en s'appuyant sur des connaissances syntaxiques puis, éventuellement, sémantiques.

- Génération d'une théorie descendante :

  - En exploitant une hypothèse provenant de module de gestion de dialogue.
  - Par transformation d'une théorie descendante : Une théorie descendante ne deviendra active qu'après la validation d'au moins un mot par le niveau lexical. La validation entraine la génération d'une nouvelle théorie descendante.
  - Par fusion de deux théories, l'une descendante et l'autre ascendante : Si une analyse ascendante et une analyse descendante se rejoignent, on les unifie et la théorie résultante est de type descendante.

- Génération d'une théorie ascendante :

  - Prise en compte d'un mot reconnu : on crée une nouvelle théorie ascendante réduite à ce mot.
  - Union de deux ou plusieurs théories ascendantes : Si un non terminal syntaxique correspond à plusieurs théories ascendantes, ou/et si un lien sémantique existe entre plusieurs théories ascendantes, celles-ci sont regroupées en une nouvelle théorie ascendante.

2.5 Le module DIAL

Ce module est développé par Azim Roussanaly [ROUSSANALY 88]. Son rôle principal est de dialoguer avec l'utilisateur du système, sa vraie finalité étant de

répondre aux questions que celui pose si l'information demandée est stockée dans la base de connaissances du système. Toutefois, cette composante jouera un deuxième rôle dans notre système en émettant des hypothèses structurelles ou lexicales. Les premières donneront les différentes formes d'énoncé les plus probables, en fonction de la phase de dialogue et de l'historique des échanges passés du dialogue en cours.

Plusieurs fonctions vont coopérer pour permettre la génération d'une réponse intelligente :

- La première étape consiste à interpréter la structure syntaxico-sémantique fournie par les analyseurs, qui représente le dernier énoncé émis par l'utilisateur : deux modules complémentaires vont réaliser ce travail :

  - L'interpréteur va utiliser des informations concernant l'univers de l'application et une base de règles d'interprétation pour traiter toutes les informations fournies par les autres modules.

  - Le raisonneur va résoudre les anaphores et détecter les incohérences, à l'aide de l'historique des échanges. Il sera aussi capable de faire des déductions simples, en se servant des informations sur l'univers de la tâche.

- C'est au module de gestion de dialogue qu'appartient la décision finale concernant la réponse à apporter à une requête de l'usager. Il prendra en compte non seulement les résultats du raisonneur obtenus à partir des énoncés relatifs à la tâche, mais il prendra aussi en considération l'ensemble des énoncés portant sur le dialogue lui-même ou sur la gestion du canal de communication. Quand plusieurs solutions sont en concurence, c'est ce module qui fera le choix final parmi les différentes éventualités.

  Il assumera une deuxième fonction très importante dans un système de reconnaissance, qui est la génération d'hypothèses : en fonction du contexte de dialogue (par exemple le système vient de poser une question à l'usager pour obtenir un complémenent d'information), des hypothèses à la fois lexicales et structurelles seront transmises aux processeurs des niveaux inférieurs pour initialiser ceux d'entre eux qui travaillent de façon descendante.

  Ce module s'appuie sur des connaissances dynamiques, représentant l'historique du dialogue en cours. Cet historique contient essentiellement une représentation chronologique des interventions passées et de l'état de la tâche.

2.6 Le module LEX

Si la définition de l'ensemble du système a été faite de façon collégiale par l'ensemble de notre groupe de travail, le développement de ce module constitue la partie essentielle de mon travail. Il sera décrit de façon plus appronfondie dans le chapitre suivant, qui lui est entièrement consacré.

2.7 Architecture du système DIAL

Les différents modules que nous venons de présenter et les bases de connaissances qui leur sont associées seront implémentés de la façon la plus indépendante possible, la modularité dans la conception du système sera donc reconduite dans sa réalisation. Cela présente plusieurs avantages :

- Ce découpage permet de mener en parallèle la réalisation de chacun de ces modules.
- Une fois les interfaces et les flux d'informations précisés, un module pourra être réutilisé dans d'autres applications.
- La paramétrisation du système ainsi obtenu permet de changer la conception interne de l'un des modules sans avoir à remettre en cause ses autres constituants. Ce système sera donc très flexible, et la mise en oeuvre d'une nouvelle application de même type que l'application développée en est grandement facilitée.
- Chaque module est vu comme un producteur–consommateur d'hypothèses. Si APHON n'est à ce jour qu'un producteur, il est prévu dans un avenir proche de lui demander des confirmations de présence de phonèmes, pour permettre une validation plus fine de certains. Nous avons déjà fait remarqué qu'une mauvaise segmentation était catastrophique pour la suite des traitements :

  `exemple : six ==> /s/i/s/ devient /s/ si le noyau vocalique est omis.`

  Prendre en compte une telle erreur au niveau lexical revient à reconnaître le mot "six" sur tous les segments fricatifs sourds ou presque. Par contre, s'il est possible d'émettre une hypothèse /i/ vers APHON, que celui-ci pourra valider ou exclure, le score fourni pour le mot "six" sera beaucoup plus significatif.

Nous permettrons donc ce type d'échanges dans notre architecture. Le schéma général est forni par la figure 3.1.

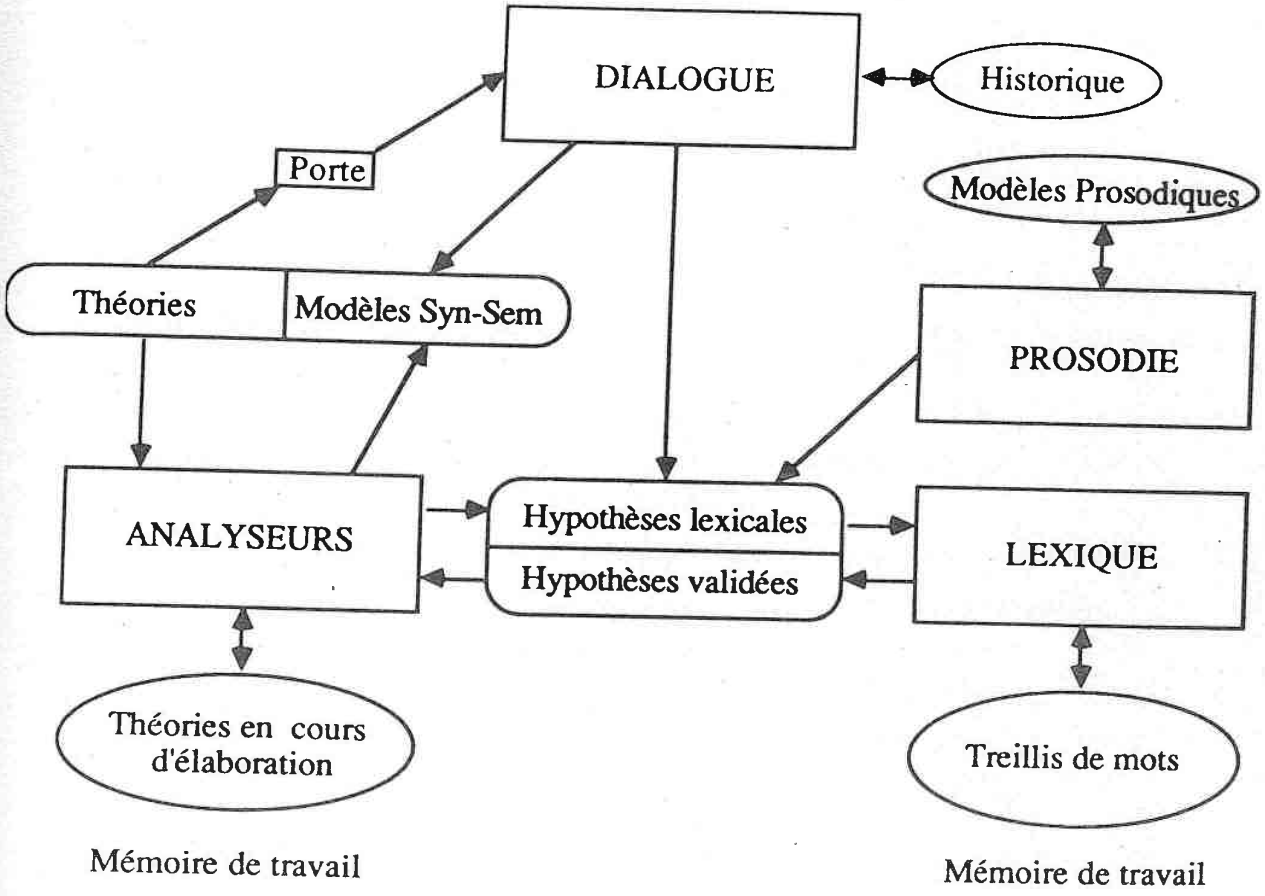


Figure 3.1: Schéma général de l'architecture de DIALOG

2.8 Flux d'informations entre les modules

Dans ce schéma général, nous n'avons pas mentionné dans un but de clarification les connaissances statiques propres à chaque module. Chaque flux

d'information est symbolisé par une flèche reliant les deux modules concernés.

Les informations qui vont ainsi être échangées sont de quatre types :

1. Informations concernant l'état du système.

   Exemples :

   - processeur d'acquisition non disponible,
   - lexique à utiliser,
   - processus prêt, en attente d'hypothèses à consommer,
   - ...

2. Hypothèses émises par un niveau supérieur.

   Ces informations n'ont qu'un caractère prédictif. Elles restent bien sûr à valider par un niveau inférieur, et elles ne conduisent le plus souvent qu'à modifier l'ordre de vérification de l'ensemble de toutes les hypothèses devant être envisagées.

   Toutefois, certaines sont des questions fermées, et demandent le renvoi d'un score représentant leur plausibilité :

   - Le mot "*demande*" est-il présent dans le premier quart de l'énoncé?

   Ces hypothèses peuvent être de quatre types dans notre système :

   (a) Des prédictions de type syntaxico-sémantiques.
   Ces prédictions font le plus souvent suite à une question posée à l'utilisateur, ou s'appuient sur la structure supposée du premier énoncé de l'usager. Elles seront utilisées par le module SYN–SEM pour initialiser son processus d'analyse descendante

   (b) Des prédictions lexicales en provenance de DIAL.
   Ce sont des listes de mots signifiants, dont la probalité d'apparition dans l'énoncé courant est importante, vu l'historique du dialogue.

   (c) Des prédictions lexicales en provenance de SYN–SEM.
   Ce sont des mots ou des sous-lexiques valides pour compléter une analyse descendante et dont la présence reste à vérifier sur le signal.

   (d) Des listes de phonèmes correspond à une représentation phonétique d'un mot, et dont la plausibilité reste à vérifier sur le signal. Cette possibilité est liée à la réimplémentation (en cours) de la partie experte SYSTEXP du décodage acoustico-phonétique.

3. Des informations ponctuelles concernant l'énoncé courant.

   Intuitivement, elles permettent la transmission d'informations d'un processus à l'autre, et remplace la notion de "variable globale" disparue à

cause du choix d'implémentation modulaire du système. Ces informations sont très diverses.

(a) La durée vocalique moyenne.
Celle-ci sert de façon interne à APHON pour ses propres traitements, mais elle est aussi nécessaire à PROSO, et à LEX pour calculer une longueur approximative d'un mot sur le signal, connaissant sa transcription exacte.

(b) La longueur totale de l'énoncé.
Celle-ci sert à tous les niveaux de traitement.

(c) Les marques prosodiques.
Celles-ci correspondent à des frontières de mots et de syntagmes. Elles servent à la fois à LEX pour modifier les scores des mots selon leur adéquation avec ces contraintes, et à SYN–SEM pour renforcer ou atténuer les hypothèses syntaxiques selon leur correspondance avec le découpage ainsi préconisé.

4. Les représentations de l'énoncé.
L'énoncé subit un grand nombre de transformations successives depuis le signal analogique issu du micro jusqu'à sa représentation interprétative nécessaire à DIAL : numérisation, segmentation, transformation en treillis phonétique, transformation en treillis lexical, mise sous forme d'un ensemble de structure syntaxico-sémantiques.
Sauf les deux premières, ces représentations sont obtenues après un va-et-vient hypothétisation–vérification entre deux ou plusieurs processus.

2.9 Contrôle et stratégie

La grande supériorité de l'humain sur la machine réside dans sa vision globale des problèmes à résoudre. Une stratégie souhaitable consiste donc à partager au mieux les connaissances entre les différents processeurs qui composent notre système. Jusqu'à présent, les différents systèmes développés admettaient une hiérarchie prédéfinie entre les niveaux infra-lexicaux —ou niveaux bas— et les niveaux supra-lexicaux — ou niveaux hauts —. Le sens de cette hiérarchie dépendait de la stratégie adoptée, par le concepteur : stratégie ascendante ou stratégie descendante. Ces deux approches présentent les mêmes inconvénients : les premiers processus mis en oeuvre sont privilégiés, car ils sélectionnent une sous-partie du lexique, et éliminent en fait de façon brutale l'autre partie du lexique, et surtout une remise en cause de ces décisions parfois prématurées est très difficile. Dans notre système, l'ensemble du lexique est valide à tout moment. Toutefois, seuls

les mots dont le score est suffisant seront examinés à un moment donné. Un apport d'information, quelqu'en soit l'origine, va modifier les scores des mots du sous-lexique concerné. La sélection de ce sous-lexique particulier se fait à l'aide d'une structuration prédéfinie, sous forme d'arborescences multiples correspondant chacune à une organisation particulière. syntaxique, sémantique, phonétique. A chaque noeud interne est associé le sous-lexique encore compatible avec le chemin parcouru pour parvenir à cet endroit. Toutes les contraintes utiles et ne correspondant pas à un chemin élémentaire pourront recalculer le sous-lexique adéquat, à l'aide d'opérations ensemblistes sur les sous-lexiques sélectionnés par chacune des sous-contraintes élémentaires qui les composent. [ROMARY 88]

Exemple : pour atteindre tous les mots du lexique autre que des verbes qui peuvent exprimer une action contrôlée ou qui acceptent un bénéficiare, on utilisera un expression du type :

```
(inter(^verbe)(union( ctr+ ben+)))
```

Ce type de commande pourra être utilisé pour généraliser à tous les mots du lexique la prise en compte de la présence probable d'un mot ayant les traits ctr+ ben+ , alors que le traitement a déjà été fait pour les verbes.

Une fonction spéciale niveau(x) pourra à un moment donné sélectionner tous les mots dont le score dépasse x.

Une stratégie standard consistera à initialiser le processus par une recherche ascendante de mots, de APHON vers LEX d'une part, et une analyse descendante des hypothèses de DIAL vers SYN–SEM et LEX d'autre part. Ces deux traitements se font de façon parallèle et indépendante. Le module SYN-SEM prend ensuite le contrôle en utilisant de façon ascendante des mots validés par LEX, et en générant de nouvelles hypothèses descendantes après les avoir traitées. Ce va-et-vient va se poursuivre jusqu'à ce qu'une ou plusieurs structures complètes soient construites, ou que ce processus soit bloqué faute d'informations nouvelles. DIAL sera alors réveillé par la transmission des résultats.

3 Le système Partner

3.1 Présentation

Le système Partner [MORIN 87] est un système de dialogue oral finalisé, utilisant un langage artificiel à consonnante naturelle. Les utilisateurs potentiels devront connaître l'application mise en oeuvre et la machine supportant cette application, et se montrer coopératifs. Ces hypothèses étant posées, le dialogue devra

être le plus naturel possible, et devra prendre en compte des énoncés elliptiques, anaphoriques ou incomplets. Il devra aussi être capable de faire des hypothèses raisonnables, à l'aide de scripts ou en affectant des valeurs par défaut à certain objects. Enfin, ce système étant développé en coopération avec l'université de Münich, il devra pouvoir fonctionner à la fois en langue française et en langue allemande. Son architecture sera bien sûr modulaire, trois modules étant responsables respectivement de la gestion de la tâche, de la gestion du dialogue, et de la reconnaissance de phrases. Ce dernier module partage toute sa partie infra-lexicale avec le système DIAL que nous venons de décrire. Il utilise également certaines fonctionnalités de LEX, en particulier des procédures de véfications de mots, et des demandes d'informations phonétiques correspondant à un mot donné.

Un des objectifs de Partner est de développer un module de dialogue indépendamment de l'application choisie, ce type de dialogue étant relativement stable pour un ensemble assez large de systèmes de commande dont le vocabulaire ne dépasse guère la centaine de mots. Des mécanismes assez généraux ont donc été mis en évidence, pour permettre un changement d'application rapide, sans avoir à modifier ce module.

L'autre objectif est de valider rapidement les idées dégagées dans l'étude de DIAL, et d'obtenir à court terme un système opérationnel intégrant l'ensemble des processus : décodage, reconnaissance dce mots, compréhension de phrases, gestion du dialogue et de la tâche et réponse engendrée par le système.

3.2 Architecture

Comme dans le système DIAL que nous venons de détailler ci-dessus, l'architecture retenue est hiérarchisée et modulaire. Chacun des modules ainsi défini communique avec les modules avec lesquels il doit échanger des informations grâce à un protocole de communication adapté. Pour les versions du système développées en C sous le système Unix, on utilise les outils mis à notre disposition dans ce cadre pour mettre au point ces échanges. Nous détaillerons cette mise en oeuvre dans le chapitre V de ce manuscrit.

Cette modularité rend chaque partie interchangeable sans affecter les autres composantes, et on pourrait même remplacer l'acquisition et le décodage acoustico-phonétique par une entrée du texte en langue naturelle, comportant éventuellement des fautes de frappe ou d'orthographe. L'intérêt de ce type d'expériences est de mettre en évidence la grande complexité de la reconnaissance de la parole continu, et de permettre une mise au point plus facile du système face à certaines erreurs des niveaux bas, l'entrée textuelle pouvant être entâchée, "*à la demande*", d'erreurs

sur mesure pour le point particulier à tester.

Les différents modules de Partner

1. Pour la version opérationnelle, fonctionnant avec entrée vocale, les deux premiers modules permettent, pour le premier de faire une acquisition suivie d'un décodage acoustico-phonétique, pour le second de reconnaître à le demande les mots d'un sous-lexique passé en paramètre. Ce sous-lexique sera à la fois fonction de l'application choisie pour la session en cours, mais aussi de la langue sélectionnée. Ces deux modules sont de larges sous-ensembles des modules APHON et LEX développé pour le système DIAL.

2. A partir des mots validés par LEX, un module de reconnaissances de phrases CASSIS tente de construire une structure syntaxique arborescente. La grammaire est elle aussi un paramètre dépendant de l'application et de la langue choisies, et servira à émettre des hypothèses concernant les mots à reconnaître. Il est donc guidé par par la syntaxe qui donne les mots attendus à chaque pas de l'analyseur. Toutefois, pour permettre au locuteur une certaine liberté par rapport au langage défini par la grammaire, on retrouve au niveau de ce module des possibilités d'élision de mots, de substitution d'un mot par un autre ou d'insertion de mot ou de bruit parasite en cours d'énoncé. Le résultat fourni correspond à la structure syntaxique d'une phrase correcte pour la grammaire, quel que soit le mécanisme qui a permis son analyse. Cette phrase est supposée être la plus "*proche*" de la phrase émise par le locuteur, cette proximité étant évaluée en fonction des mécanismes d'omission, d'insertion ou de substitution. Le score associé à cet structure dépendra à la fois des scores de reconnaissance des différents mots qui la composent, ainsi que de l'importance des plages non utilisées (bruits, mots parasites ou non reconnus). Chaque mot élidé sera supposé reconnu avec un score nul.

   Exemple de structure fourni pour l'énoncé :

```
"Prend C_un sur la pile alpha"  produira :

I recognize :
Tree = prend[78] le[0] cube[0] c_un[75] sur[66] la[50] pile[80]
alpha[100]
Score = 56
I understand :
Semantic = [ ACT=TAKE[61](OBJ=CUBE[48](LIEU=ON[66](OBJ=PILE[70]
```

```
                (PILENAME=ALPHA[100]())), CUBENAME=C_ONE[75]())) ]
Score = 56
```

3. Le module de gestion de la tâche VOC gère le bon déroulement de l'application. Les tâches prises en compte par Partner étant relativement simples (en général commande d'un système ou d'un processus), le modèle correspondant se réduira aux seules possibilités de cet univers. La tâche est modélisée par un automate, une commande possible est associée à une séquence d'états à atteindre, et chaque besoin correspond à un état.

   Pour résoudre un besoin, la tâche guide le dialogue en émettant des requêtes vers ce module.

4. Le module de gestion du dialogue, —DIALOG—, va regrouper tout ce qui est spécifiques au dialogue : gestion des entrées–sorties, gestion de l'historique des échanges déjà faits, évaluation de la situation du dialogue en cours et enfin décision de la conduite à adopter : Y-a-t-il lieu de poser une question à l'usager?

   De plus, en cas de reprise, de contestation, il est indispensable de resynchroniser la tâche en fonction des informations remplacées ou invalidées.

Tous les échanges entre les modules VOC et DIALOG se font à l'aide d'un protocole particulièrement simple : c'est toujours le module VOC qui a l'initiative, et il pose des questions à DIALOG. Une réponse ou une réaction est obligatoire avant la poursuite des traitements.

Les échanges de DIALOG vers les niveaux infra-lexicaux sont un peu plus complexes : chaque service possible correspond à un nom de fonction : lorsqu'une communication a pu être établie, un protocole se met en place à l'initiative de DIALOG. Une demande de service est suivie d'une réponse souvent multi-valuée de la part des modules lexicaux et infra-lexicaux, indiquant d'une part si la fonctionnalité demandée a été mise en oeuvre avec succès, et éventuellement en fournissant le (ou les) résultat(s) des traitements demandés.

