Projet OUTILEX

Rapport final

LINGWAY *

Hugues.de-Mazancourt@lingway.com Vincent.LeMaout@lingway.com

25 octobre 2006

1 Lingway Knowledge Management

Le volume sans cesse croissant de l'information disponible (notamment sur Internet) nécessite des outils de plus en plus performants. De plus, la variabilité des langues est un fait de plus en plus incontournable, que ce soit au niveau de l'utilisateur ou des textes. **Lingway Knowledge Management** (LKM) est une application permettant de retrouver, organiser et comprendre l'information plus facilement et plus efficacement grâce à une approche sémantique du document. Elle regroupe un certain nombre d'outils de gestion, d'accès et de suivi de l'information qui s'articulent autour de quatre grands axes :

- L'acquisition et la veille avec un « crawler web » et la gestion d'alertes ;
- La recherche d'information avec un moteur sémantique ;
- L'organisation avec un moteur d'indexation et de catégorisation ;
- L'aide à la compréhension avec un moteur de présentation et d'aide à la lecture.

Les moteurs de recherche, d'indexation et de catégorisation intègrent de nombreuses fonctionnalités basées sur des modules linguistiques implémentés grâce à des automates à états finis. Plus précisemment, la contribution du projet Outilex à LKM concerne la reconnaissance des mots composés qui intervient lors de l'analyse morphologique et syntaxique d'une requête destinée au moteur de recherche et le « chunking » qui lui intervient pendant la phase d'analyse des documents.

1.1 Recherche sémantique multilingue

Le moteur de recherche sémantique permet d'interroger divers formats de bases textuelles :

*http://www.lingway.com

– des bases dites « full text » ;
– des bases indexées par des méta-données (éventuellement générées automatiquement par le module d'indexation-extraction) ;
– des bases organisées autour d'un plan de classement ou taxonomie.

Ces différents modes de recherche peuvent être utilisés séparément ou combinés entre eux, se renforçant ainsi mutuellement. Basé sur une ontologie de 150.000 concepts mis en correspondance avec 6 langues, ce moteur est capable de recherches « cross-language », c'est-à-dire de trouver des documents écrits dans une autre langue que celle de la question (les langues actuellement traitées sont l'anglais, le français, l'allemand, l'espagnol, le néerlandais et le portugais).

Le moteur de recherche Lingway ne pose pas de contraintes fortes sur le modèle d'indexation : il peut s'adapter à des bases indexées de manière traditionnelle (liste inverse classique) ou, au contraire, s'adapter à des textes ayant subi un traitement plus élaboré lors de l'indexation (lemmatisation, indexation par entités nommées, par termes extraits, etc.).

Grâce aux possibilités d'analyse des questions en langage naturel, le moteur est simple d'utilisation et particulièrement bien adapté à des applications destinées à des utilisateurs non-spécialistes. Cette facilité d'utilisation s'accompagne d'une amélioration sensible de la qualité des résultats obtenus : mesurée en termes de précision et rappel, elle est largement supérieure à la qualité habituelle des moteurs basés sur d'autres méthodes.

1.1.1 Analyse de la requête

Le module linguistique produit une représentation interne de la requête de l'utilisateur en réalisant une analyse linguistique complète. Cette analyse se décompose en trois niveaux :

– **morphologique** : correspond à la reconnaissance des mots simples et des mots composés sous toutes leurs formes (genre, nombre, verbes conjugués, etc.). La reconnaissance de mots composés met en œuvre des traitements linguistiques, comme la décoordination : par exemple, on identifiera les mots composés « ligne à haute tension » et « ligne à basse tension » dans la requête « lignes à haute et basse tension ». Ce traitement fait partie du module Outilex décrit à la section 2 page 7 ;

– **syntaxique** : identifie la structure des requêtes et les rôles des mots les uns par rapport aux autres dans la requête d'origine. Une des conséquences de ce traitement est l'attribution de différents poids relatifs aux différents mots en fonction de leur rôle syntaxique dans la question ;

– **sémantique** : identifie le sens des mots. Ceci permet par la suite la recherche par synonymes ou mots proches ou encore dérivés, (par exemple « généticien » - « génétique »). Ces mots constituent « l'expansion sémantique » du mot initial. Cette phase d'identification du sens de chaque mot est importante, car l'expansion de la question vers d'autres mots dépend du sens reconnu. Par exemple, le mot « avocat » dans « avocat spécialiste de la fiscalité » pourra être expansé « juriste » ou « lawyer » (si l'on fait une recherche multilingue), alors que « recette de salade d'avocat » sera expansé vers « fruit exotique », « cuisine », etc.

1.1.2 Interprétation des requêtes

Sur la base de l'analyse linguistique, le logiciel propose une série d'interprétations de la requête de l'utilisateur. Ces interprétations comportent :

- Un expansion en une des série de mots de sens équivalent ou proche des mots de l'utilisateur (en fonction de l'analyse sémantique), classées en fonction des sens du mot initial, puis de leur distance sémantique à ce mot ;

- Une dégradation (ou « généralisation ») de la requête initiale. Par exemple, on proposera une requête analogue à «production de fraises» à partir de la requête «production de fraises en Limousin», ou «RTT et emploi des jeunes» à partir de «conséquence de la RTT sur l'emploi des jeunes».

Deux modes d'interprétation des questions sont possibles :

1. **en langage naturel** : dans ce mode l'utilisateur entre une phrase en langage libre et le moteur la traduit dans une requête booléenne en utilisant les mécanismes d'expansion et de dégradation en clauses décrits ci-dessus. Cela permet de réduire considérablement le silence, en élargissant la question à des formulations voisines.
   Dans ce mode « veille concurrentielle en ligne » peut trouver des réponses à « veille Internet » par expansion de « en ligne » sur « internet » et abandon de l'adjectif « concurrentiel ».
   À partir de l'analyse de la question, le moteur de recherche sémantique construit une requête documentaire qui dépend du modèle d'indexation et de la syntaxe particulière de l'indexeur, dans le cas où l'on utilise un connecteur vers un autre système de gestion documentaire.

   L'exemple de la figure 1 page 4 montre le résultat d'une recherche en langage naturel. Dans cet exemple, sur une base exemple d'articles de presse, la question posée est « programme nucléaire iranien ». 48 documents sont proposés en réponse. Pour chaque document est donnée la liste des mots qui ont permis de le sélectionner. La colonne de gauche donne une liste des termes les plus fréquemment extraits dans la liste des documents en réponse ainsi que l'analyse de la requête (à chaque mot considéré comme sémantiquement intéressant sont associés un certain nombre d'expansions avec leurs domaines sémantiques).
   En cliquant sur l'icône correspondant à un document on ouvre sa version « highlightée » où les mots trouvés par le moteur sont colorisés ainsi que les entités nommées détectée. On peut remarquer que « nucléaire » s'est étendu sémantiquement vers « atomique » et « iranien » vers « Iran ».

2. **en mode « avancé » ou « pseudo booléen »** : dans ce mode l'utilisateur a plus de contrôle sur la formulation de la question, en interdisant notamment la dégradation. Le moteur passe automatiquement en mode avancé dès qu'un opérateur booléen est entré