3.3 Partie lexicale et infra-lexicale

Le niveau lexical a plusieurs rôles dans Partner : celui de pourvoyeur d'informations, celui de vérificateur d'hypothèses, et enfin celui d'interface avec les niveaux bas d'acquisition, de prétraitement et de décodage.

Chronologiquement, la première demande faire par Partner est d'acquérir un énoncé de l'usager. Cela déclenche l'activation du module d'acquisition, suivie

d'un enchaînement automatique des modules de segmentation et d'étiquetage. Le résultat rendu concerne des paramétres fournis par ces trois processeurs, comme la longueur de l'énoncé ou le nombre d'étiquettes trouvées.

Le deuxième échange concerne le lexique ou sous-lexique à employer : le système étant doublement paramétrable par d'une part l'application, d'autre part la langue, un lexique est associé à chacune des combinaisons possibles. Un nouveau lexique sera chargé en mémoire à la demande du module de dialogue, un changement de langue pouvant intervenir en milieu de dialogue si l'usager le désire!

Le troisième type d'échanges concerne directement les modules de traitements lexicaux : les demandes émises correspondent à des mots succeptibles d'être présents dans l'énoncé courant. Une première demande pour un mot donné est toujours une recherche de ce mot sur une plage de signal donnée. Si ce mot n'est pas rejeté, on fournit d'une part ses limites exactes dans le signal, et d'autre part un score, estimant la plausibilité que la plage trouvée corresponde effectivement au mot demandé.

Lorsque le mot attendu est rejeté, les niveaux hauts peuvent tout de même continuer l'analyse dans cette voie. Le mot est alors supposé omis. Si une analyse de ce type réussi à se poursuivre jusqu'à son terme, une deuxième demande sera émise pour les mots non reconnus : pour calculer une pénalité d'élision tenant compte de la longueur habituelle de ces mots, on va en demander un calcul approximatif, en fonction à la fois de la représentation phonétique associée à ces mots, et de la durée vocalique moyenne de l'énoncé que nous sommes en train d'analyser. Cette longueur servira donc à ajuster le score global de la phrase a posteriori.

3.4 Analyseur et interpréteur : notions de langage pivot

Chaque besoin de la tâche est défini par un sous-langage particulier. Ces sous-langages sont décrits à l'aide de grammaires sémantiques, c'est-à-dire des grammaires qui regroupent en un même modèle les informations syntaxiques et sémantiques. Les non-terminaux utilisés correspondent à la fois à des classes grammaticales et à des classes sémantiques liées à l'application mise en oeuvre. Des non-terminaux variables permettent de factoriser sous un même non-terminal des réalisations syntaxiques différentes d'une même structure sémantique. Les non-terminaux variables contiennent un symbole particulier : "–" qui va séparer une partie gauche variable d'une partie droite appelée extension, qui pourra se réaliser syntaxiquement de différentes manières.

Exemple :

```
<verb-del-infinitif> ---> Detruire
<verb-del-imperatif> ---> Detruis
```

La primitive "EXTEND" permet d'accéder à l'extension, tandis qu'une fonction "VALUE" permet d'obtenir la réalisation syntaxique

Exemple : Dans la phrase : Phrase ::= ...<verb-del>(détruis) ...
EXTEND(<verb>, Phrase) renverra "del" et
VALUE(<verb>, Phrase) renverra "détruis"

Cette technique permet d'obtenir une représentation sémantique canonique directement, quelle que soit la réalisation syntaxique trouvée dans l'énoncé prononcé par le locuteur. La fonction VALUE va servir à mémoriser cette réalisation syntaxique dans l'historique, car elle peut être utile ultérieurement pour résoudre une ellipse ou une anaphore.

Comme la représentation sémantique d'une commande à un système ne dépend pas de la langue, seuls les niveaux terminaux de la grammaire devront être modifiés, tout le reste de la grammaire sera invariant quand on passera d'une langue à l'autre. Ceci est dû à la nature des applications traitées, car il y a une correspondance parfaite entre la représentation sémantique d'un énoncé, et l'interprétation qui en est faite pour aboutir à l'exécution de l'ordre ainsi passé.

4 Le système DIAPASON

4.1 Présentation

Ce système est fortement inspiré du système MYRTILLE I développé par Jean-Marie Pierrel [PIERREL 75]. C'est un système de commande à une console sonar (**dia**logue **p**our une **a**pplication **sonar**), et développé conjointement par le CRIN et par Thomson DASM (direction des activités sous-marines) avec le soutien de la DRET (direction des recherches et études techniques du ministère de la défense) [ALINAT 87]. Son but est d'augmenter la convivialité d'une console sonar par adjonction d'un système de compréhension de phrases, doté d'un dialogue minimal permettant de lever des ambiguïtés ou de corriger des erreurs ponctuelles. Ce langage n'a pas été défini a priori pour un système de reconnaissance vocale, et le vocabulaire, bien que restreint (réduit à 71 mots), est difficile, surtout à cause de la présence des dix chiffres 0–9. La partie basse est en principe développé chez Thomson, mais a pu être remplacé pour des raisons de tests mutuels au CRIN

par les niveaux lexicaux de vérification de mots et les niveaux infra-lexicaux du système DIALOG correspondants.

L'architecture de ce système est essentiellement descendante, jusqu'à la vérification de mots. La partie basse reste toutefois ascendante (segmentation et étiquetage). Le schéma de cette architecture est donné par la figure 3.2

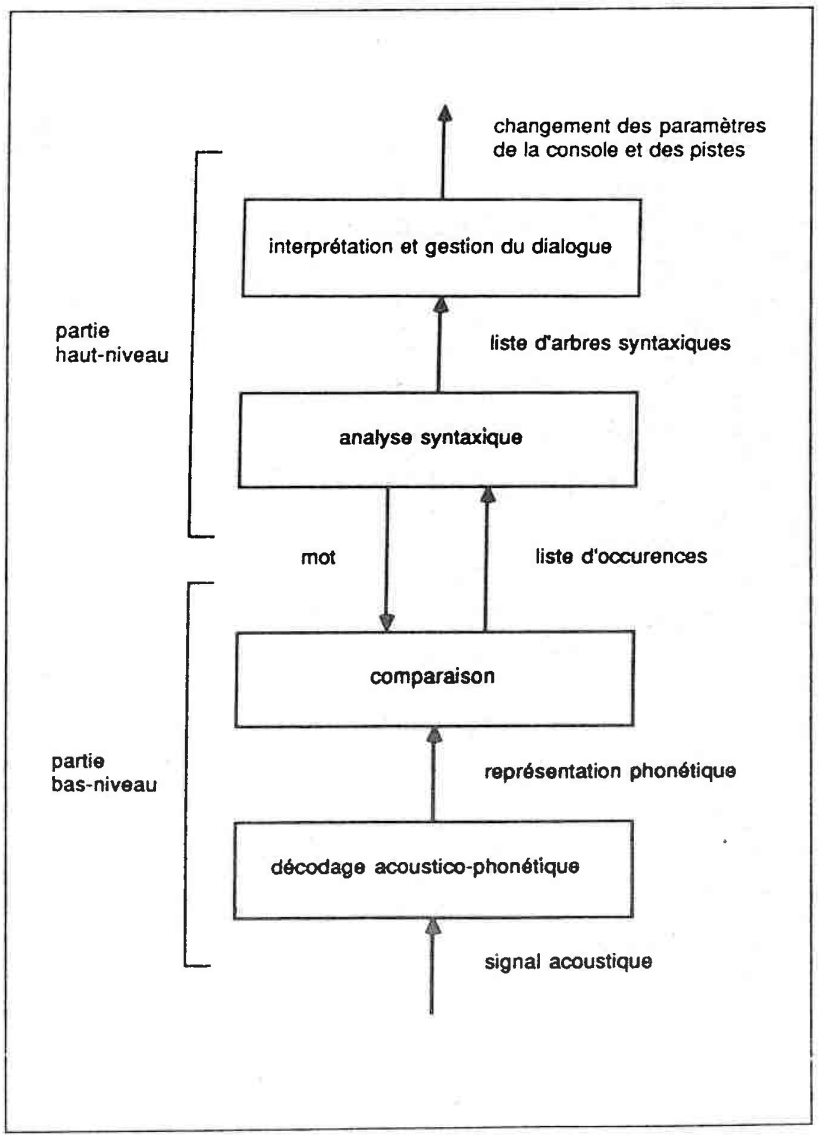


Figure 3.2: Architecture du système Diapason

4.2 Partie lexicale et infra-lexicale

Le décodage acoustico-phonétique se réduit à une première phase de segmentation grossière, suivi d'un étiquetage des segments ainsi obtenus par les deux ou trois phonèmes compatibles avec ces segments.

La partie correspondant à PROSO d'extraction d'indices prosodiques n'a pas été mise en oeuvre.

Seule la partie vérification d'une hypothèse mot sur une plage de signal donnée a été utilisée. Les seules informations mises en oeuvre ici sont les représentations phonétiques exactes des mots du vocabulaire, ainsi qu'une base de donnée indiquant pour chaque phonème des plausibilités de confusion avec des phonèmes peu différents, d'élision pure et simple, ou d'apparition spontanée (insertion dans la chaîne donnée comme résultat du décodage, due soit à une erreur de segmentation, soit à un phénomène phonologique).

Quelques fonctions supplémentaires ont été implémentées telle que possibilité d'une fin de signal de parole erronée. En particulier, on pourra abandonnée la partie de signal restant en fin d'enoncé si celui-ci est complet, et si cette partie peut correspondre à du bruit.

4.3 Partie haut-niveau

Cette partie est aussi assez restreinte, et seules des informations syntaxiques et sémantiques sont mises en oeuvre à ce niveau. Comme le langage traité est de type artificiel, il est entièrement décrit par une grammaire hors contexte, non récursive à gauche.

Analyse syntaxique

Le contrôle est entièrement descendant à ce niveau : Guidé par la grammaire du langage autorisé, l'analyseur émet des mots hypothèses pour la plage courante de signal à analyser. Chaque ainsi pressenti fera l'objet d'une demande de confirmation explicite envoyée au niveau lexical. Lorsqu'un mot est considéré comme reconnu (plausibilité de présence acceptable, vu le contexte), on recherchera le mot suivant, dans l'ordre d'énonciation de la phrase. Une deuxième stratégie plus résistante aux erreurs ou aux mauvaises reconnaissances des débuts d'énoncé consiste à créér des ilôts de confiance, le plus souvent en milieu d'énoncé, ou tout du moins à un endroit a priori quelconque de cet énoncé, puis d'élargir ensuite ces ilôts pour les raccorder.

Pour chaque mot ainsi pressenti, le module de vérification lexicale va retourner un score d'autant plus élevé que le mot a été mieux reconnu. A une partie d'énoncé déjà étiquetée en terme de mots, correspond un score résultant de la combinaison des scores de chacun des mots. Si ce score descend en dessous d'un seuil de rejet "global", l'analyse en cours est abandonnée. Toutefois, certains non-terminaux associés à des groupes de mots sont parfois considérés comme terminaux, et conservés avec leur score.

La stratégie de parcours peu varier en cours d'analyse. Un premier passage se fait avec une stratégie du meilleur d'abord. La situation considérée comme la plus plausible est développée jusqu'à l'impasse. On fait alors un retour arrière vers la meilleure solution restant mémorisée. En cas d'impossibilité totale, on déclenche une deuxième phase, capable de reconnaître des phrases avec élision, substitution ou insertion de mots parasites. Cette deuxième phase applique une stratégie de recherche en faisceau, chacune des possibilités étant éxaminé de façon parallèle.

En plus des contraintes syntaxiques et sémantiques, on peut préfiltrer les hypothèses en fonction de l'état courant du sonar. Ainsi, à la suite du mot "effacer", on ne pourra trouver (en principe) que le nom d'une piste affichée à l'écran. Chaque piste étant référencée par une séquence lettre–chiffre, le nombre de possibilités passe de 260 au nombre de pistes réellement affichées, en général moins d'une dizaine.

interprétation et dialogue

Le but final du système est de reconnaître la commande associée à la phrase reconnue, et de lancer son exécution. Un petit dialogue peut être nécessaire pour lever des ambiguïtés, faire des demandes de confirmation, accepter des corrections. Les ellipses et les références anaphoriques seront traitées grâce à l'historique des derniers échanges, dans lequel les structures profondes des énoncés du locuteur seront conservés. Cela permet une suite de commandes plus naturelle du type :

- afficher alpha 3
- béta 1
- charly 2

La première commande est complète, et correspond bien à la grammaire. Les deux suivantes ne correspondent qu'à des noms de pistes, et la partie commande qui correspond à l'action à accomplir a été omise. On supposera dans ce cas une extention de la dernière commande référencée, –*afficher*–, à ces deux pistes.

Selon l'importance de la commande, trois niveaux de confirmation sont prévus :

1. confirmation explicite exigée : l'utilisateur doit valider sa commande avant exécution.
2. confirmation implicite : la commande sera exécutée si l'utilisateur n'émet pas de contestation.
3. confirmation inexistante : l'exécution est immédiate.

Chapitre 4

Mise en oeuvre du niveau lexical

1 Introduction

Chacun des systèmes que nous avons décrits au chapitre précédent est en cours de réalisation, et un certain nombre de modules sont déjà opérationnels. En particulier, plusieurs versions de Partner et une version de Diapason ont été implémentées. Le projet Dialog étant plus important, seules des réalisations plus ponctuelles ont été testées, mais la mise au point imminente des analyseurs syntaxico-sémantiques devrait bientôt permettre des tests en vraie grandeur, en interconnectant tous ses constituants, car les niveaux APHON, PROSO, et LEX peuvent fournir les services attendus en aval, et le module DIAL est lui aussi opérationnel. Toutefois, les interfaces restent à mettre au point entre LEX et SYNTSEM d'une part, et surtout entre SYNTSEM et DIAL, ce qui risque d'être un peu plus compliqué, car cette dernière interface nécessite une transformation de la structure syntaxico-sémantique obtenue en une structure prédicative interprétable par DIAL.

L'approche phonologique du module APHON consiste à considérer que les constituants de base d'un énoncé de parole sont les phonèmes. Notre unité phonologique de base restera donc le phonème dans toute la suite de ce chapitre, même si le décodage se fait de façon contextuelle, en utilisant les deux voisinages gauche et droit, ce qui pourrait conduire à étudier des triphones.

Ce choix n'était pas le seul possible : d'autres systèmes utilisent des unités de base différentes pour segmenter et étiqueter le signal de parole. Shoup [SHOUP 79] dans [LEA 80] nous en propose un certain nombre :

1. L'allophone[1] : Shoup définit un allophone comme un ensemble de phones, qui possèdent tous les mêmes traits. Un même phonème peut engendrer plusieurs allophones, si plusieurs réalisations de ce phonème possèdent

[1] Ce terme est généralement utilisé pour désigner l'unité phonétique minimale, mais aucune

des traits distincts. Cette approche semble intéressante, les différents allophones permettant de lever des ambiguïtés syntaxiques ou sémantiques, mais le nombre de ces allophones peut s'avérer important (un allophone pourrait correspondre à chaque type de co-articulation) et surtout il n'y a pas encore à ce jour d'algorithmes suffisamment sophistiqués pour permettre leur identification.

Exemple :

Annick lassait les gens

et

Annie classait les gens

ont la même transcription phonétique. Seul le /*i*/ de Annie est plus long, ce qui permet de distinguer la séquence i-consonne dans un mot, de la séquence i-fin_de_mot-consonne, et de lever ainsi l'ambiguïté.

2. Le phonème : c'est donc l'unité retenue au CRIN. Son grand avantage réside dans le petit nombre de phonèmes, et dans le codage aisé, car naturel, des mots à reconnaître. Les difficultés sont doubles : les algorithmes de segmentation sont complexes, et surtout les phonèmes sont le plus souvent coarticulés avec leurs voisins, donc variables.

3. Les diphones : la plus grande partie de la variabilité des phonèmes réside dans les transitions entre une consonne et une voyelle. Une idée simple était de prendre une unité qui englobait cette information sur les transitions. Si la variabilité disparaît, Shoup déplore deux inconvénients majeurs : le nombre de diphones est important (de l'ordre du millier) et, si les règles phonologiques concernant les phonèmes sont connues, elles ne semblent pas s'appliquer facilement à des diphones.

4. Les syllabes ou demi-syllabes : les syllabes sont faciles à localiser, ou plutôt leurs centres (constitués par un noyau vocalique) sont aisément repérables. Leurs limites exactes sont plus difficiles à déterminer. Là encore, la coarticulation est incluse. Gunther Ruske et Walter Weigel [RUSKE 86] utilisent les demi-syllabes pour la version allemande de Partner développée à Munich.

---

définition rigoureuse n'existe. Plusieurs réalisations différentes d'un même phonème seront autant de phones différents, appelés allophones. Un phonème sera l'ensemble des allophones possibles pouvant représenter un même son.

La segmentation médiane se fait à l'endroit où l'énergie de la voyelle est la plus forte, et les bords sont trouvés en optimisant les distances relatives aux deux centres de voyelles de part et d'autre.

5. Le mot lui-même : la reconnaissance globale est simple si le vocabulaire n'est pas trop étendu, mais on pourra difficilement traiter les grands vocabulaires de cette façon. De plus, une adaptation au locuteur — souvent réalisée par un apprentissage toujours fastidieux — est indispensable.

Le module APHON fournit donc, selon les versions, soit des listes de phonèmes associés à des scores de plausibilité (nous utilisons une version de ce type pour nos tests actuels), soit un véritable treillis phonétique obtenu en prenant en compte les contextes gauche et droit du segment analysé , et où seuls les chemins valides (correspondant aux deux contextes étudiés) sont possibles. Cette nouvelle version est opérationnelle depuis fin juin 1988.

Pour LEX, plusieurs outils ont été développés pour permettre une saisie conviviale du lexique et des transcriptions phonétiques de chaque mot. Un calcul automatique de patron et un accès par ce patron a été mis en place pour permettre la recherche rapide de mots. Enfin, trois bases de connaissances ont été constituées et testées pour permettre la vérification non aveugle de mots sur le treillis, tenant compte, de la façon la plus juste possible, de la phonologie, mais surtout des limites et des erreurs du système de décodage.

2 Constitution des lexiques phonétiques et phonologiques

La représentation phonétique d'un mot français ne se déduit pas de son orthographe de façon immédiate. Si des programmes permettant une transcription graphèmes/phonèmes existent, ils restent imparfaits et lourds à mettre en oeuvre. De plus, ils ne s'appliquent pas dans le cadre de mots isolés, beaucoup de mots ayant une même transcription orthographique ont des prononciations différentes selon leur nature. Un exemple bien connu nous en fournit l'illustration :

Les poules du couvent [/k/u/v/ã/] couvent [/k/u/v/ə//].

Or, pour constituer un lexique phonétique, on dispose le plus souvent de la liste des mots qui devront être connus du système. Ces algorithmes peuvent donc difficilement s'appliquer. Ils peuvent tout de même aider le concepteur à créer une première transcription grossière, qui devra être validée dans un deuxième temps. Cette approche sera conseillée pour les très grands vocabulaires.

Ce type de travail n'est pas à reprendre à zéro pour chaque nouvelle application. Par exemple, le GRECO communication parlée a développé une base de donnée lexicale du français BDLEX, pour constituer un dictionnaire adapté au traitement automatique de la parole. Ce lexique, développé à Toulouse par l'équipe de Guy Pérennou [PERENNOU 86], est très complet et compte 25.000 entrées (ce qui représente environ 300.000 formes fléchies). C'est une base de données relationnelle, qui comporte trois composantes :

- une composante phonologique
- une composante morphologique
- un module permettant son interrogation ainsi que sa mise à jour.

La composante phonologique fonctionne à l'aide d'un ensemble de règles, fournies par des phonologues.

Pour les grands vocabulaires, on peut donc envisager un programme d'extraction d'informations d'une base importante telle que BDLEX, cette opération ne devant être effectuée qu'une seule fois, après avoir déterminé le lexique de l'application.

Pour les applications plus restreintes, cette approche nous semble un peu lourde. De plus, il n'est pas indispensable de représenter toutes les altérations phonologiques de chaque mot, il suffit de savoir au moment de la reconnaissance de ce mot si une déformation de la forme à reconnaître peut conduire ou non à la forme inconnue en cours d'examen.

Là encore, deux approches sont permises :

- une approche déclarative, sous forme de règles de transformation
- une approche procédurale, où les informations sont codées dans les algorithmes de comparaison eux-mêmes.

Chacune de ces approches a ses avantages : la première permet un ajout aisé de nouvelles règles, ou l'affinage des règles déjà présentes. De plus, ces modifications peuvent être faites par un phonologue non informaticien.

La deuxième approche met sur un pied d'égalité les altérations phonologiques et les limites ou les erreurs du module de segmentation et d'étiquetage phonétique. Elle permet des traitements plus pragmatiques, en regroupant deux problèmes assez proches, et d'ailleurs non disjoints. Elle permet aussi d'éviter d'envisager des transformations peu réalistes, que l'on pourrait obtenir en appliquant de façon successive deux ensembles de règles — les premières concernant la phonologie, les

secondes explicitant les connaissances a priori sur les limites actuelles des modules de décodage —.

Pour les trois applications que nous mettons en œuvre au CRIN, nous avons choisi d'implémenter ces connaissances de façon procédurale, en regroupant des connaissances phonologiques, ces dernières traduisant les limites du module de décodage utilisé.

2.1 Partie purement phonétique

Les concepteurs de lexiques phonétiques et phonologiques peuvent implémenter leurs connaissances de plusieurs manières :

1. implémenter un lexique contenant à la fois les informations phonétiques et phonologiques.

2. séparer complètement les deux types de connaissances, en implémentant deux modules distincts, l'un purement phonétique, l'autre prenant en compte les aspects phonologiques. C'est le choix que nous avons retenu pour LEX.

Compte-tenu du nombre d'applications développées en parallèle au CRIN, nous avions à constituer plusieurs lexiques, chacun d'entre eux devant comporter la description phonétique des mots d'une application donnée. Comme nous ne disposons pas de programme permettant une passage automatique graphème/phonème, nous avons choisi d'entrer les transcriptions standard des mots sous leur forme la plus naturelle possible, à la fois pour rendre les données du programme de lecture les plus lisibles possibles, mais aussi pour permettre une vérification plus facile de ces données. En effet, toute erreur subsistant à ce niveau va grever directement les performances de la reconnaissance, et la détection ultérieure sera difficile, car il est rare qu'un mot soit parfaitement décodé (une erreur serait alors mise en évidence si ce mot *parfait* n'avait pas un score de reconnaissance de 100%).

Outre cette description phonétique, le lexique servant à hypothétiser les mots et à les vérifier contient trois autres informations :

1. la forme orthographique de chaque mot pour l'affichage des mots reconnus, et pour une vérification plus aisée de la cohérence des codages, rarement identiques, entre les différents modules.

2. des informations syntaxiques et sémantiques permettant les transferts d'informations entre le niveau LEX et les niveaux syntaxico-sémantiques.

3. un nombre identifiant du patron phonétique que nous détaillerons ci-dessous.

Pour la partie phonétique, plusieurs représentations internes seront utiles en fonction de l'usage que nous en ferons. Au niveau externe, il suffit de donner l'ensemble de ces informations sous une forme non ambiguë pour le système, et si possible aisément codable et compréhensible pour le concepteur. Un fichier texte, manipulable par tout éditeur, nous a semblé idéal. Chaque transcription orthographique de mot doit être suivie de sa transcription phonétique. Un identifiant numérique est attribué de façon automatique par le système.

Les codages des phonèmes doivent être mnémoniques et naturels, et nous avons choisi le transcodage donné par la figure 4.1.

Ce fichier va donc contenir de façon alternée une transcription orthographique et sa transcription phonétique standard. Un filtrage vérifie que chacun des phonèmes fournis correspond bien à l'un des codes connus du système. Après cette première passe, plusieurs traitements peuvent être appliqués pour construire des représentations lexicales internes, à l'usage d'un sous-module de LEX : vérification, hypothétisation rapide, ou tout simplement calcul d'informations demandées par un niveau supra-lexical : longueur moyenne de ce mot –en supposant qu'il soit présent–, dans l'énoncé courant, en fonction de la vitesse d'élocution. Celle-ci est évaluée en fonction de la durée vocalique moyenne, c'est-à-dire de la durée moyenne des noyaux vocaliques détectés dans l'énoncé en cours d'analyse.

2.2 Prise en compte de la phonologie et des résultats des niveaux infra-lexicaux

Il existe plusieurs moyens différents pour coder — et prendre en compte lors d'une comparaison ultérieure — les informations phonologiques.

1. La première est la plus simple à mettre en œuvre, surtout si le lexique n'est pas étendu. L'ensemble des transcriptions acceptables est codé de façon explicite, soit en multipliant le nombre de descriptions d'un même mot, soit en décrivant chaque mot avec des conventions un peu plus complexes, telle que celles adoptées par Jean-Marie Pierrel dans MYRTILLE I [PIERREL 75] et que nous rappelons dans la figure 4.2

2. Une deuxième approche consiste à séparer phonétique et phonologie, la partie phonologique s'exprimant alors sous la forme d'un ensemble de règles, comme le préconise SHOUP [SHOUP 79] dans l'ouvrage de Lea [LEA 80].

PHONEMES	EXEMPLE	CODAGE
i	cri	/i/
e	blé	/e/
ϵ	sel	/ai/
y	sucre	/y/
œ	peur	/eu/
∂	petit	/&/
ϕ	deux	/eu/
a	patte et pâte	/a/
$\supset$	pomme	/o/
o	trop	/o/
u	loup	/u/
$\tilde{\text{œ}}$	brin et brun	/in/
$\tilde{\text{a}}$	plan	/an/
$\tilde{\text{o}}$	thon	/on/
w	oui	/w/
j	bille	/j/
s	bosse	/s/
z	zibeline	/z/
$\int$	chou	/ch/
z	jeu	/gh/
f	fou	/f/
v	vite	/v/
n	non	/n/
m	mine	/n/
nj	signe	/gn/
l	ligne	/l/
r	rat	/R/
p	pli	/p/
t	tarte	/t/
k	cour	/k/
b	boue	/b/
d	don	/d/
g	gui	/g/

Figure 4.1: Codage des différents phonèmes pour l'entrée du lexique phonétique.

Exemple de règle (BBN)[WOODS 76] :

FOR ALL IVOBS-FRIC [2] SEQUENCES, THE TWO SEGMENTS ARE REPLACED (NON-OPTIONNAL) WITH A SINGLE FRIC

[2]IVOBS pour Inter-Vocalic-Obstruent en anglais représente quatre phonèmes occlusifs voisés, /v/, le th voisé du mot /that/, le /h/ aspiré du mot /hat/ et le "*flapped*" /t/ du mot ba*tt*er. Ces quatre phonèmes seront donc réunis à la fricative voisine de façon systématique d'après cette règle

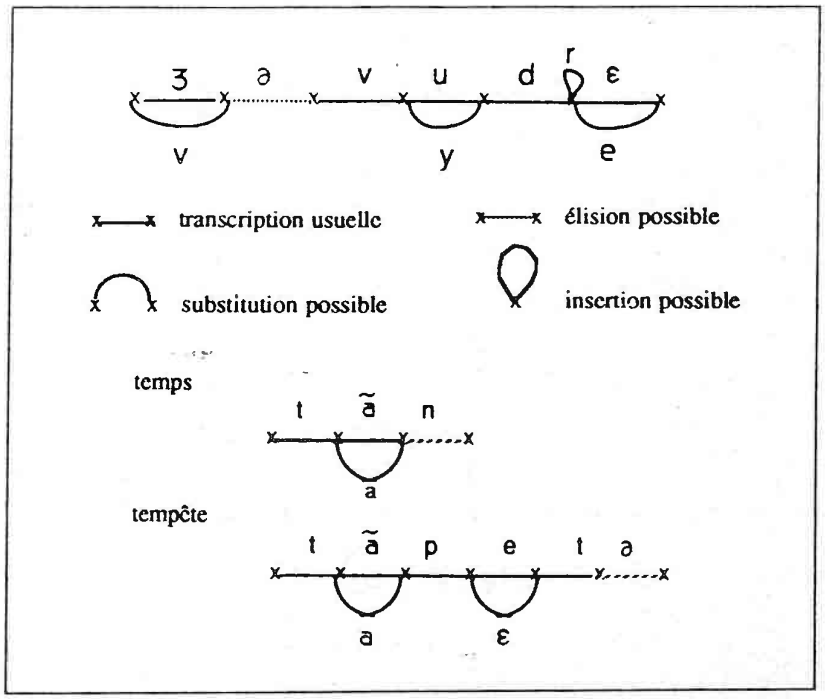


Figure 4.2: **Exemple de représentation de mots permettant des élisions de phonèmes**

3. Enfin, au lieu d'écrire les règles de façon formelle, on peut les traduire directement sous forme d'algorithme dans les modules de vérification lexicale. Cette méthode permet en outre de prendre en compte au même niveau des connaissances pragmatiques, traduisant les performances ou les limites du décodage, sans redondance avec des altérations phonologiques.

Nous avons choisi de ne pas introduire d'informations phonologiques au niveau de la représentation phonétique exacte du mot. Comme le décodage acoustico-phonétique ne fournit quasiment jamais une transcription exacte ou qui se déduit de la transcription exacte par une transformation phonologique prévisible, une comparaison souple et intelligente doit être mise en œuvre lors de la vérification des mots. Beaucoup d'erreurs d'étiquetage rejoignent des altérations phonologiques, les deux phénomènes n'étant pas totalement disjoints : lorsqu'un locuteur fait une assimilation ou une autre altération, il le fait dans un but simplificateur vis-à-vis de son système articulatoire. Les règles phonologiques d'altérations connues ne font que constater cet état de fait, et sont liées à des causes *mécaniques* (une séquence phonétique un peu dure à produire s'est vue simplifier pour faciliter sa

production). Toutefois, ce phénomène est constant dans toute production continue de parole, le locuteur ne fait que se contenter d'approcher suffisament ses cibles phonétiques pour render audible son message. Les "*erreurs*" du décodage ne sont donc parfois que des imperfections, le phonème fourni étant une interprétation non-contextuelle du son relevé. Il est à noter que l'auditeur humain aura beaucoup de mal à étiqueter un son ôté de son contexte ou l'étiquettera de façon différente dans des contextes opposés.

Les altérations phonologiques et les erreurs de décodage seront donc traitées simultanément par la procédure de vérification lexicale, et feront appel à une même base de connaissance pour estimer la plausibilité du phénomène rencontré. Bien sûr, les altérations phonologiques prévisibles pénaliseront peu, — ou même ne pénaliseront pas —, les scores de reconnaissance.

La procédure de vérification utilise deux mécanismes pour modéliser ces phénomènes :

1. des connaissances statiques, et non contextuelles.

   Pour chaque phonème, trois informations sont disponibles :

   (a) un coût d'élision : il est d'autant plus important que ce phonème est bien reconnu par le système, et qu'il est rarement altéré.

   Par exemple, /*a* / se verra attribuer une forte pénalité d'élision et le /*∂* / muet une pénalité nulle.

   (b) un coût d'insertion : nous avons remarqué que peu de phonèmes des trois classes majeures étaient insérés par les modules de prétraitements. Par contre, des erreurs de segmentation (coupure trop courte d'une voyelle par exemple) génèrent souvent des insertions de liquides si la voyelle est orale, et des insertions de nasales sinon. De même, des sons fricatifs peu énergétiques tels que /*f* / peuvent voir une de leurs parties tronquée et étiquetée comme une plosive, telle que /*p* /.

   De plus, beaucoup d'altérations phonologiques conduisent à des nasalisations, ou des semi-nasalisations, qui conduisent au même résultat, sans que cette insertion soit liée à une imperfection du système de décodage. Une pénalité de base est donc associée à chaque phonème. Cette pénalité de base pourra être modifiée lors du traitement, de façon contextuelle. En particulier, lors d'une vérification de mot qui doit couvrir exactement la plage fournie, une insertion de plosive sourde en tête ou en fin de mot est ignorée. De même, une insertion d'une liquide ou d'une nasale très courte sera beaucoup moins pénalisante qu'une insertion d'un même phonème mais long.

(c) une matrice de confusion : à chaque phonème correspond un coût de remplacement par chacun des autres. Là encore, ce coût de remplacement sera d'autant plus faible si :

- Une altération prévisible permet ce remplacement : /*n* / → /*d* /
- Les deux phonèmes sont proches articulatoirement : /*i* / et /*é* /
- Les deux phonèmes sont parfois confondus à cause d'une imperfection du système : /*é* / → /*l* / correspondra par exemple à un /*é* / non détecté par la segmentation en tant que noyau vocalique. /*õ* / → /*é* / correspond à un /*õ* / qui a vu ses formants mal étiquetés : le second formant atténué par la nasalition a disparu pour le système, et le formant 3 joue ce rôle. La voyelle dont les valeurs formantiques sont les plus proches devient alors un /*é* /. Un exemple d'une telle matrice de confusion nous est donné par la figure 4.3

2. Toutes les autres connaissances, faisant appel au contexte, sont prises en compte de façon procédurale. Toutes les connaissances statiques que nous venons de décrire pourront être modifiées en fonction des connaissances particulières dont nous disposons au moment de leur usage :
Un /*i* / est souvent court, donc a priori c'est la voyelle qui est la plus souvent omise, soit complètement, soit par substitution par /*l* /. Ce phénomène va s'aggraver si cette voyelle se trouve en contexte défavorable, par exemple entre deux fricatives sourdes :

saucisson → /s/o/s/i/s/õ/ → /s/o/s/õ/

Ce genre de règles aurait pû aussi être implémenté de façon déclarative, mais nous avons préféré les coder directement dans l'algorithme de comparaison, car des mesures de longueur de phonème interviennent souvent, en particulier dans ce cas précis : le son /*s* / résultant sera "*plus long que la normale* ". De plus, à terme, nous envisageons de faire des demandes de vérification de l'hypothèse utilisée (exemple : disparition d'un /*i* / entre deux /*s* /) au module de décodage acoustico-phonétique, au lieu de nous contenter de réviser à la baisse une pénalité correspondant à un phénomène ayant une explication plausible.

3 Vérification de mots

Avec un lexique réduit, le niveau lexical peut vérifier chacun des mots du lexique sur tout le signal, pour construire un treillis lexical. Ce type de

	i	e	ɛ	y	œ	ə	a	o	u	œ̃	ã	ɔ̃	w	j	ɥ	z	ʃ	ʒ	f	v	n	m	nj	l	r	p	t	k	b	d	g
i	0	2	3	1	6	6	x	x	x	x	x	x	x	x	x	x	x	4	x	x	x	x	x	4	x	x	x	x	x	x	x
e	2	0	0	6	3	2	3	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	5	x	x	x	x	x	x	x
ɛ	3	0	0	x	4	5	4	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	4	x	x	x	x	x	x	x
y	1	x	x	0	2	2	x	4	3	x	x	x	x	x	x	x	x	x	x	x	x	x	4	x	x	x	x	x	x	x	x
œ	x	x	x	x	0	0	1	2	x	x	x	x	x	x	x	x	x	x	x	x	5	x	x	x	x	x	x	x	x	x	x
ə	x	3	x	3	0	0	5	3	5	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
a	x	x	x	x	2	4	0	5	x	1	1	x	x	x	x	x	x	x	x	x	x	x	x	5	x	x	x	x	x	x	x
o	x	x	x	x	1	3	x	0	1	3	3	1	x	x	x	x	x	x	x	x	x	x	x	5	x	x	x	x	x	x	x
u	x	x	x	3	2	3	x	1	0	x	x	4	x	x	x	x	x	x	x	x	x	x	x	4	x	x	x	x	x	x	x
œ̃	x	x	x	x	2	x	1	x	x	0	1	1	x	x	x	x	x	x	x	x	x	x	x	4	x	x	x	x	x	x	x
ã	x	x	x	x	3	x	1	3	x	1	0	1	x	x	x	x	x	x	x	x	x	x	x	4	x	x	x	x	x	x	x
ɔ̃	x	x	x	x	3	x	x	1	x	1	1	0	x	x	x	x	x	x	x	x	5	x	x	x	x	x	x	x	x	x	x
w	x	x	x	x	x	x	1	x	x	2	0	3	0	x	x	x	x	x	x	x	x	x	x	x	0	x	x	x	x	x	x
j	1	x	x	x	x	x	x	x	x	x	x	x	x	0	8	3	x	3	x	3	x	x	x	4	x	x	x	x	x	x	x
s	x	x	x	x	x	x	x	x	x	x	x	x	x	6	0	1	1	5	2	5	x	x	x	x	x	x	x	x	x	x	x
z	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	4	1	4	1	x	x	x	x	x	x	x	x	x	x	x
ʃ	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	4	0	1	2	4	x	x	x	x	4	x	x	x	x	x	x
ʒ	x	x	x	x	x	x	x	x	x	x	x	x	x	x	4	1	1	0	4	1	x	x	x	x	x	x	x	x	x	x	x
f	x	x	x	x	x	x	x	x	x	x	x	x	x	x	2	5	2	5	0	1	x	x	x	x	4	6	x	x	x	x	x
v	x	x	x	x	x	x	x	x	x	x	x	x	x	x	5	1	5	1	5	0	x	x	x	x	x	4	x	x	1	x	x
n	x	x	x	x	5	x	x	x	x	4	4	4	x	x	x	x	x	x	x	x	0	0	1	3	5	x	x	x	x	x	x
m	x	x	x	x	5	x	x	x	x	4	4	4	x	x	x	x	x	x	x	x	0	0	1	4	5	x	x	x	x	x	x
nj	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	[illegible]	1	[illegible]	[illegible]	x	x	x	x	x	x	x
l	x	x	x	4	x	x	x	x	x	x	x	x	x	5	x	x	x	x	x	x	4	3	x	0	2	x	x	x	x	5	x
r	x	x	x	x	x	x	3	x	x	3	3	x	1	x	x	4	x	4	4	x	5	5	x	2	0	x	x	x	x	x	x
p	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	6	6	x	x	x	x	x	0	1	1	1	4	4
t	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	1	4	1	4
k	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	0	4	4	1
b	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	7	1	5	5	x	x	x	1	4	4	0	1	1
d	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	5	5	x	x	x	4	1	4	1	0	1
g	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	5	5	x	x	x	4	4	1	1	1	0

Figure 4.3: Exemple de matrice de confusion

méthode est employé dans les systèmes ascendants, et on utilise alors un lexique phonétique factorisé, pour optimiser le nombre de comparaisons. Un exemple de ce type de lexique est fourni en figure 4.4.

Lorsque le lexique est de grande taille, il est exclu de chercher aveuglément tous les mots qui le composent à n'importe quel endroit du signal. Les traitements supra-lexicaux vont nous servir à focaliser ce type de recherche, et seuls des mots déjà mis en avant par des informations de haut niveau

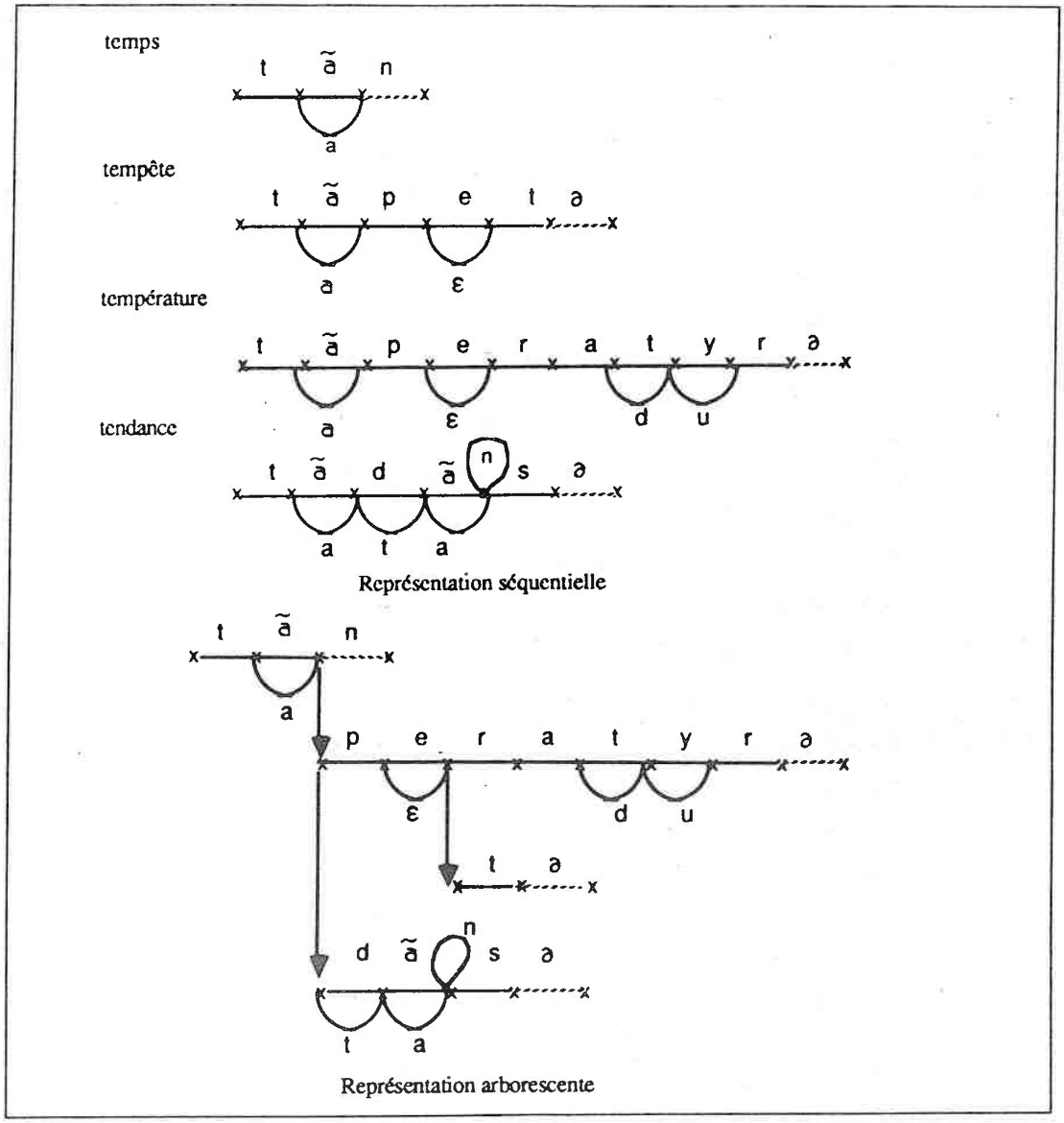


Figure 4.4: Exemple de lexique factorisé d'après [PIERREL 87]

seront à tester. Une procédure spécialisée a été développée pour vérifier la présence ou la possibilité de présence d'un mot dans une plage de signal donnée. Au niveau le plus interne, cette fonction de vérification admet cinq paramètres d'appel, qui sont :

(a) nummot : un identifiant du mot à tester, qui va permettre de récupérer les informations phonétiques qui le concernent dans la base de connais-

sance correspondante.

(b) code : un code précisant des contraintes sur le mot à trouver. Quatre possibilités sont prévues :

i. 0 : le mot peut se trouver n'importe où dans la plage donnée : d'autres mots ou du bruit peuvent apparaître avant et après lui dans la plage de recherche.

ii. 1 : le mot doit être cadré à gauche : il est un voisin immédiat d'un mot qui s'arrête au début de la plage de recherche, ou c'est un mot qui peut apparaître en début d'énoncé.

iii. 2 : le mot doit être cadré à droite : il est le prédécesseur immédiat d'un mot déjà trouvé juste à droite de la plage de recherche, ou c'est un mot qui peut terminer un énoncé.

iv. 3 : le mot doit couvrir exactement la plage donnée : ce mot va servir à terminer une analyse, et ses voisins sont déjà trouvés.

(c) début : un paramètre précisant le début de la plage de recherche.
Celui-ci va servir à renvoyer le début réel du mot en cas de succès.

(d) fin : un paramètre indiquant la fin de la plage de recherche.
Là encore, nous nous en servons pour retourner la fin effective d'un mot trouvé.

(e) score : un seuil indiquant la qualité minimale de la reconnaissance. En cas de succès, ce minimum est remplacé par la qualité évaluée du mot sur la plage considérée.

Lorsque l'on recherche des mots, le degré de confiance minimum pour retenir ces mots et surtout pour poursuivre une analyse dans une voie donnée va dépendre du degré d'avancement du processus, des scores des mots déjà trouvés. Ce seuil va donc varier de façon dynamique en fonction du contexte de reconnaissance, et il est indispensable pour appliquer une telle stratégie de passer ce seuil en paramètre de la procédure elle-même. Nous pouvons abandonner la vérification avant son échéance, si la dissemblance devient incompatible avec le seuil minimum exigé.

Voici les algorithmes simplifiés d'une fonction de vérification d'un mot sur une plage donnée :

```
Chercher (mot,typrech,debut,fin,score)
cumulmax = fct(score,longueur[mot]);
```

```
Si typrech = 3 alors verifmot(mot,debut,fin,0,cumulmax,1,debut)
Si typrech = 2 alors
   pour f variant de (debut + longueur[mot] -4) a
                     (debut + longueur[mot]  +2)
   faire si (f >= debut et f <= fintreillis)
         verifmot(mot,debut,f,0,cumulmax,1,debut)
Si typrech = 1 alors
   pour d variant de (fin - longueur[mot] -2) a
                     (fin - longueur[mot] +4)
   faire si (fin >= d et d >= 1)
         verifmot(mot,d,fin,0,cumulmax,1,d)

Si typrech = 0 alors
   pour d variant de debut a (fin - longueur[mot]  +4)
    pour f variant de (d + longueur[mot] -4) a
                      (d + longueur[mot]  +2)
    faire si (f >= d et f <= fintreillis)
          verifmot(mot,d,f,0,cumulmax,1,d)
%fin Chercher
```

Cette fonction suppose a priori que la variation de la longueur du mot en terme de phonèmes n'excède pas trois.

```
fonction verifmot(mot,debut,fin,cumul,cumulmax,courantmot
                 ,couranttreillis)

 tant que ( cumul < cumulmax et courantmot < longueur[mot] )
         faire
         meilleurc = INFINI;
         meilleuri = INFINI;
         meilleuro = omission(phonememot[courantmot]);

         pour phonemetr appartenant a treillis[couranttreillis]
         si confusion(phonemetr,phonememot[courantmot])
                 < meilleurc
            alors
         meilleurc = confusion(phonemetr,phonememot[courantmot])
```

```
si insertion(phonemetr) < meilleuri
   alors
meilleuri = insertion(phonemetr)
finpour


si (cumul + meilleurc < cumulmax) alors
   meilleurc = meilleurc +
    verifmot(mot,debut,fin,cumul+meilleur,cumulmax,
             courantmot+1,couranttreillis+1)
sinon meilleurc = INFINI

si (cumul + meilleuri < cumulmax) alors
   meilleuri = meilleuri +
     verifmot(mot,debut,fin,cumul+meilleur,cumulmax,
             courantmot,couranttreillis+1)
sinon meilleuri = INFINI

si (cumul + meilleuro < cumulmax) alors
   meilleuro = meilleuro +
    verifmot(mot,debut,fin,cumul+meilleur,cumulmax,
             courantmot+1,couranttreillis)
sinon meilleuro = INFINI

meilleur = min(meilleurc,meilleuri,meiileuro);
retourner(meilleur);
```

Une autre solution possible, mais moins intéressante en terme de temps de calcul, consiste à rechercher tous les mots avec le seuil minimum, et à ne retenir dans un deuxième temps que ceux d'entre eux dont le score dépasse le minimum courant exigé. De plus, si le vocabulaire est important, un sous-lexique devra être sélectionné avant cette recherche si la plage de recherche est étendue.

4 Recherche de mots

Nous avons déjà fait remarquer que la construction d'un treillis lexical complet, de façon ascendante avec un vocabulaire étendu était difficilement concevable, et

que cette méthode risquait de conduire à une explosion combinatoire au niveau des analyseurs lexicaux. Une hypothétisation de mots constitue donc un problème assez différent de celui que nous venons d'étudier. Peu d'auteurs l'ont abordé à ce jour. L'équipe de Guy Pérennou avait étudié ce problème dès 1976 [CAUSSE 76] . Les solutions proposées sont à mi-chemin entre les deux approches successives que nous avons testées. En effet, les mots sont découpés en syllabes, et chaque syllabe est ensuite codée sous formes de suite de 0,1,2 et 3

Les 0 correspondent au code des plosives, les 1 à celui des fricatives (ou plutôt à celui de toutes les consonnes autres que les plosives ou les liquides), les 2 à celui des semi-voyelles et des liquides, et enfin les 3 représentent les noyaux vocaliques.

Par exemple, le mot *particulier* devient :

/p/a/r/—/t/i/—/k/y/—/l/j/e/ → 032—03—03—2230

Il est à noter que la lecture de cet article est postérieure à la réalisation de la partie du travail qui lui est similaire. Les quatre classes retenues par l'équipe de Guy Perennou semblent être les mêmes que celle que nous avons nous-mêmes sélectionnées (et le codage a bien failli être strictement identique!!). Toutefois, le découpage du mot en syllabes conduit à des algorithmes de reconnaissance différents.

Nous avons donc développé une fonction de détection rapide de mots. Le but de celle -ci est assez différent de la précédente : il s'agit de détecter le plus rapidement possible une certain nombre de mots porteurs de sens, et avec un bon degré de confiance, pour donner des points d'ancrage aux analyseurs syntaxico-sémantiques, qui pourront ainsi initialiser une analyse par îlots de confiance. Lorsque le lexique est important, une technique consistant à rechercher tous les mots à tous les endroits du signal d'un énoncé pouvant contenir presque une centaine de segments est beaucoup trop coûteuse, quelles que soient par ailleurs les performances de la machine. En effet, une technique de programmation dynamique génère trois hypothèses à chaque pas pour chaque mot. En l'absence d'hypothèses des niveaux supérieurs (lès analyseurs ne pourront en faire que dans un deuxième temps), il convient de tester tous les mots à tous les endroits :
pour 1000 mots à vérifier sur un signal comportant 50 phonèmes, cela génère 50.000 possibilités de début, donc autant de vérifications à lancer. Celles-ci devront avancer de deux ou trois pas au minimum avant d'être rejetées, et cela peut conduire à examiner 27 sous-hypothèses (3^3) au titre de la programmation dynamique.

Nous avons donc imaginé une nouvelle technique de recherche, beaucoup plus rapide, et prenant en compte la spécificité du problème, ainsi que le type de

résultats fournis par le décodage acoustico-phonétique.

Nous ne recherchons que des mots avec un bon degré de confiance : nous nous limitons donc aux mots pour lesquels l'ensemble des segments plosifs, fricatifs et l'ensemble des noyaux vocaliques sont détectés. Seule la présence d'un segment de la classe voulue est éxigée, une erreur dans l'étiquetage final en terme de phonème n'est pas fatale. Cette contrainte n'est pas absurde ni trop forte compte-tenu des résultats des modules de prétraitement : chacun de ces constituants est correctement trouvé avec une fréquence supérieure à 90 %, voire 95 % pour des phrases enregistrées avec soin. Les derniers taux relevés pour une évaluation de notre système de décodage, effectués sur le corpus du Greco communication parlée, "*La bise et le soleil*", sont les suivants :

Classe	nombre total	trouvées	insérées	% trouvées	% insérées
Plosives	159	157	18	98	11
Fricatives	138	132	1	95	1
Voyelles	504	490	14	97	3

Il est à noter que les trois algorithmes sont appliqués de façon indépendante, et que les nombreuses insertions relatives aux segments plosifs ont deux explications :

- Elles correspondent tout simplement à des pauses entre deux mots, et n'affecteront pas la recherche d'un seul mot.
- Elles correspondent à un double étiquetage d'un son /*f* / ou /*v* / peu énergétique, mais ce segment reçoit souvent le double label plosif et fricatif. Loin d'être une erreur, cela constitue une information supplémentaire, son identification étant quasi évidente après cette remarque. Certains /*v* / n'ont que l'étiquette "*plosif*" (et voisé), et seront étiquetés /*b* /. Toutefois, ce phénomène est classique, certains phonéticiens parlent même de la classe {/*b* /, /*v* /}. Il suffira d'en tenir compte, en permettant par exemple une double description en terme de patron pour des mots contenant des /*v* /, et que l'on juge important de reconnaître a priori pour une application donnée.

Si l'on suppose qu'un mot est entièrement bien segmenté pour les trois classes de phonèmes qui nous préoccupent, on peut appliquer un algorithme de recherche

dont le coût sera proportionnel au nombre de segments trouvés dans le signal en entrée. L'idée mise en œuvre est simple :
A chaque mot correspond un patron phonétique, qui se réduit à la liste des grandes classes phonétiques qui le composent, parmi plosives, fricatives et voyelles.

Exemple :
Ainsi, les mots "*chapeau*" et "*chacun*" auront le même patron soit :
"F V P V"

Un affinage consisterait à augmenter le nombre de classes de phonèmes pouvant apparaître dans ces patrons, en prenant en compte des traits phonétiques supplémentaires, tels que le voisement des consonnes et la nasalisation des voyelles, à condition qu'ils soient obtenus avec des degrés de confiance suffisants, c'est-à-dire comparables aux autres résultats que nous exposons ci-dessus. Le patron serait alors plus sélectif, en particulier "*chapeau*" et "*chacun*" seraient différenciés, mais une seule erreur sur ces indices empêche le repérage d'un mot présent. Dans un premier temps, nous avons donc choisi d'implanter la version la plus simple, avec les trois classes que nous avons étudiées ci-dessus.

Le patron du mot est alors équivalent à une écriture d'un nombre en base n, où n est le nombre de classes retenues. "*chapeau*" peut donc être associé à 1202_3 si l'on code le caractère plosif par 0, le caractère fricatif par 1 et les noyaux vocaliques par 2. Repérer "*chapeau*" sur un signal correctement segmenté revient alors à rechercher ce nombre sur quatre positions contiguës. Pour cela, il suffit de promener un masque de largeur quatre sur la segmentation fournie par le traitement, après transformation des labels retenus selon le codage désiré et en élimination des segments non pris en compte :
(/*r*/, /*l*/, /*m*/, /*n*/,/ *ɲ* /, /*w*/ ....).

Chaque valeur associée au masque courant se déduit de la valeur précédente en éliminant la contribution du nombre sortant à gauche (si l'on a choisi d'opérer de gauche à droite) et en ajoutant celle du nombre entrant à droite. On peut ainsi repérer tous les mots du lexique de longueur L et L – 1 en un balayage. Lorsque le nombre correspondant au masque est une entrée du lexique des patrons, on peut vérifier tous les mots qui sont associés à ce patron. Ils sont en général peu nombreux, et une comparaison phonème par phonème est alors tout à fait possible. Par exemple, le patron 1202 ou "F V P V" ne correspond qu'à une dizaine de mots parmi plus de 1300 entrées [3], comme "*chapeau*", "*chaque*", "*chacun*", "*Jean-paul*", "*jeudi*", "*jusqu'à*", "*jusque*", "*Jacques*", ... La deuxième phase de compara-

[3] Ce lexique est celui utilisé par François Charpillet [CHARPILLET 85] pour la machine à dicter, et a été constitué par un grand nombre de correspondances (surtout des lettres) échangées dans le cadre du GRECO communication parlée

ison, phonème après phonème, va permettre d'en éliminer plus ou moins, selon leur proximité relative, en prenant en compte les traits supplémentaires trouvés pour chaque segment.

Ainsi, sur la phrase :

```
" Ils ont de beaux chapeaux tyroliens "
```

seuls "*chapeaux*" et "*Jean-paul*" ont été retenus, alors que le trait de voisement était erroné pour le phonème /*ch* /, et qu'une segmentation trop courte du /*o* / laissait supposer la présence d'un /*l* / à la fin de *chapeaux*.

5 Résultats obtenus

5.1 Repérage rapide de mots pouvant servir de points d'ancrage

1. **Résultats permis en théorie :**

   Calculons la probabilité de trouver un mot comportant L segments des classes retenues pour constituer son patron phonétique.

   Pour simplifier les calculs, nous ferons les hypothèses (que nous espérons pessimistes!) suivantes :

   (a) La probabilité de repérer correctement une fricative, une plosive ou une voyelle est supérieure ou égale à 95 %. Soit p cette probabilité.

   (b) La probabilité q d'insérer l'un de ces trois phonèmes à **l'intérieur** d'un mot, entre deux phonèmes contigüs est inférieure ou égale à 5 %.

   Pour trouver un mot ayant L constituants dans son patron de façon contiguë, il faut déjà trouver ces L constituants simultanément :
   Si $p = 0.95$ est la probabilité pour l'un d'entre eux, la probabilité P pour les repérer tous est alors $P = p^L$ soit 0.95^L.

   Pour L = 6, cela donne 73.5 % de cas favorables.

   Il faut ensuite qu'aucune insertion ne se produise entre ces L constituants, dans (L − 1) intervalles possibles.

   La probabilité de cet événement est $P' = (1 - p')^{L-1}$ soit pour p' = 5 % et L = 6 cela donne :

   P' = 77.4 %

La probabilité globale de repérage est donc :

PT = P * P' = 56% de mots présents détectés.

Pour des mots plus courts, la proportion augmente rapidement : 73 % des mots de longueur 4, 77 % des mots de longueur 3 peuvent être reconnus, mais bien sûr l'ambiguïté augmente, et le nombre de mots détectés à tord sera beaucoup plus important.

2. **Selon les premières estimations :**

Les résultats de cette recherche a priori sont très dépendants du locuteur, et de son rythme d'élocution. Sur un corpus pré-enregistré dans de bonnes conditions, les résultats sont proches de l'approximation précédente.

Dans un énoncé du type :

"Descends la pince de deux"

le repérage des quatre mots *descends, pince, de, deux*, n'est pas rare.

Sur des énoncés plus longs, correspondant à des phrases énonciatives du type :

"J'aimerais savoir a qui il faut s'adresser pour faire une demande de double nationalite"

les résultats chutent si l'élocution est peu soignée.

Des exemples de ce type de recherche sur un lexique important (en particulier les deux exemples abordés ci-dessus) sont fournis en annexe.

Une deuxième méthode pour une recherche ascendante a également été testée. L'idée de départ était la même : l'explosion combinatoire de la procédure de recherche est liée à la programmation dynamique, ou plutôt aux hypothèses d'insertions et de substitutions que l'on fait dès que le phonème attendu n'est pas présent. Pour supprimer ce facteur de branchement trois assez fâcheux, il suffit de s'appuyer sur des portions de mots toujours présentes. La grande difficulté est de déterminer l'unité à choisir, à la fois pour qu'elle reste sélective (identifiante de peu de mots) et stable (c'est-à-dire presque toujours détectée par le niveau de décodage acoustico-phonétique). Les syllabes et les demi-syllabes ne nous ont pas semblé parfaites , car à la fois trop grossières et non toujours détectées : par exemple, le mot "*vingt-et-un*" ne se verra pas systématiquement crédité de trois noyaux vocaliques et donc des trois syllabes *ving t-et -un*

En revanche, les résultats actuels des niveaux infra-lexicaux permettent d'affirmer sans trop de risque que la plosive (le *t*) et que probablement deux des noyaux vocaliques seront trouvés.

On remarque donc que l'invariant à chercher est du type :

paquet vocalique–plosive–paquet vocalique

Chaque mot sera donc découpé de cette manière, en macro-phonèmes. Les différentes classes possibles sont :

- NV pour les groupes vocaliques. Ce groupe peut : soit être réduit à une voyelle V, soit se composer d'une suite de voyelles SV, soit contenir des voyelles et des sonantes :
  NV → V—NV—SSnV
  Le cas SSnV est très utile pour prendre en compte les mauvais résultats de la segmentation de séquences comme */o/r/ə/* ou */o/l/ə/*

- SC pour des suites consonantiques regroupant des plosives P et des fricatives F. On distinguera des groupes ne contenant que des plosives GP (pas de fricatives, de burst ou de /r/ fricatifs possibles), et des groupes pouvant à la fois contenir des segments plosifs et fricatifs GPF.

  Cette classification résout des problèmes de sur-segmentation de parties occlusives, ou d'apparition de segments étiquetés comme plosifs dans des fricatives pas assez énergétiques.

Ce découpage est fait de façon automatique, en regroupant les phonèmes correspondant à la transcription standard du mot de façon contextuelle. Un outil générant les patrons a été développé à l'aide du logiciel lex d'Unix.

Ainsi, un mot comme "*exactitude* " recevra les étiquettes suivantes :

/ɛ/g/z/a/k/t/i/t/y/d/ə/ → NV GPF NV GPF NV GP GV GP GV

Éventuellement, un mot peut être associé à plusieurs patrons, si des variantes ou des altérations phonologiques sont permises. Par exemple, le mot *petit* pourra se prononcer :

1. /p/ə/t/i/ et sera transcrit GP NV GPF NV
2. /p/t/i/ et sera transcrit GPF NV

Le treillis fourni par le décodage subira la même transformation, et une recherche exacte est alors effectuée avec ces macro-phonèmes.

En cas de succès, une deuxième passe utilisant la description précise des

macro-phonèmes fournira un score de reconnaissance affiné, ou pourra même rejeter un mot retenu dans un premier temps.

Ce type de méthode résout un certain nombre de problèmes liés à la phonologie ou aux limites du système de façon élégante. Ainsi, les bursts fricatifs des "*t* " devant des voyelles comme "*i* " ou "*é* " seront parfaitement pris en compte par cette méthode, ou les élisions d'une plosive (deux plosives se regroupent en une) ou les doublements de plosives (un bruit interne a *découpé* une occlusion en deux parties) ne poseront pas de problèmes.

Par contre, l'élision d'un noyau vocalique entre deux consonnes (par exemple du "*i* " de "*saucisson, afficher* ") empêchera le repérage du mot. Un paliatif consiste à ajouter une deuxième description pour les mots de ce type s'il est primordial de les reconnaître a priori, en y incluant l'erreur prévisible de segmentation.

Cette deuxième approche, pourtant plus complexe, n'a pas fourni de résultats meilleurs que l'approche précédente.

5.2 Validation de mots

La grande difficulté dans la mise au point de ce module réside dans l'évaluation des seuils de rejet. Si ce seuil est trop faible, beaucoup trop de mots seront reconnus, et le nombre de chemins à examiner pour les analyseurs syntaxico-sémantiques va croître de façon trop importante.

En cas de seuil trop élevé, des mots présents seront rejetés. Cela est bien sûr catastrophique, car l'énoncé ne pourra pas être reconnu.

Les pages suivantes présentent trois exemples d'exécution d'une reconnaissance de phrase, avec un même signal de parole, donc le même treillis phonétique, mais avec trois seuils différents.

La phrase prononcée était :

DÉGAGE LA PILE BÉTA!

Un premier essai avec un seuil fixé à 75 % aboutit à une interprétation mauvaise du nom de la pile, le score du mot n'ayant pas d'influence pour l'instant dans le score d'interprétation. A égalité, la première phrase correcte est retenue, et cela donne :

DÉGAGE LA PILE DELTA (Pénalité : 33 %)

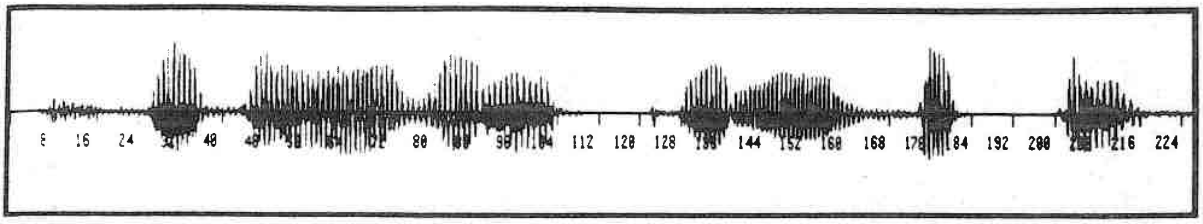
Le deuxième essai avec un seuil fixé à 80% est entièrement correct (toujours avec 33 % de pénalités). Un troisième essai avec cette fois un seuil de 85

% réduit encore les ambiguïtés, mais l'absence d'un mot exprimant la commande à effectuer conduit à un résultat pour le moins inattendu! En général, le système ne fournira pas d'interprétation si le seuil est trop élevé. Cet exemple n'est qu'un contre-exemple amusant... ou facheux pour le concepteur de tels systèmes. Une trop grande souplesse est donc elle aussi à prohiber (ici, le seuil de reconnaissance total pour l'ensemble de la phrase a été mis à un niveau assez faible, ce qui conduit à trouver une interprétation — ici avec 53 % de pénalités, moins d'un mot sur deux a été reconnu — le plus souvent possible, avec les inconvénients que cela induit. Le module lexical travaille de façon ascendante, et recherche les mots de façon aveugle : tous les mots sont susceptibles d'être présents partout).

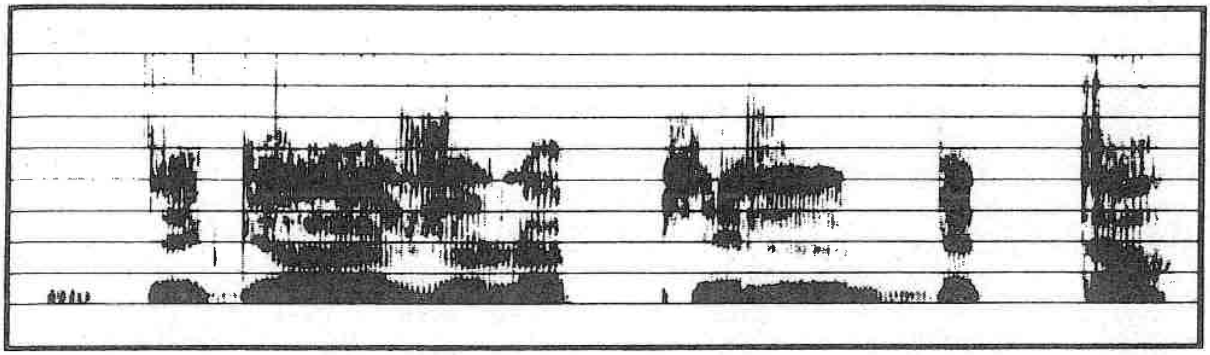
Dans les pages qui suivent, on voit alternativement des copies d'écrans correspondant aux exécutions des niveaux infra-lexicaux et de LEX d'une part (quatre fenêtres graphiques), puis ce que l'on peut voir sur la console où Partner s'exécute (texte). Les traits verticaux dans la fenêtre MOTS représentent les frontières temporelles de l'occurrence détectée. Des couleurs différentes précisant la plus ou moins bonne qualité du mot n'ont pû être reproduites, mais le score de reconnaissance est fourni à Partner, qui l'affiche. On pourra donc se reporter à la trace d'exécution qui suit pour retrouver les scores de mots. Ainsi, la ligne :

```
In [  2 -  7 ]  with  79  ->  [11]  deplace
```

doit être interprétée comme : Le mot *déplace*, d'identifiant 11, a été trouvé entre les phonèmes 2 et 7, avec une plausibilité de 0.79.



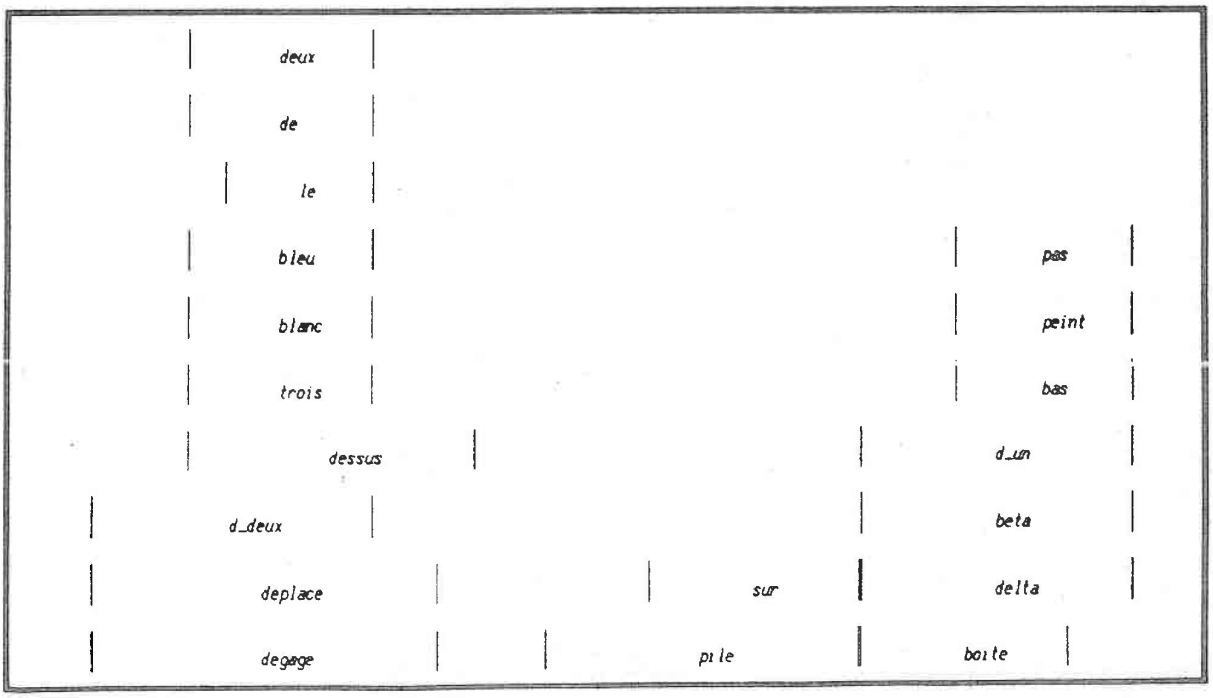
Signal brut



Spectrogramme

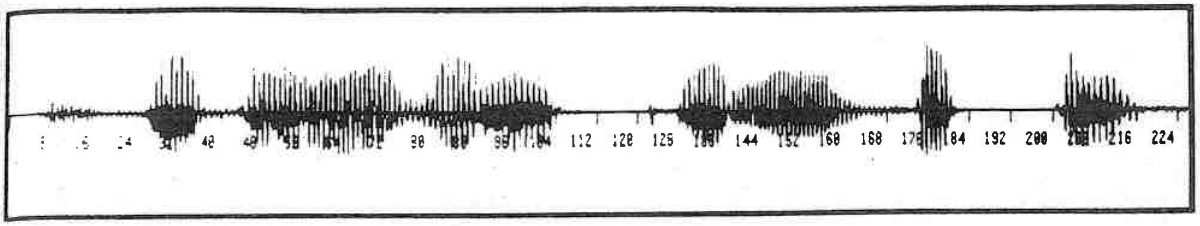
b		d	i	d	R	oe	gh	y	a	p	f	i	l	b	e	p	a	#
	y		e			a	v	i	oe		ch	y			ai		in	
v					l		z			k	s			v	y	k		

Decodage

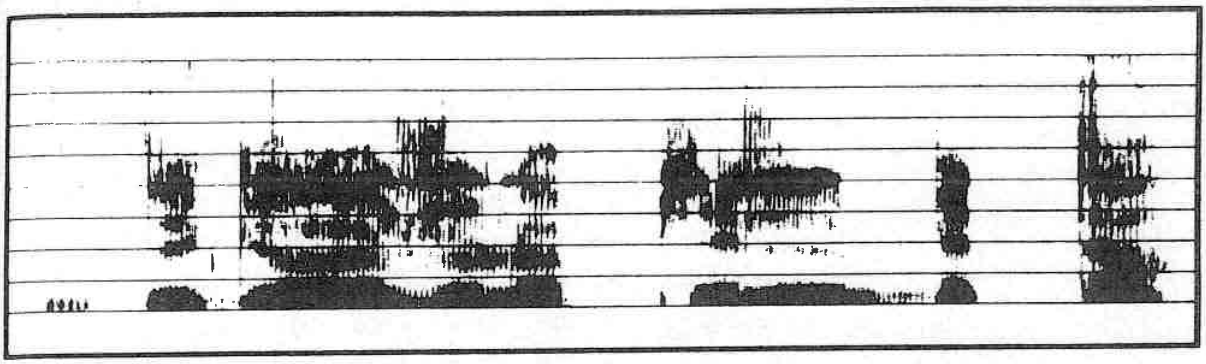


MOTS avec 75%

```
- Loading compiled lexicon 'pince.lx'...
- Loading compiled grammar 'pince.gr'...
- Connecting to SNORRI...
- Hearing speech signal...
- Making lattice...
Lattice :
 In [  2 -  7 ]  with  84  ->  [27]  degage
 In [  2 -  7 ]  with  79  ->  [11]  deplace
 In [ 14 - 16 ]  with  75  ->  [38]  boite
 In [ 14 - 17 ]  with  76  ->  [18]  delta
 In [  2 -  6 ]  with  92  ->  [46]  d_deux
 In [ 14 - 17 ]  with  96  ->  [16]  beta
 In [  4 -  8 ]  with  78  ->  [14]  dessus
 In [  4 -  6 ]  with  76  ->  [ 6]  trois
 In [ 14 - 17 ]  with  85  ->  [45]  d_un
 In [  4 -  6 ]  with  81  ->  [42]  blanc
 In [  4 -  6 ]  with  94  ->  [41]  bleu
 In [ 10 - 13 ]  with  80  ->  [13]  pile
 In [ 11 - 13 ]  with  85  ->  [12]  sur
 In [ 16 - 17 ]  with  92  ->  [49]  bas
 In [ 16 - 17 ]  with 100  ->  [39]  peint
 In [  5 -  6 ]  with 100  ->  [29]  le
 In [ 16 - 17 ]  with 100  ->  [21]  pas
 In [  4 -  6 ]  with  80  ->  [ 9]  de
 In [  4 -  6 ]  with  80  ->  [ 5]  deux
 In [  5 -  6 ]  with 100  ->  [ 2]  la
20 word(s) found.
- Building tas...
- Making tree...
Tree = degage [84] la [0] pile [80] delta [76]
Rate =  33
- Disconnecting from SNORRI...
I recognize : - Displaying tree...
Tree = degage [84] la [0] pile [80] delta [76]
Rate =  33
I understand : - Displaying semantic...
Semantic = [ACT=FREE[70](OBJ=PILE[63](PILENAME=DELTA[76]()))]
Rate =  33
```



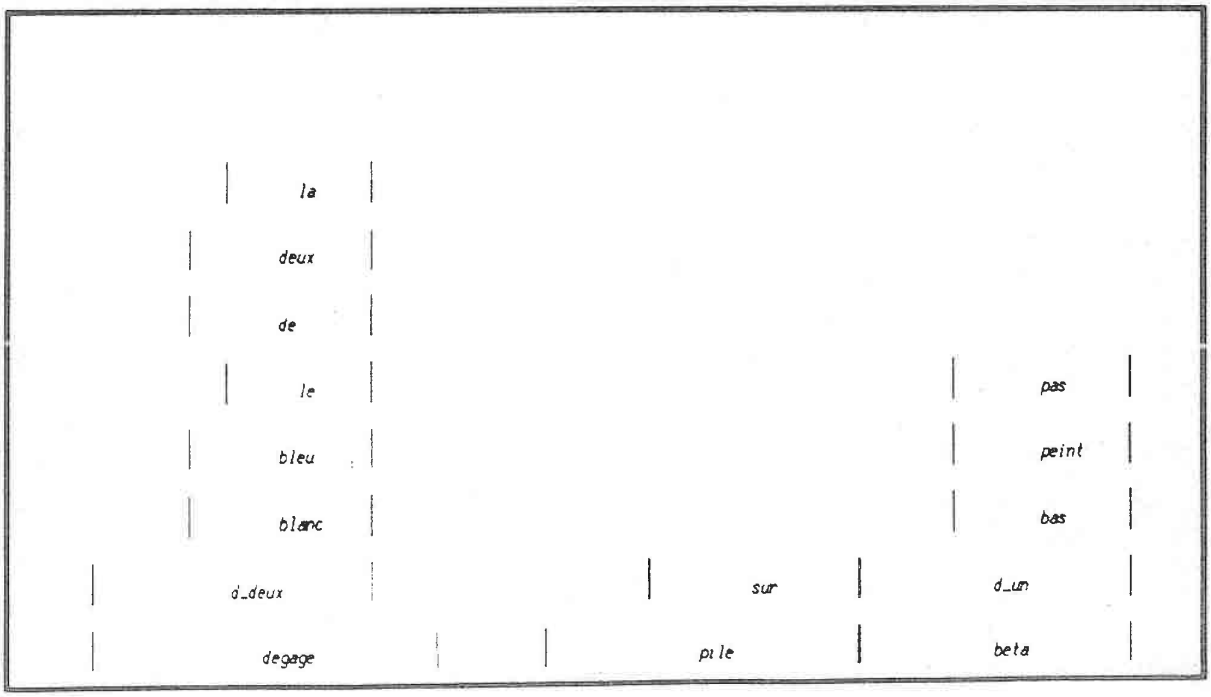
Signal brut



Spectrogramme

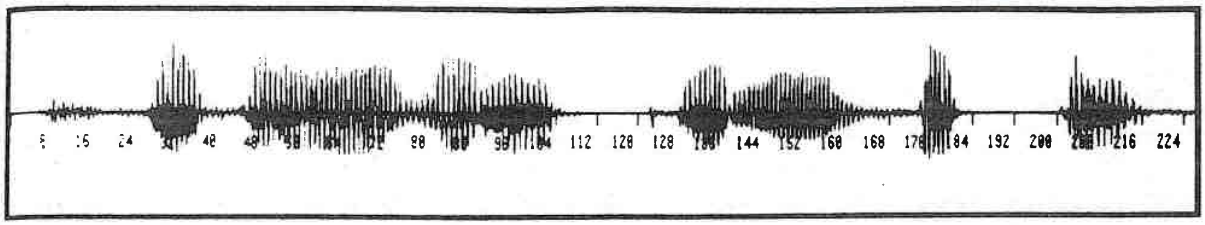
b	d	i	d	R	oe	gh	y	a	p	f	i	l	b	e	p	a	#
	y		e			a	v	i	oe		ch	y			ai		in	
v				l			z			k	s			v	y	k		

Decodage

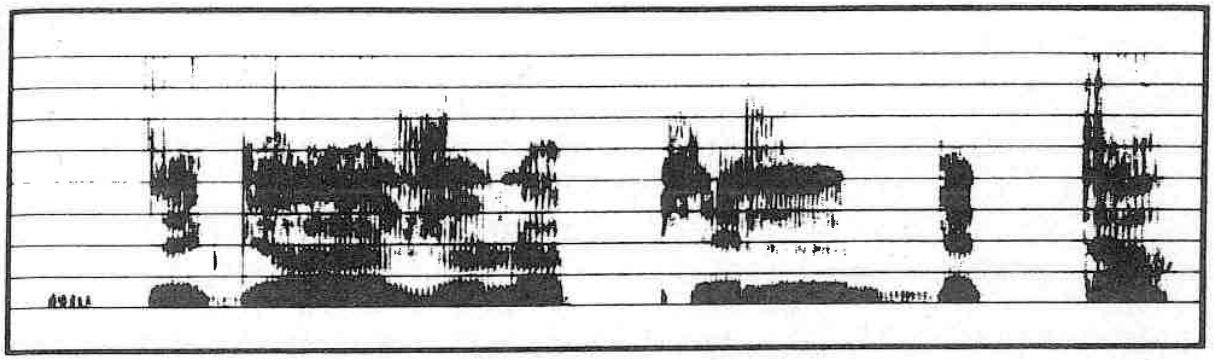


MOTS avec 80%

```
- Loading compiled lexicon 'pince.lx'...
- Loading compiled grammar 'pince.gr'...
- Connecting to SNORRI...
- Hearing speech signal...
- Making lattice...
Lattice :
 In [  2 -  7 ]  with  84  ->  [27]  degage
 In [  2 -  6 ]  with  92  ->  [46]  d_deux
 In [ 14 - 17 ]  with  96  ->  [16]  beta
 In [ 14 - 17 ]  with  85  ->  [45]  d_un
 In [  4 -  6 ]  with  81  ->  [42]  blanc
 In [  4 -  6 ]  with  94  ->  [41]  bleu
 In [ 10 - 13 ]  with  80  ->  [13]  pile
 In [ 11 - 13 ]  with  85  ->  [12]  sur
 In [ 16 - 17 ]  with  92  ->  [49]  bas
 In [ 16 - 17 ]  with 100  ->  [39]  peint
 In [  5 -  6 ]  with 100  ->  [29]  le
 In [ 16 - 17 ]  with 100  ->  [21]  pas
 In [  4 -  6 ]  with  80  ->  [ 9]  de
 In [  4 -  6 ]  with  80  ->  [ 5]  deux
 In [  5 -  6 ]  with 100  ->  [ 2]  la
15 word(s) found.
- Building tas...
- Making tree...
Tree =  degage [84]  la [0]  pile [80]  beta [96]
Rate =  33
- Making tree...
- Disconnecting from SNORRI...
I recognize : - Displaying tree...
Tree =  degage [84]  la [0]  pile [80]  beta [96]
Rate =  33
I understand : - Displaying semantic...
Semantic = [ACT=FREE[75](OBJ=PILE[70](PILENAME=BETA[96]()))]
Rate =  33
```



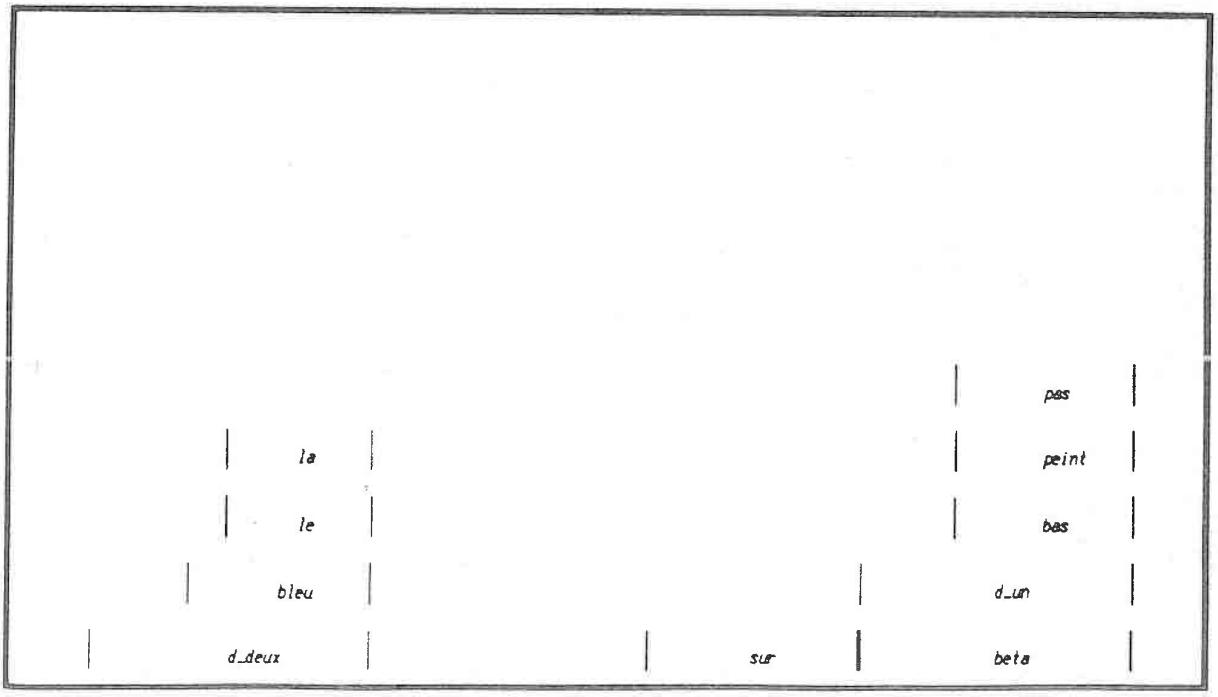
Signal brut



Spectrogramme

b		d	i	d	R	oe	gh	y	a	p	f	i	l	b	e	p	a	#
	y		e			a	v	i	oe		ch	y			ai		in	
v					l		z			k	s			v	y	k		

Decodage



MOTS avec 85%

```
- Loading compiled lexicon 'pince.lx'...
- Loading compiled grammar 'pince.gr'...
- Connecting to SNORRI...
- Hearing speech signal...
- Making lattice...
Lattice :
 In [  2 -  6 ]  with  92  ->  [46]  d_deux
 In [ 14 - 17 ]  with  96  ->  [16]  beta
 In [ 14 - 17 ]  with  85  ->  [45]  d_un
 In [  4 -  6 ]  with  94  ->  [41]  bleu
 In [ 11 - 13 ]  with  85  ->  [12]  sur
 In [ 16 - 17 ]  with  92  ->  [49]  bas
 In [ 16 - 17 ]  with 100  ->  [39]  peint
 In [  5 -  6 ]  with 100  ->  [29]  le
 In [ 16 - 17 ]  with 100  ->  [21]  pas
 In [  5 -  6 ]  with 100  ->  [ 2]  la
10 word(s) found.
- Building tas...
- Making tree...
Tree = prend [0] le [100] cube [0] sur [85] beta [96]
Rate =  59
- Making tree...
Tree = peint [0] le [100] cube [0] sur [85] beta [96]
Rate =  57
- Making tree...
- Disconnecting from SNORRI...
I recognize : - Displaying tree...
Tree = peint [0] le [100] cube [0] sur [85] beta [96]
Rate =  57
I understand : - Displaying semantic...
Semantic = [ACT=PAINT[46](OBJ=CUBE[64](LIEU=ON[91](OBJ=PILE[91]
           (PILENAME=BETA[96]())))))]
Rate =  57
```

Chapitre 5

Perspectives d'avenir

1 Autres niveaux

1.1 Fonctionnalités

La deuxième partie importante d'une composante lexicale est son interface avec les niveaux supra-lexicaux. Cette composante jouera un double rôle vis-à-vis des analyseurs syntaxico-sémantiques. Ces analyseurs ont besoin de sélectionner à chaque pas des sous-lexiques, correspondant à des traits syntaxiques ou sémantiques. Il faut donc être capable à tout moment de fournir l'information attendue. Lorsque les analyseurs s'appuient sur une grammaire sémantique, le lexique devra connaître l'ensemble des traits sémantiques utilisés, et connaître pour chacun des mots qui le composent la valeur de ce trait. Chaque trait associé à une valeur (le plus souvent booléenne) détermine un partitionnement des mots du lexique. L'ensemble des traits possibles, associés aux différentes valeurs que ces traits peuvent prendre, va partitionner le lexique en pseudo-classes d'équivalence.

Le seul point intéressant à ce niveau est de fournir la liste des entrées lexicales appartenant à la pseudo-classe sélectionnée, et l'information à fournir se limite à une clef d'accès. Seule une partie de l'information est utile à ce niveau, il est inutile de donner une description complète de chaque entrée lexicale, il suffit de permettre à tous les processus qui vont avoir à manipuler ce mot de retrouver les informations dont ils ont besoin pour leurs propres traitements, à un moment donné.

Il sera inutile également pour un processus donné de stocker l'information statique se rapportant à un mot. S'il a besoin de connaître le genre d'un nom, il suffit qu'il interroge la partie lexicale chargée de gérer ces informations, et de prendre en compte le résultat. Si cette information est appelée à resservir

ultérieurement, la même interrogation fournira le même résultat, il est donc inutile de dupliquer cette information dans le système.

Par contre, lorsqu'un mot a reçu un apport d'information dynamique, par exemple à la suite d'une comparaison avec le treillis phonétique, il est nécessaire de stocker cette information. De même, lorsqu'un trait sémantique a déjà été utilisé pour renforcer les scores des mots qui le possèdent, une deuxième apparition de ce trait ne doit plus provoquer de modifications dans le sous-lexique concerné.

Par contre, une remise en cause de ce trait devra annuler les modifications que sa prise en compte avait entraînées. La fonction de modification doit être inversible.

En résumé, cette partie du lexique devra remplir deux contraintes :

(a) Pour fournir les informations relatives à un mot, l'accès doit être rapide et complet

(b) Pour modifier de façon efficace et prendre en compte des informations de tous horizons, il faut être capable de faire ressortir un sous-lexique correspondant à une information, ou à un groupe d'informations.

Les modèles purement théoriques que l'on est tenté de construire se heurtent rapidement à la très grande diversité des informations à traiter, tant par leur origine que par leur contenu.

Nous avons donc choisi une représentation a priori, qui permet de mettre en oeuvre le maximum de fonctionnalités. Ce modèle permet d'une part d'implanter une hiérarchie en relation avec les traits syntaxiques et sémantiques que nous utilisons, et d'autre part de gérer efficacement les scores.

1.2 Représentation

Outils actuels : les langages à objets

Le nombre de langages et d'outils offerts aux programmeurs et aux concepteurs de systèmes ne cesse d'augmenter. Une nouvelle famille de langages, les langages à objets [COLNET 86], issus de concepts nés il y a presque vingt ans, commencent à être opérationnels. Ces langages permettent une grande modularité, où chaque objet informatique incorpore des aspects statiques et dynamiques. Un programme est alors un ensemble d'objets, chacun de ces

objets étant caractérisé par les opérations qu'il connaît. C'est donc une sorte de boîte qui fonctionne de façon indépendante. Les échanges avec le monde extérieur se font à l'aide de messages, qui provoquent des appels fonctionnels, exécutés dans l'environnement de l'objet concerné.

Chaque objet possède sa propre zone mémoire, que lui seul est capable de modifier. Sa définition initiale et son comportement ultérieur vont dépendre de liens d'héritages qu'il possède avec d'autres objets, ou avec des éléments génériques, appelés classes. Ces classes peuvent être structurées, et la hiérarchie va servir à propager les caractéristiques dont chaque objet hérite.

Un objet va interpréter un message reçu en exécutant une *méthode*, définie au préalable, soit au niveau même de cet objet, soit au niveau de l'un de ses ancêtres, si cet objet hérite à la fois de la *méthode* et de l'environnement définis au niveau de cet ancêtre. En particulier, toutes les variables référencées par cette *méthode* doivent exister dans l'univers de cet objet.

L'intérêt de ce type d'approche réside dans la possibilité d'éxécuter de façon simultanée des fonctions sur un ensemble d'objets. Ainsi, si l'on choisit de représenter chaque mot du lexique comme un objet particulier, on pourra définir des *méthodes* d'accès à ces mots correspondant à chaque hiérarchie des traits qui le composent. Lorsqu'un noeud reçoit une information (par exemple un trait est validé), il transmet cette information à tous ses ancêtres ou à tous ses descendants. Un mécanisme de marquage est nécessaire pour empêcher toute nouvelle réception d'un message déjà traité, si deux chemins différents aboutissent à une même entité.

Choix d'implémentations

Laurent Romary a implémenté cette partie du lexique dans le cadre de son DEA. Un gestionnaire d'hypothèses a donc été développé sur un lexique à hiérarchies multiples, en utilisant les possibilités d'héritages multiples permises par certains langages à objets. Le langage choisi pour cette réalisation est un sur-langage de lisp, développé au MIT, les Flavors [MOON 86]. Dans ce langage, les objets sont représentés sous forme de classes définies par un certain nombre de variables d'instance, et par la liste des sur-classes dont devra hériter cette classe. Les variables d'instance servent à décrire l'environnement local des objets attachés.

Structure du lexique syntaxico-sémantique

Chaque trait syntaxique ou sémantique du lexique est mis de façon naturelle sous forme de Flavors. Chaque noeud d'une arborescence correspond à une classe, donc aussi à un Flavor. Chaque mot du lexique correspond lui aussi à une classe (réduite à un seul élément). Une classe est donc générée par entrée lexicale. La définition du lexique est faite avant la mise en route du système, et nous n'envisageons pas de permettre à un utilisateur de définir un nouveau mot de façon dynamique.

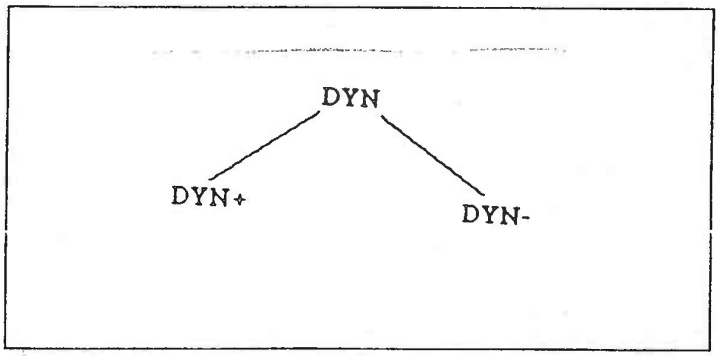
exemple de lexique

Nous utilisons ici le modèle fourni par Guy Deville et Hans Paulussen que nous avons détaillé au chapitre III [DEVILLE 87b].

Par exemple, le mot "*signer* " ayant pour primitive ACTOBJ (acte objet) caractérisée par la liste de traits suivantes :

```
(ctr+ dyn+ ani+ mvt- trs+ )
```

Chaque trait de base est représenté au niveau du lexique par un morceau d'arbre binaire :



Cette représentation n'est pas idéale, car il est impossible d'utiliser directement des arcs valués (inconnu, présent, absent). Seuls des traits ayant une valeur (+ ou -) peuvent être utilisés.

Chaque primitive hérite donc directement des noeuds correspondant à ses traits de base. Le sous-lexique correspondant à une primitive pourra éventuellement être structuré en sous-classes sémantiques.

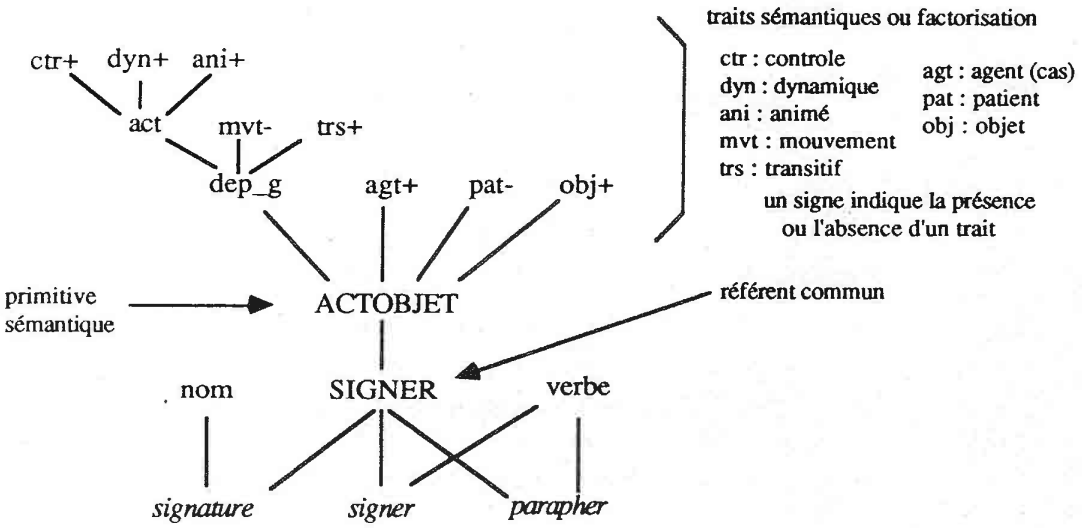


Figure 5.1: Exemple de factorisation d'une action sémantique

Dans la figure 5.1, une factorisation d'une même action sémantique a été faite, cette action pouvant s'exprimer par plusieurs terminaux de la grammaire, ayant un profil identique, la seule différence résidant dans leurs transcriptions orthographique et phonétique.

Les classes syntaxiques habituelles (nom, verbe, déterminant, adjectif, ...) sont aussi représentées sour forme de Flavor dans l'arborescence, et constituent donc un accès aux sous-lexiques correspondants.

Les primitives verbales ne se rapportent pas forcément uniquement à des verbes, car il existe des formes nominales qui traduisent la même action sémantique. Un noeud sémantique est alors créé, sur lequel on attache tous les mots correspondants, avec en plus des liens avec les noeuds représentant leurs catégories syntaxiques. Un exemple nous est founi par la figure 5.2

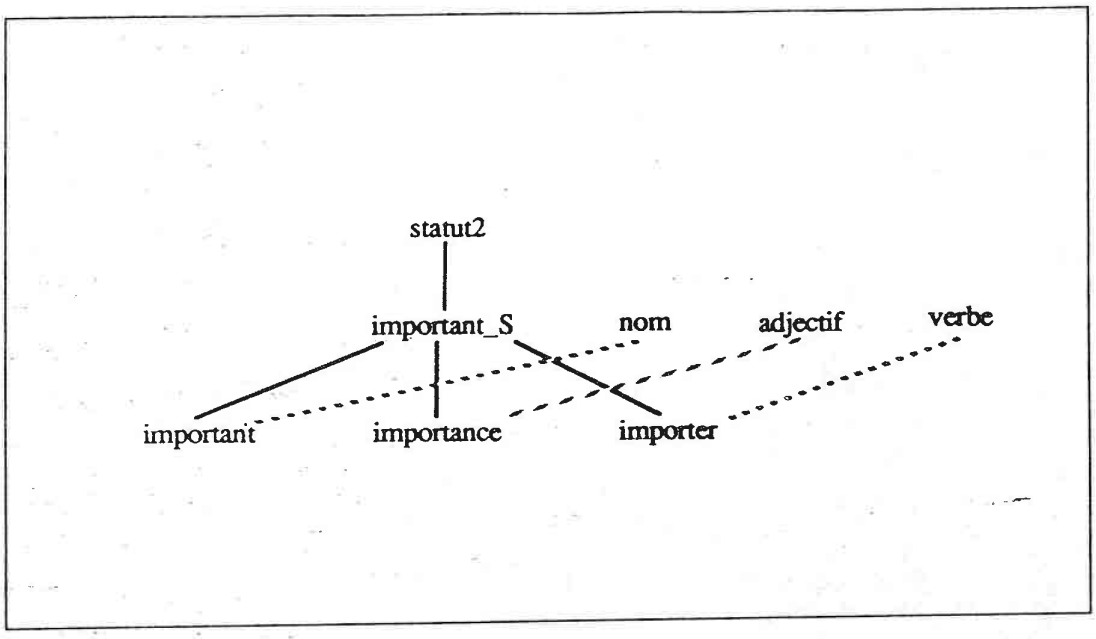


Figure 5.2: Exemple de sous-lexique traduisant "l'importance"

1.3 Gestion des hypothèses lexicales

L'accès à un sous-lexique se fait par un marqueur positionné au niveau de chaque élément. Une *méthode* particulière **Propage_marqueur** peut être lancée au niveau du noeud correspondant au sous-lexique à sélectionner et marquer, génération par génération, tous les descendants de ce noeud. Tout sous-ensemble élémentaire peur être ainsi sélectionné directement.

Pour un sous-ensemble correspondant à plusieurs traits, il suffit d'utiliser des *méthodes* **réunion, intersection et complémentation** qui vont travailler à partir des marqueurs précédents.

1.4 Fonctionnement

La première chose à faire est d'initialiser l'environnement. Une première *méthode* initialise tous les scores, et chaque feuille qui reçoit ce message d'initialisation renvoie en résultat son nombre d'occurences, ce qui permet de calculer la somme totale au niveau du lexique. Chaque noeud reçoit ensuite un message qui lui permet de calculer son propre score, comme étant le rapport du nombre d'occurrences sur le nombre total si ce noeud est une feuille, ou comme la somme des scores de ses fils si c'est un noeud interne.

Considérons maintenant un apport d'information : une première information a déjà mis un sous lexique X avec un score p.

On a donc au départ :

m(X) = p et m(– X) = 1 - p

Une nouvelle information concenant ce même sous-lexique X arrive avec un score q :

m'(X) = q et m'(– X) = 1 - q

La relation de Dempster [ROMARY 87] nous permet de calculer les nouveaux scores de X et –X

On obtient ainsi après combinaison :

m"(X) = p/(1-q - p*q) et m"(–X) = (1-p)*(1-q)/(1-p - p*q)

Si s_i était le score initial d'un élément particulier, on a donc la relation :

(s_i/p)*p/(1-q - p*q) = s_i/(1-q -p*q)

Il suffit donc d'appliquer à chaque élément du sous-lexique considéré un coefficient correctif égal à (1 - q)/(1 - p - p*q).

Ce mode de gestion des hypothèses lexicales a été programmé par Laurent Romary dans le cadre de son DEA, et donne des résultats très encourageants.

2 Mise en oeuvre des liaisons avec les autres processeurs :

2.1 Outils d'Unix

Nous avons déjà présenté le coté modulaire de notre système dans le chapitre II de ce manuscrit. Chacun des modules du système correspondra à un programme distinct, ou à un ensemble de programmes, et pourra être lancé de façon séparée. Une demande d'information ou une réponse à une question posée à une autre partie du système sera vue comme une entrée–sortie un peu particulière. A moyen terme, il est prévu que des versions du système soient implémentées de façon répartie, c'est-à-dire sur des machines différentes, soit d'un même constructeur, soit de plusieurs constructeurs, avec des systèmes d'exploitation différents. Tous les modules fonctionnant actuellement, ou en cours de développement, sont écrits en langage C, sous différentes variantes du système Unix. Lorsque nous avons interconnecté les premiers modules opérationnels, nous avons bien sûr tenu compte de cette spécificité, et nous

avons utilisé les outils offerts dans les versions d'Unix les plus récentes, issues des milieux universitaires de Berkeley.

Plusieurs possibilités sont offertes pour échanger des informations entre processus. Nous distinguerons deux cas :

(a) Les deux processus s'exécutent sur la même machine.

(b) Les deux processus s'exécutent sur deux machines différentes, interconnectées, par exemple dans un réseau local.

Outils permettant des échanges entre deux processus frères

Le problème est un peu plus simple lorsque les deux modules désirant échanger des informations s'exécutent en parallèle, sur un même unité centrale. Le système Unix fournit un mécanisme de communication appelé *pipe* , qui modélise un flux d'entrées–sorties géré selon la technique FIFO (First in, First out ou premier entré, premier sorti) , les deux sens de communication correspondant à deux files distinctes. La gestion des files en termes d'accès en lecture/écriture peut être faite soit par l'utilisateur, soit par le système qui utilise alors ses procédures de gestion standard d'entrée-sortie.

Le "*pipe* " sera vu comme un double fichier par les processus qui l'utilisent. L'un des descripteurs permettra d'ouvrir cette ressource en écriture, et le second en lecture. Les accès en lecture et écriture se feront par les fonctions standard d'entrée–sortie du système, par des ordres write et read. Toutefois, la taille de l'information *bufferisée*, —c'est-à-dire écrite par les processus jouant le rôle de producteurs—, ne pourra dépasser 4096 octets, sinon, l'écriture sera bloquante.

Pour la lecture, selon le système utilisé et l'option choisie par l'utilisateur, une demande d'informations sur un pipe vide pourra soit se faire sans délai, soit être bloquante si aucune information n'est présente.

L'usage par défaut banalise la fonction de lecture comme une entrée-sortie standard, et une demande de lecture est bloquante pour le processus demandeur jusqu'à l'arrivée de la valeur attendue. Toutefois, des mécanismes de time-out sont prévus, mais c'est à l'usager de les gérer.

Dans l'option NO_DELAY, c'est l'usager qui doit gérer lui-même les interruptions systèmes et les signaux entre les processus partageant un "*pipe* " pour utiliser correctement celui-ci. Une lecture lui renverra toujours le buffer tête de liste, mais il devra tester la présence d'information avant d'en faire

l'usage : il se peut qu'aucune information n'ait été écrite sur le support lu. Cette option est très pénalisante pour les autres usagers dans le cadre d'un usage multi-utilisateurs de la machine support, car les processus coopérants de cette façon doivent être résidents en mémoire centrale. Leur taille importante prend la plus grande partie de celle-ci, provoquant ainsi un swapping important si d'autres processus sont actifs. Le comportement du système devient très bizarre si la taille totale demandée dépasse ses propres ressources, et des blocages du système liés à une étreinte fatale peuvent se produire (deux processus essaient de se bloquer simultanément en mémoire, mais ne peuvent plus acquérir la partie de mémoire complémentaire nécessaire à leur chargement, celle-ci appartenant à un autre module devant être résidant lui aussi). L'utilisateur devra donc veiller à la taille de ses programmes exécutables avant de les rendre résidents, et segmenter suffisamment finement ses modules pour ne bloquer en mémoire que les procédures d'interfaces entre les diverses composantes du système.

Une deuxième option moins contraignante banalise complètement le "*pipe* " comme deux fichiers standards d'entrées–sorties, et la lecture et l'écriture se feront exactement comme une banale lecture clavier ou une écriture écran. Le processus demandeur sera alors placé en attente d'entrées–sorties, et sera bloqué dans cet état si l'information n'est pas disponible. Cette approche est bien sûr nettement plus lente à l'éxécution, — le processus perd la main, et peut attendre quelques instants avant d'être réactivé par le système — mais ne pénalise pas les autres usagers. Comme le but principal de nos recherches n'est pas d'implémenter à tout prix un système temps réel, nous avons choisi de travailler de cette façon pour nos tests. L'aspect le plus désagréable réside dans le blocage lorsque le processus distant n'est pas en mesure d'écrire l'information attendue.

Dans les versions de systèmes les plus récentes, et interfacées avec le réseau ETHERNET, les "*pipe* " sont implémentées à l'aide des "*sockets* " servant à l'interconnexion des hôtes d'un réseau hétérogène. Un mécanisme de "*look_up* " permet de passer l'information à transmettre au processus frère, en changeant simplement le bit indiquant le statut du message (a_envoyer → reçu et le nom du propriétaire de ce message) , au lieu de le poster sur le cable. Nous allons étudier le cas d'un système réparti dans le paragraphe suivant. Chaque module pourra alors être implémenté sur une machine distincte d'un réseau local, et le parallèle permis lors de la conception de nos systèmes pourra alors être mis en oeuvre de façon réelle.

Outils permettant des échanges dans un réseau local

Une des plus grandes nouveautés des systèmes Unix Berkeley postérieurs à la version 4.2 est d'avoir introduit de nouveaux concepts pour la communication entre processus (IPC pour InterProcessus Communication). Le grand défaut des "*pipes* " que nous venons de décrire était la restriction à des communications entre processus s'éxécutant sur une même machine. L'intérêt du découpage du système en modules indépendants diminue alors légèrement, car chacun de ces modules est géré par le système hôte, et le parallélisme n'est plus absolu. Comme nous ne disposons que d'un seul processeur central sur notre Masscomp, chaque module actif ne pourra s'exécuter que lorsque le système lui donnera la main, c'est-à-dire quand les processus frères seront stoppés.

Les "*sockets* " mis au point à Berkeley constituent une interface d'accès entre les programmes d'un utilisateur, —chacun de nos différents modules dans notre cas—, et la couche transport du réseau local hôte du système. L'interface fournie à l'utilisateur est indépendante des couches basses du système, en particulier du type de machine et du type de réseau utilisé pour relier les deux systèmes qui doivent communiquer.

Là encore, la description d'une "*socket* " est assimilée à une description de fichier, présentant des caractéristiques spéciales, et n'ayant pas d'existence physique sur une mémoire de masse.

Chaque "*socket* " possède donc un type, et un ou plusieurs processus associés. Elle appartient à un domaine de communication, ce domaine étant un concept permettant de regrouper des descripttions de propriétés communes à un certain nombre de processus succeptibles de communiquer via ce canal.

Par exemple, le nommage des "*sockets* " va dépendre du système hôte. Dans un environnement UNIX, un nom de "*socket* " sera un chemin de fichier classique de l'arborescence UNIX, comme */dev/toto*. Deux "*sockets* " devront être du même domaine de communication pour échanger des informations. Deux machines UNIX pourront donc échanger des informations par ce canal sans problème.

Les "*sockets* " sont également typées suivant les propriétés de communications voulues par l'utilisateur. Trois possibilités lui sont offertes :

(a) Stream "*socket* " :

Ce premier type de "*socket* " est utilisé pour des flots de données bidirectionnels, fiables, séquencés : l'ordre d'émission et de reception est le même, deux données ne peuvent pas se *doubler* pendant le transfert. Deux streams "*sockets* " connectés donne une interface ayant la même spécification que deux "*pipes* " bidirectionnels, et d'ailleurs c'est ce mécanisme qui a été retenu pour implantés les "*pipes* " dans les versions d'UNIX actuelles. Les Stream "*sockets* " permettent donc de créer une interface de type producteur–consommateur entre deux processus se déroulant sur deux machines UNIX interconnectées par un réseau local de type ETHERNET, comme celui du CRIN.

(b) "*sockets* " datagram : Les "*sockets* " datagram supportent des flots de données sans assurance de séquencement (intuitivement un message peut en *doubler* un autre pendant le transfert, et le recepteur peut recevoir les données dans un ordre différent de celui dans lequel elles avaient été envoyées). Elles permettent une communication au coup par coup, et vont servir à règler des problèmes ponctuels : demande d'une information unique, mais aussi établissement d'une liaison virtuelle avec des stream "*sockets* " pour une série de demandes d'un même type, par exemple vérification d'une série de mots.

(c) Les raw "*sockets* " fournissent des accès aux protocoles de niveau bas (pour les machines du CRIN ce protocole est IP, le protocole de l'Internet). Ces "*sockets* " sont similaires aux "*sockets* " datagram, mais leurs caractéristiques vont dépendre de l'interface fournie par le protocole de bas niveau. Elles permettent d'écrire de nouveaux protocoles de communication, adaptés aux besoins de l'usager.

Des outils pour permettre la création, la mise en corespondance, la connexion, le transfert de données et une fin de communication *propre* sont mis à la disposition de l'utilisateur pour interconnecter deux ou plusieurs processus. Nous allons illustrer ces diverses possibilités dans le cadre de notre travail :

(a) Création de "*sockets* " :

Une fonction C permet de créer une socket :

s = *socket*(domain, type, protocol);

int s, domain, protocol;

avec

s	descripteur du socket
domain	numéro du domaine du socket
type	numéro de type du socket
protocol	numéro du protocole utilisé

Bien sûr, des constantes sont définies dans divers fichiers comme

`<sys/socket.h> , <sys/un.h> , <netinet/in.h>`

qui devront être inclus dans les programmes créant des "*sockets* ".

Les numéros de protocoles connus du système sont définis dans le fichier /etc/protocols, et des fonctions prédéfinies permettent un accès à ces informations.

(b) Liaison de noms :

Lorsqu'une "*socket* " est créée, elle n'a pas de nom, et les processus n'ont alors aucun moyen de la référencer. Aucun message ne peut donc être transmis par ce canal : pour appeler quelqu'un par téléphone, vous devez connaître son numéro, identifiant de sa ligne.

Une fonction permet de repérer une "*socket* " en l'associant à une adresse locale. Les paquets d'information portant cette adresse seront alors déposés dans la "*socket* " ainsi repérée.

bind(s, nom, longueur)

va associer la "*socket* " de numéro de descripteur s à l'adresse nom.

s	descripteur du socket
nom	adresse locale
longueur	longueur de cette adresse locale

(c) Établissement d'une connexion :

Lorsqu'une "*socket* " a été associée à une adresse, il est alors possible d'avoir un rendez-vous avec un autre processus. A priori, cette opération est asymétrique, l'un des processus étant client, l'autre serveur.

Un serveur est un processus qui s'exécute en permanence, et qui scrute à intervalles réguliers un port particulier. Dès que le serveur est lancé, le plus souvent à la fin du boot du système, il autorise un certain nombre de connexions en provenance de modules distants sur l'adresse de transport qui lui est reservée en lançant la fonction :

```
listen(s,5);
```

Cette fonction autorise les connexions distantes à la "*socket* " associée à cette adresse. Le processus distant devra donc faire un *connect* à cette adresse :

```
connect(s, &server, sizeof(server))
```

Bien sûr, chaque serveur dispose d'une adresse privée, qui lui est réservée par le système. Le deuxième paramètre spécifie le nombre maximum de requêtes pouvant être en attente d'acceptation par le serveur.

Lorsqu'un processus distant se connecte à une "*socket* " , le processus local doit accepter la communication :

```
svrai = accept(s, &from, &fromlen)
```

où from est le nom du client, fromlen la longueur de ce nom.

svrai sera un nouveau descripteur, qui servira à gérer les échanges avec le client qui a effectué le *connect*. cet appel est bloquant jusqu'à l'établissement d'une liaison. Dès que celle-ci est établie, la main est rendue au processus qui peut éventuellement accepter d'autres connexions.

(d) échanges de messages

Lorsqu'une connexion est établie, deux types de fonctions système permettent d'échanger des informations :

i. write et read

Un tampon contenant un message ou devant recevoir un message est nécessaire.

Exemple :

```
char tampon[TAILLETAMPON];
write(s, tampon, sizeof(tampon));
read (s, tampon, sizeof(tampon));
```

ii. send et receive

Ces deux fonctions sont identiques aux deux précédentes, seul un paramètre supplémentaire permet une gestion plus souple du canal de communication :

```
send(s, tampon, sizeof(tampon), flags)
recv(s, tampon, sizeof(tampon), flags)
```

flags peut prendre les valeurs suivantes :

MSG_PEEK : permet de voir si des données sont arrivées, sans être bloqué dans le cas contraire. Cette option est bien sûr très intéressante pour implanter un système permettant un déroulement parallèle de ses différentes composantes.

MSG_OOB : permet d'utiliser des données express (pour l'envoi et pour la reception). Ceci n'est utile que dans le cadre d'un pro-

tocole qui gère ce type de données, et ne présente pas d'intérêt dans le cadre de ce travail.

(e) déconnexion

Lorsqu'une "*socket* " ne sert plus, on peut la fermer de façon brutale :

close(s);

Pour détruire une socket,on utilisera :

unlink();

Si des données étaient en cours de transfert, le système va détecter une anomalie, et va continuer à essayer d'effectuer correctement ce transfert pendant plusieurs minutes. Ceci peut-être évité en utilisant :

shutdown(s,how)

how est un paramètre qui précise de quelle façon la "*socket* " va cesser de travailler :

en lecture.

en écriture.

à la fois en lecture et en écriture.

Lorsqu'un l'un des modules (client ou serveur) se crashe, la "*socket* " correspondante reste ouverte sur l'autre machine. Une lecture renverra le signal EOF, et une écriture provoquera un signal SIGPIPE.

(f) Exemple d'utilisation :

L'un des modules devra comporter une procédure de type *serveur*, et l'autre devra jouer le rôle de *client*.

La structure générale de la procédure à inclure dans le module *serveur* est donc la suivante :

```
pserv()
{
 s = socket (AF_INET, SOCK_STREAM, 0);
 bind(s, &so_in, sizeof(so_in));
 listen (s,2);


/* ici on autorise deux modules distants à dialoguer simultanément via
ce canal */


 for(;;)
```

```
    {
     int g, long=sizeof(from);
     if ( (g=accept(s, &from, &long)) < 0 )  continue;
     /* On boucle sur ce test en attendant un appel distant
 */

     if (fork() == 0)
        {
        close(s);

       /*****************************************************/
       /*inserer ici tous les ordres de transfert souhaites*/
       /*...................................................*/
       /*****************************************************/


        /* fin de la communication */
        close(g);
        exit(0);
        }
     close(g);
    }
}
```

Une procédure jouant un rôle symétrique sera à installer dans le module *client*. Sa structure générale sera la suivante :

```
pcli()
{
 struct sockaddr_in so_in, from;
 s = socket(AF_INET, SOCK_STREAM,0);

if ( bind (s, &so_in, sizeof(so_in)) < 0 )
   {
   fprintf (" Erreur au moment du bind : arret\n ");
   exit(1);
   }
if ( connect (s, &so_in, sizeof(so_in)) < 0)
   {
   fprintf (" Erreur au moment du connect: arret\n ");
```

```
    exit(1);
    }

/*************************************************************/
/*inserer ici les ordres symetriques de communication (tout*/
/*ce qui est ecrit dans la procedure ci-dessus doit etre lu*/
/*ici et reciproquement )                                    */
/*************************************************************/


/* après la fin des échanges */


close(s);
}
```

Ici seuls les grands principes sont fournis. Toutes les déclarations et les initialisations de variables n'ont pas été recopiées, dans un but clarificateur, la finalité étant d'expliquer le mécanisme mis en oeuvre pour établir une liaison.

2.2 Protocoles utilisés

Le protocole actuellement utilisé dans nos échanges entre processus est minimum. Les conventions simplificatrices sont appliquées pour l'instant :

- Au moment d'une demande de connexion :

  Il se contente de vérifier les droits des deux protocoles à communiquer par le canal utilisé, et de rendre la main au processus appelant si la liaison demandée est impossible. Si la liaison est a priori possible (le canal existe, le processus distant autorise bien les connexions s'il est le *serveur*, le processus demandeur est lui-même *serveur*. ....), on attend que la connexion soit établie. Nous n'avons pas implémenté de fonction de time_out dans le système réel, bien que des tests sur des procédures d'essais fonctionnent. A terme, il sera nécessaire d'incorporer ce mécanisme. car les programmes sont bloqués en attente de connexion lorsque le réseau est défectueux, ou plus simplement lorsque le module distant ne fonctionne pas tout à fait comme prévu.

- En cours de dialogue :

Une liste exhaustive de demandes possibles, associée chacune à une réponse obligatoire, a été établie en fonction des besoins de chacun des modules. Cette liste est propre à chaque interface. Lorsqu'une demande inconnue est détectée (erreur de protocole, ou de transfert), on se contente de la signaler au module qui vient de recevoir cette information, et la liaison est fermée. Aucune reprise sur erreur n'est donc prévue pour l'instant. La même stratégie est adoptée en cas de réponse non conforme avec celle attendue.

Par contre, en cas d'absence de demande ou de réponse, les deux correspondants attendent le temps nécessaire. Là encore, nous n'avons pas mis en oeuvre pour l'instant de mécanisme de time_out.

- En cas d'arrêt brutal de l'activité de l'un des correspondants.

  Cet arrêt ne peut-être détecté que lors d'une lecture ou d'une écriture. (une lecture renverra alors fin de fichier, une écriture provoquera un signal SIGPIPE). Bien sûr, la communication est rompue de fait. La seule chose à faire est de réinitialiser proprement le coté encore actif, en fermant proprement les connexions encore ouvertes.

Ces procédures ont été mises en oeuvre pour permettre de tester le procédures de LEX en utilisant des résultats réels. Toutefois, seuls des tests avec les sytèmes Partner et Diapason ont été faits, la partie SYN–SEM du système DIAL étant en cours de réalisation. Le comportement de ce type de liaison est parfois surprenant sur Masscomp, surtout lorsque la machine doit gérer plusieurs exclusions mutuelles simultanées (si deux utilisateurs différents font à la fois des acquisitions de parole et communiquent à l'aide des outils décrits ci-dessus, il n'est pas rare de voir des blocages système se produire, se traduisant par un perte de contrôle de la part de l'usager....). Dans les autres cas, la liaison entre processus se comporte correctement, et permet de lancer de façon parallèle les niveaux lexicaux et supra-lexicaux (ce parallélisme est fictif, notre machine étant mono-processeur).

3 Modifications souhaitables

3.1 Modifications proposées vis-à-vis du décodage actuel

Si le module de décodage acoustico-phonétique fournit des résultats en progrès constants, ceux-ci restent entachés d'erreurs. Nous avons vu dans

le chapitre précédent que ces erreurs pouvaient être surmontées par une procédure adaptée de comparaison de mots, au moins dans une phase de vérification. Lorsqu'un phonème est omis, inséré ou substitué par l'un de ses "*voisins*", cela va diminuer le score de reconnaissance du mot considéré, mais sa bonne identification n'est pas exclue. Toutefois, les hypothèses sont faites de façon totalement aveugle par la procédure de comparaison, en s'appuyant sur des connaissances pragmatiques générales, et plus rarement sur des indices relevés dans l'énoncé courant. Ces hypothèses sont aveugles surtout parce qu'elles ne conduisent pas à une vérification a posteriori sur le signal. Pourtant, cette vérification est désormais possible dans un système conçu de façon modulaire comme DIAL. Il suffit d'élargir un peu l'interface existante entre APHON et LEX, pour permettre à ce dernier de poser des questions relative à la présence d'un phonème sur une plage de signal restreinte, en rappelant alors les contextes gauche et droit souhaités.

SYSTEXP était parfaitement capable de répondre à ce type de question, et ce module est en train d'être intégré à APHON.

La procédure de vérification lexicale d'un mot serait alors légèrement modifiée. Au lieu de pénaliser le score du mot en cours de test à chaque fois qu'une mise en correspondance n'est pas possible, pour toutes les erreurs liées aux limites du système de décodage acoustico-phonétique, une demande de validation de l'hypothèse sera faite.

Exemples :

(a) Phonème omis.

Plusieurs sous cas sont à envisager :

i. Le phonème attendu est le seul à manquer : ses prédécesseurs et successeurs sont présents.

ii. Le phonème omis a été englobé dans ses deux voisins, de même nature, et l'ensemble est vu comme un seul et même segment. Ce cas est plus génant que le précédent pour la reconnaissance lexicale, car il conduit à deux hypothèses d'élision avant de mettre en correspondance le phonème suivant.

Une première fonction sera donc chargée de retrouver le segment manquant. Un paramètre va indiquer le type de recherche à effectuer :

chercher (entre, ph, prec, suiv)

Cet appel provoquera la recherche du phonème ph entre les deux phonèmes prec et suiv. En cas de succès, les trois phonèmes passés

en paramètre seront remplacés par les trois nouveaux phonèmes trouvés. Les modifications porteront bien sûr sur les limites internes de ces trois segments, mais aussi sur les étiquettes. Une segmentation différente des deux phonèmes initiaux remettra en cause une étiquette basée sur l'étude d'un sur-ensemble de prélèvements. Lors de l'appel, les descripteurs de phonèmes contiendront :

Pour prec et suiv : son début et sa fin actuelle.
l'étiquette souhaitée (pour avoir un contexte exact au cas où notre hypothèse est valide)

Pour ph : début=fin de prec et fin=début de suiv
L'étiquette souhaitée.

chercher (dans, ph, cour, cour)

Cet appel provoquera la recherche d'un phonème de type ph inclus dans cour. En cas de succès, le phonème cour sera remplacé par les trois nouveaux phonèmes trouvés. Les renseignements associés lors de l'appel sont identiques à ceux de l'appel ci-dessus.

(b) Classe d'un phonème présent erronée.

Nous avons déjà expliqué que la segmentation en classes phonétiques était le résultat de l'application de trois algorithmes indépendants, qui recherchent plus particulièrement une classe de phonème.

Examinons les différents cas d'erreurs qui peuvent apparaître, et le moyen d'y remédier :

i. Omission d'un noyau vocalique :

Lorsque le segment est présent, il est le plus souvent étiqueté comme n'appartenant pas aux trois classes cherchées, et reçoit donc le label de classe "I", pour inconnu. (C'est une classe où sont rangées à la fois les liquides, les consonnes nasales, les semi-voyelles et certains bursts non fricatifs).

Il suffit de vérifier que le segment est assez proche d'un noyau vocalique. Une façon simple de procéder consiste à appeler la fonction standard de détection des noyaux vocaliques, en prenant des paramètres de sélection un peu plus souples. Une autre méthode consiste à vérifier que le segment est voisé, et contient des formants à des valeurs pas trop éloignées des valeurs théoriques attendues, dans le contexte particulier où la voyelle omise devrait se trouver. Celui-ci est parfaitement connu si le mot en cours de test est correct.

Si le segment est étiqueté comme fricatif et voisé, et si la voyelle attendue est une voyelle d'avant ("*i* ", à la rigueur "*é* "), une vérification de la présence de formants sera suffisante.
Les autres étiquettes (Plosives ou non voisées) continueront à provoquer une pénalisation forte du score de reconnaissance.

ii. Omission d'une fricative intense (autre que "*f* " et "*v* ")
L'algorithme de recherche de fricatives sera appliqué en diminuant légèrement le seuil minimum.

iii. Omission de "*f* " et "*v* ")
Deux étiquettes sont plausibles : plosive ou inconnue.
Pour une étiquette "plosive", il faudra vérifier que l'occlusion n'est pas parfaite : une présence de bruit diffus, même inférieur au seuil requis, sera un élément favorable. Si la médiane de ce bruit est élevée, l'hypothèse sera à retenir.

iv. Omission d'une plosive :
Une plosive sourde, réalisée par le locuteur, ne pourra être complètement omise lors de l'analyse automatique du signal. Il suffira de regarder si un tel segment n'avait pas été détecté, puis rejeté à cause d'un critère sélectif trop sévère dans ce cas, telle une longueur insuffisante.
Ce type de vérification s'applique aussi aux plosives voisées, mais un remplacement par une consonne nasale dans un contexte nasalisé est fréquent. Si une règle de phonologie ne s'applique pas directement (par exemple parce que les nasales ne sont pas des voisins immédiats de la plosive considérée : *Tout le MONDe N'est pas venu.*
Il faudra vérifier que cette nasale est peu énergétique, et que son énergie diminue à la fin.

v. Omission d'une nasale :
Le plus souvent, des nasales peu énergétiques reçoivent une étiquette de plosive voisée. Un test de présence de formants aux valeurs attendues suffira à valider une hypothèse de remplacement.

(c) Segment omis :

i. Omission d'une plosive voisée :
Lorsque l'élocution est rapide, et/ou les conditions d'acquisition non parfaites, la partie occlusive d'une plosive voisée, déjà peu importante, peut être entièrement couverte par un écho provenant de la voyelle précédente

Exemple : la dent → /l/a/d/ã/ pourra produire un décodage du type :
/l/ã/ã/
Une très petite portion du signal, correspondant à la fin de la partie occlusive , non étiquetée, peut toutefois subsister. Un suivi de formant et une étude de la variation d'énergie permettent d'évaluer la plausibilité de présence d'un son /d/, très bref, entre les deux noyaux vocaliques trouvés.

ii. Omission d'une liquide : Elle est en général incluse dans le noyau vocalique voisin. Il faudra donc vérifier que la partie correspondante du noyau a des formants qui convergent vers les valeurs attendues pour cette liquide.

iii. Omission d'une nasale :
Deux cas sont à considérer :

cette nasale est suivie d'une plosive, elle est peu énergétique et a été incluse dans celle-ci. Il suffit de vérifier le présence de formants.

cette nasale a été incluse dans une voyelle : un suivi de formant permet de le vérifier.

(d) trait de voisement erroné

Le critère voisé/non-voisé est attribué selon le nombre relatif de prélèvements présentant cet indice pour ce phonème. Une correction est possible si la majorité trouvée était peu significative.

Si l'on se trouve en début d'énoncé, un dévoisement presque total est possible, car le lancement des cordes vocales n'est pas un phénomène instantané. Un petit nombre de prélèvements voisés, mais situés en fin du premier phonème, peut être suffisant pour attribuer le critère au phonème initial.

3.2 Adaptation automatique à la voix du locuteur

Le module de décodage acoustico-phonétique utilise beaucoup de paramètres représentant des seuils, qui servent d'une part à la segmentation, d'autre part à l'étiquetage. Tous ces paramètres sont fixés au départ du système, à une valeur moyenne, calculée sur un ensemble de locuteurs. Le système devant être multi-locuteurs, sans apprentissage préalable, cette solution est la seule possible pour traiter le premier énoncé prononcé par un locuteur inconnu.

Toutefois, dès que le premier énoncé a été reconnu de façon correcte, on peut se servir des indices calculés pour ajuster de façon dynamique l'ensemble des paramètres du système.

Adaptation des paramètres seuils (fricatives, plosives)

Les paramètres les plus faciles à ajuster sont ceux qui correspondent à des seuils. Par exemple, à partir de quelle hauteur du centre de gravité du bruit doit-on considérer un son comme fricatif?

Une valeur moyenne a été déterminée pour une acquisition à 16000 Hz, nous avons retenu 4000 Hz. Ce seuil dépend toutefois du fondamental du locuteur, un homme ayant une voix grave verra une partie de ses fricatives non reconnues. Un ajustement dynamique en fonction du pitch et de la valeur relevée sur des fricatives présentes est donc souhaitable.

Représentation des voyelles : recalcul des traits caractéristiques

Chaque voyelle est décrite dans le système par la hauteur moyenne de ses trois premiers formants. Lorsqu'un noyau vocalique a été isolé, nous calculons pour chaque prélèvement appartenant à ce noyau une transformée de Fourrier rapide, sur laquelle nous recherchons ces trois premiers formants. Le noyau recevra alors pour étiquettes les voyelles les plus proches de l'ensemble de ces prélèvements.

Mais, si la position relative des voyelles dans l'espaceF_1F_2 correspondant aux deux premiers formants est la même, celui-ci est plus ou moins étendu ou translaté en fonction de caractéristiques physiques du locuteur (longueur du conduit vocal, volume des cavités buccales et nasales, ...) La figure 5.3 nous montre cette distorsion avec deux locuteurs, l'un masculin, l'autre féminin.

Une transformation faisant coïncider les points extrêmes de ce triangle des voyelles avec des valeurs mesurées améliorerait sans doute la qualité d'étiquetage. De tels travaux ont déjà été faits au CRIN [PISTER 84], il faut pour cela recaler après identification de l'énoncé les phonèmes avec leur réalisation, afin de faire des mesures précises des indices qui nous intéressent. Pour effectuer le recalage, des mots contenant les trois voyelles extrêmes (/a/, /i/ et /u/) et bien reconnus sont nécessaires. On utilisera ensuite leur transcription exacte pour faire le recalage, c'est à dire une re-segmentation a posteriori de la plage de signal correspondante.

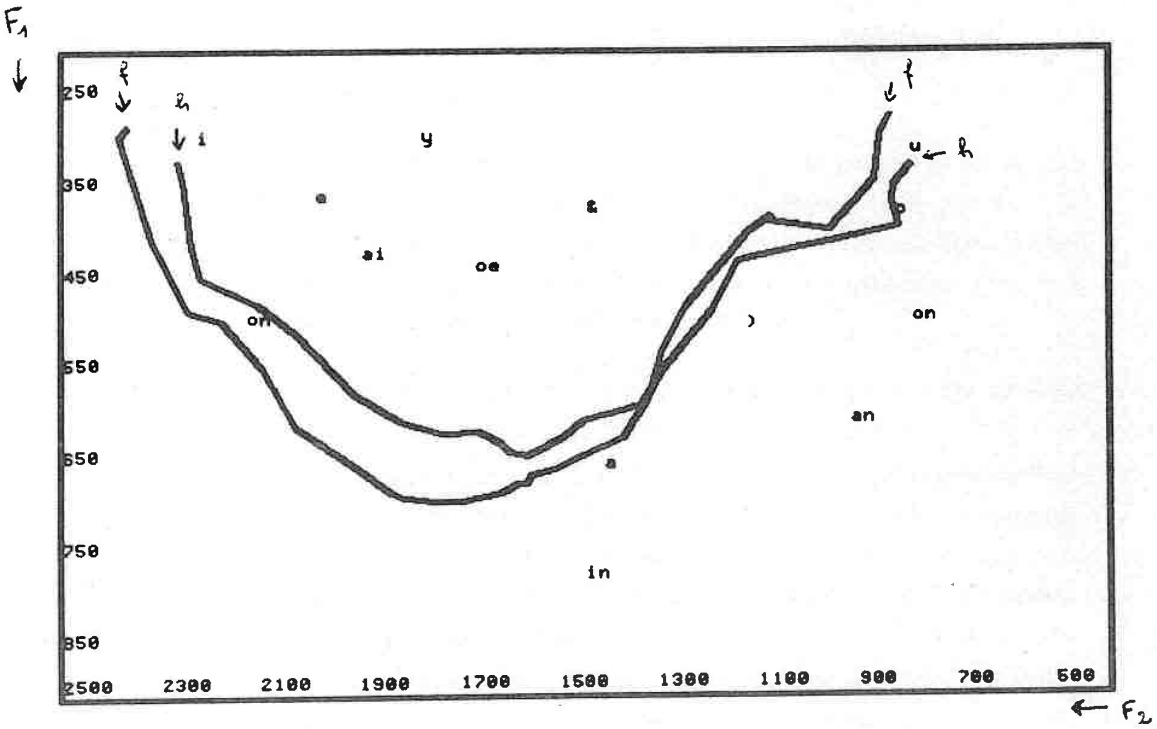


Figure 5.3: Triangle des voyelles /a/, /i/ et /u/ pour 2 locuteurs

de faire des mesures précises des indices qui nous intéressent. Pour effectuer le recalage, des mots contenant les trois voyelles extrêmes (/a/, /i/ et /u/) et bien reconnus sont nécessaires. On utilisera ensuite leur transcription exacte pour faire le recalage, c'est à dire une re-segmentation a posteriori de la plage de signal correspondante.

Conclusion

Dans ce manuscrit, nous avons abordé les problèmes qui se posent au niveau lexical dans un système de reconnaissance automatique de la parole. La position centrale du module de traitement lexical nous a constamment obligé à faire référence aux niveaux voisins, infra-lexicaux et supra-lexicaux, et à l'ensemble du système. Notre objectif était double :

(a) Beaucoup de systèmes limités avaient déjà été étudiés, voire réalisés dans ce domaine.

Les systèmes MYRTILLE I et II mis au point dans notre équipe avaient permis de tester un certain nombre de solutions permettant la compréhension de phrases, d'abord en langage artificiel, puis pseudo-naturel. Toutefois, dans ces deux systèmes, les différentes phases de traitement, comme la reconnaissance et la compréhension, étaient nettement distinctes, et interagissaient un minimum entre elles. Les progrès de la technique permettent d'envisager aujourd'hui des architectures beaucoup plus étoffées, avec une grande coopération entre les différents modules. Notre groupe de travail a donc décidé de reprendre une réflexion à plus long terme pour réaliser un tel système, modulaire, privilégiant les interactions entre ses différentes composantes. Cette réflexion de fond a conduit à la définition de l'architecture d'un nouveau système, DIAL, qui a constitué l'une de nos préoccupations tout au long de ce manuscrit.

Si l'architecture et les spécifications de ce système sont aujourd'hui parfaitement claires, l'ensemble des procédures de traitement n'est pas encore achevé, mais un certain nombre sont d'ores et déjà en phase de validation.

(b) Parmi les résultats et les idées nouvelles développés lors de la définition de DIAL, beaucoup s'appliquent à des systèmes moins ambitieux. Nous avons donc essayé de les valider en développant deux applications concrètes, à court terme. Cela a donc conduit notre équipe à la réalisation

des deux systèmes Partner et Diapason. Le module lexical de ces deux application,, objet d'une partie de ce travail, est opérationnel.

Pour chacun des ces trois systèmes, notre principale préoccupation n'a pas été de mettre en oeuvre à tout prix un système entièrement opérationnel, en temps réel, mais plutôt de tester un certain nombre d'idées, tant au niveau du modèle des systèmes eux-mêmes, qu'à des niveaux beaucoup plus limités tels la recherche rapide de mots dans un treillis phonétique. Si des systèmes relativement contraints, comme Diapason ou Partner, doivent déboucher à court terme sur des systèmes opérationnels multi-locuteurs, réservés à des locuteurs habitués à leur utilisation, et dans le cadre d'applications finalisées et relativement restreintes, DIAL n'est qu'une étape vers la réalisation de systèmes plus ambitieux et plus conviviaux.

Seule l'inter-connexion effective des différents modules de DIAL permettra d'évaluer ce nouveau système. Cependant, on peut d'ores et déjà prévoir que des améliorations du décodage acoustico-phonétique, ou plutôt de l'interface entre le module de traitement lexical et le module de décodage acoustico-phonétique, seront nécessaires. D'autre part, des études plus fondamentales, débordant du cadre strictement informatique et faisant appel à des spécialistes en linguistique, en ergonomie, en phonétique, voire en psychologie, seront sans doute nécessaires pour affiner les connaissances du système et pour établir ses limites.

Beaucoup de travail reste à faire et la poursuite des travaux de notre groupe devra se faire en faisant le plus d'échanges possibles avec des chercheurs d'autres disciplines. Nous avons définitivement dépassé le stade où chaque chercheur effectue un travail personnel très focalisé. Une telle recherche ne pourra aboutir que si elle est menée par une équipe à la fois pluri-disciplinaire et parlant un même langage, c'est-à-dire travaillant dans ce cadre précis. Les progrès souhaitables à différents niveaux (décodage acoustico-phonétique, traitements linguistiques, ...), ne pourront s'accomplir qu'à l'aide des spécialistes de ces domaines, qui collaborent déjà avec notre équipe, ou qui accepteront de s'intéresser à nos problèmes. C'est la clef de la réussite, surtout si l'on prétend fournir au grand public des systèmes de dialogue oral homme-machine en langage continu, suffisamment sophistiqués pour offrir des degrés de compréhension et de production de parole comparables à un humain.

Bien sûr, ce type de système reste un objectif de recherche à long terme, mais il faut commencer dès à présent à développer des systèmes de dialogue limités à des applications simples, compatibles avec l'utilisation de langages

restreints. Nous nous y attachons et, bien que nos travaux restent modestes, il n'est pas exclu de permettre un jour au grand public d'interroger des centres de renseignements automatisés par téléphone, sans être contraint par un langage très limité.

Bibliographie

[ALINAT 87] P. ALINAT, E. GALLAIS, J.P. HATON, J.M. PIERREL, et P. RICHARD. A continuous speech dialog system for oral control of sonar console. *Proceedings of IEEE ICASSP-87*, 1987.

[AMALBERTI 86] R. AMALBERTI, N. CARBONELL, P. FALZON, et C. JOLIVET. La communication orale homme-homme : un modèle de référence pour la communication orale homme-machine? Rapport d'ATP, ARI communication, 1986.

[BONIN 87] J.J. BONIN. Détection d'indices prosodiques linguistiquement significatifs. Rapport de DEA, Nancy 1, 1987.

[CARBONELL 84a] N. CARBONELL, F. CHARPILLET, J.P. HATON, B. MANGEOL, P. MOUSEL, J.M. PIERREL, et A. ROUSSANALY. Dialogue oral homme-machine : bilan du projet MYRTILLE et perspectives. *Actes du séminaire GRECO sur le dialogue oral homme-machine*, pages 91–122, Nancy, 1984.

[CARBONELL 84b] N. CARBONELL, D. FOHR, J.P. HATON, F. LONCHAMP, et J.M. PIERREL. An Expert System For Automatic Reading of French Spectrograms . *Proceedings of the IEEE 1984 - ICASSP*, page , San diego, 1984.

[CARBONELL 85] N. CARBONELL, B. MANGEOL, P. MOUSEL, J.M. PIERREL, et A. ROUSSANALY. Les sources de connaissances dans un système de dialogue oral homme-machine. *Actes du 5ème Congrès AFCET Reconnaissance des Formes et*

Intelligence Artificielle. pages 263–279, Grenoble, 1985.

[CARBONELL 86] N. CARBONELL et J.M. PIERREL. Architecture and knowledge sources of an human-computer oral dialogue system. *Proceedings of the NATO workshop, structure of multimodal dialogues including voice*, Corsica, France, 1986.

[CARBONELL 88] N. CARBONELL et J.J. BONIN. Utilisation d'informations prosodiques en reconnaissance de la parole continue. *Actes des 17 ième Journées d'études sur la parole*, page , Nancy, 1988.

[CAUSSE 76] B. CAUSSE, D. DOURS, R. FACCA, et G. PERENNOU. Evaluation d'une méthode ascendante d'analyse lexicale dans le discours continu. *Actes des 7 ièmes Journées d'études sur la parole*, pages 37–53, Nancy, 1976.

[CHARPILLET 85] F. CHARPILLET. Un système de reconnaissance de la parole continue pour la saisie de textes lus. Thèse de doctorat de l'université de Nancy 1, 1985.

[COLNET 86] D. COLNET, G. MASINI, A. NAPOLI, Y. NOIRET, et K. TOMBRE. Les langages orientés objets. Rapport CRIN n. 86-R-077, 1986.

[COMBESCURE 81] P. COMBESCURE. Vingt listes de phrases phonétiquement équilibrées. *Revue d'acoustique*, 14(56):, 1981.

[COULON 86] D. COULON et D. KAYSER. Informatique et langage naturel : présentation générale des méthodes d'interprétation de textes écrits. *Techniques et Sciences Informatiques*, 5(2):103–128, 1986.

[DESCLES 82] J.P. DESCLES. Langages quasi-naturels et catégories grammaticales. *Actes du colloque ARC Domaines et objectifs de la recherche cognitive*, pages 109–140, Pont-à-Mousson, 1982.

[DEVILLE 86] G. DEVILLE et H. PAULUSSEN. A case grammar as an original linguistic model for the seman-

tic representation of utterances in a man-machine dialog system. Thesis of computational linguistics, University of Antwerpen, Belgique, 1986.

[DEVILLE 87a] G. DEVILLE. *Corpus de dialogues oraux finalisés en situation réelle : Demande de renseignements auprès du ministère de l'emploi et du travail (Cellule Action-Travail).* ed G. Deville, FUNDP Namur, 1987.

[DEVILLE 87b] G. DEVILLE, H. PAULUSSEN, et J.M. PIERREL. Une grammaire de cas comme modèle de représentation sémantique d'énoncés de dialogues oraux homme-machine finalisés. *Actes du 6ème Congrès AFCET Reconnaissance des Formes et Intelligence Artificielle*, pages 159–173, Antibes, 1987.

[DEWEZE 81] A. DEWEZE. Les réseaux sémantiques. essai de modélisation. application à l'indexation et à la recherche d'informations documentaires. 1981.

[ELMAN 84] J.L. ELMAN et J.L. Mc CLELLAND. Speech perception as a cognitive process; the interactive model. *Speech and Langage : Advances in basic reseach and practice*, 10():, 1984.

[ERMAN 77] L.D. ERMAN. A functional description of the HEARSAY II speech understanding system. *Proceedings of IEEE ICASSP-77*, 1977.

[FILLMORE 68] C. FILLMORE. *The Case for Case in Universals in Linguistic Theory* . E. Bach and R.T. Harm eds. Rinehart and Winston, 1968.

[FOHR 86] D. FOHR. APHODEX : un système expert de décodage acoustico-phonétique de la parole continue. Thèse de doctorat de l'université de Nancy 1, 1986.

[FORSTER 76] K.I. FORSTER. Acessing th emental lexicon. E.C. WALKER et R.J. WALES, éditeurs, *New approch to langage mechanisms*, page , North Holland. Amsterdam, 1976.

[GRUAZ 86] C. GRUAZ. *Formes et fonctions d'un composant phonémique en IA.* CNRS Communication parlée et GALF, Toulouse, 1986.

[HATON 85] J.P HATON. Intelligence artificielle en compréhension automatique de la parole: état des recherches et comparaison avec la vision par ordinateur. *Techniques et Sciences Informatiques*, 4-3:265–287, 1985.

[KAYSER 84] D. KAYSER. Examen de diverses méthodes utilisées en représentation des connaissances. *Actes du colloque Afcet RFIA*, pages 115–144, Paris, 1984.

[LEA 80] W.A. LEA, éditeur. *Trends in Speech Recognition.* Prentice Hall, 1980.

[MARSLENWILSON 78] W.D. MARSLEN-WILSON et A. WEHSH. Processing interaction and Lexical Access during Word recognition in Continuous Speech. *Cognitive Psychology*, 10():29–63, 1978.

[MELONI 82] H. MELONI et J. GUIZOL. Utilisation de paramètres prosodiques dans un système de reconnaissance de la parole continue. *Actes du séminaire "Prosodie et reconnaissance de la parole continue*, pages 93–120, Aix-en-Provence, 1982.

[MERCIER 77] G. MERCIER, P. QUINTON, et R. VIVES. Dialogue homme–machine avec KEAL. *Recherches acoustiques*, 4:15–22, 1977.

[MOON 86] D.A. MOON. Object-Oriented Programming with Flavors. *OOPSLA's 86 Proceedings*, page , 1986.

[MORIN 87] P. MORIN et J.M. PIERREL. PARTNER: un système de dialogue oral homme machine. *Actes du congrès COGNITIVA*, pages 354–361, Paris, 1987.

[MORTON 79] J. MORTON. Some experiments of facilitation in word and picture recognition and their relevance for the evolution of a theorical position. P.A.

KOLERS, M.E. WROLSTAD, et H. BOUMA, éditeurs. *The processing of visual langage*, page , Plenum Press, New York, 1979.

[PERENNOU 82] G. PERENNOU et J. CAELEN. Utilisation de la prosodie pour la reconnaissance de la parole dictée. *Actes du séminaire "Prosodie et reconnaissance de la parole"*, pages 25–57, Aix-en-Provence, 1982.

[PERENNOU 86] G. PERENNOU. BDLEX : A data and cognition Base of spoken french. *IEEE ICASSP*, page , 86.

[PIERREL 75] J.M. PIERREL. Contribution à la compréhension automatique du discours continu. Thèse de 3ième cycle, université de Nancy 1, 1975.

[PIERREL 81] J.M. PIERREL. Etude et mise en oeuvre de contraintes linguistiques en compréhension automatique du discours continu. Thèse d'état, Nancy 1, 1981.

[PIERREL 87] J.M. PIERREL. *Dialogue oral homme-machine.* Hermès, Paris, 1987.

[PISTER 84] C. PISTER. Adaptation au locuteur par apprentissage automatique, application à un système de reconnaissance automatique de la parole. Thèse de 3ième cycle, Nancy 1, 1984.

[RABINER 78] L.R. RABINER et R.W. SCHAFER. *Digital Processing of Speech Signal.* Bells Laboratories, 1978.

[ROMARY 87] L. ROMARY. Rapport de DEA Nancy 1, 1987.

[ROMARY 88] L. ROMARY et B. MANGEOL. Prédiction et vérification lexicale dans le cadre d'un dialogue oral homme-machine. *Actes des 17ième JEP*, (), 88.

[ROUSSANALY 86] A. ROUSSANALY, P. MOUSEL, N. CARBONELL, B. MANGEOL, et J.M. PIERREL. Réalisation d'un corpus de dialogues oraux. application aux renseignements administratifs. Rapport interne CRIN, n.86-R-083, 1986.

[ROUSSANALY 88] A. ROUSSANALY. Dial : la composante dialogue d'un système de communication orale homme machine finalisée en langage naturel. Thèse de l'université de Nancy 1, 1988.

[RUSKE 86] G. RUSKE et W. WEIGEL. Automatic recognition of spoken sentences using a demisyllabe-based Dynamyc Programming algorithm. *12th International Congress on Acoustics*, pages A1–5, Toronto, 1986.

[SEGUI 80] J. SEGUI. Code phonétique et code phonologique dans la perception de la parole : le rôle de la syllabe. *Journées d'études codage phonétique et phonologique*, page , Ecole des hautes études, 1980.

[SEGUI 88] J. SEGUI. L'accès au lexique, données expérimentales et modèles. , éditeur, *Calliope, 1988*, page , 1988.

[SHOUP 79] J. SHOUP. Phonological aspects of speech recognition. Wayne A. Lea, éditeur, *Trends in speech recognition*, pages 125–138, Prentice Hall, 1979.

[VAISSIERE 82] J. VAISSIERE. Utilisation de paramètres supra-segmentaux comme aide à la segmentation en phonèmes. *Actes du séminaire "Prosodie et reconnaissance automatique de la parole"*, pages 123–139, Aix-en-Provence, 1982.

[WALKER 78] D.E. WALKER, éditeur. *Understanding spoken language.* North-Holland, 1978.

[WOLF 80] J. WOLF et W.A. WOODS. *The HWIM speech understanding system*, pages 316–339. W.A. LEA editor, Prentice-Hall, 1980.

[WOODS 70] W.A. WOODS. Transition network grammars for natural language analysis. *Comunication of ACM*, 13-10:591–602, 1970.

[WOODS 76] W.A. WOODS et V. ZUE. Dictionary expansion via Phonological Rules for a Speech Understanding System. *IEEE International Confer-*

ence on Acoustics, Speech, and Signal Processing, pages 561–564, Philadelphia, 1976.

NOM DE L'ETUDIANT : MANGEOL Bernard

NATURE DE LA THESE : Doctorat de l'Université de NANCY I en Informatique

VU, APPROUVE ET PERMIS D'IMPRIMER

NANCY, le 15 SEP. 1988 1504

LE PRESIDENT DE L'UNIVERSITE DE NANCY I

R. MAINARD

La composante lexicale dans les les systèmes de dialogue oral Homme-machine du CRIN

Résumé

L'objet de ce travail est d'étudier la composante lexicale de divers systèmes de dialogue oral Homme-machine. après une revue rapide de l'existant, nous avons mis en œuvre la composante lexicale de trois systèmes développés au CRIN :

- Le système Diapason, qui permet d'interpréter des commandes orales à un système Sonar.
- Le système Partner, paramétré à la fois par la langue de travail et l'application désirée.
- Le système Dialog, qui constitue une recherche à plus long terme. Le but est de mettre en place un système de renseignements administratifs, destiné au grand public. Nous proposons tout d'abord une nouvelle architecture adaptée à ce type de système, et nous détaillons ensuite le type d'informations lexicales nécessaires à sa mise en œuvre. Basé sur cinq composantes (décodage acoustico-phonétique, recherche d'indices prosodiques, modules de recherche et de vérification lexicale, module d'analyse syntaxico-sémantique et gestionnaire de dialogue), cette architecture permet d'envisager un système de compréhension réparti, et chacune des composantes peut se dérouler de façon indépendante.

  En dernier lieu, nous proposons une nouvelle interface entre les modules de recherche lexicale et les niveaux infra-lexicaux de segmentation du signal et de décodage acoustico-phonétique. Celle-ci permet de remettre en cause des informations non validées, et d'éviter les erreurs parfois grossières des systèmes actuels.

Mots-clefs :

- Recherche lexicale
- Dialogue
- Compréhension automatique de la parole
- Langage naturel
- Commandes orales de systèmes.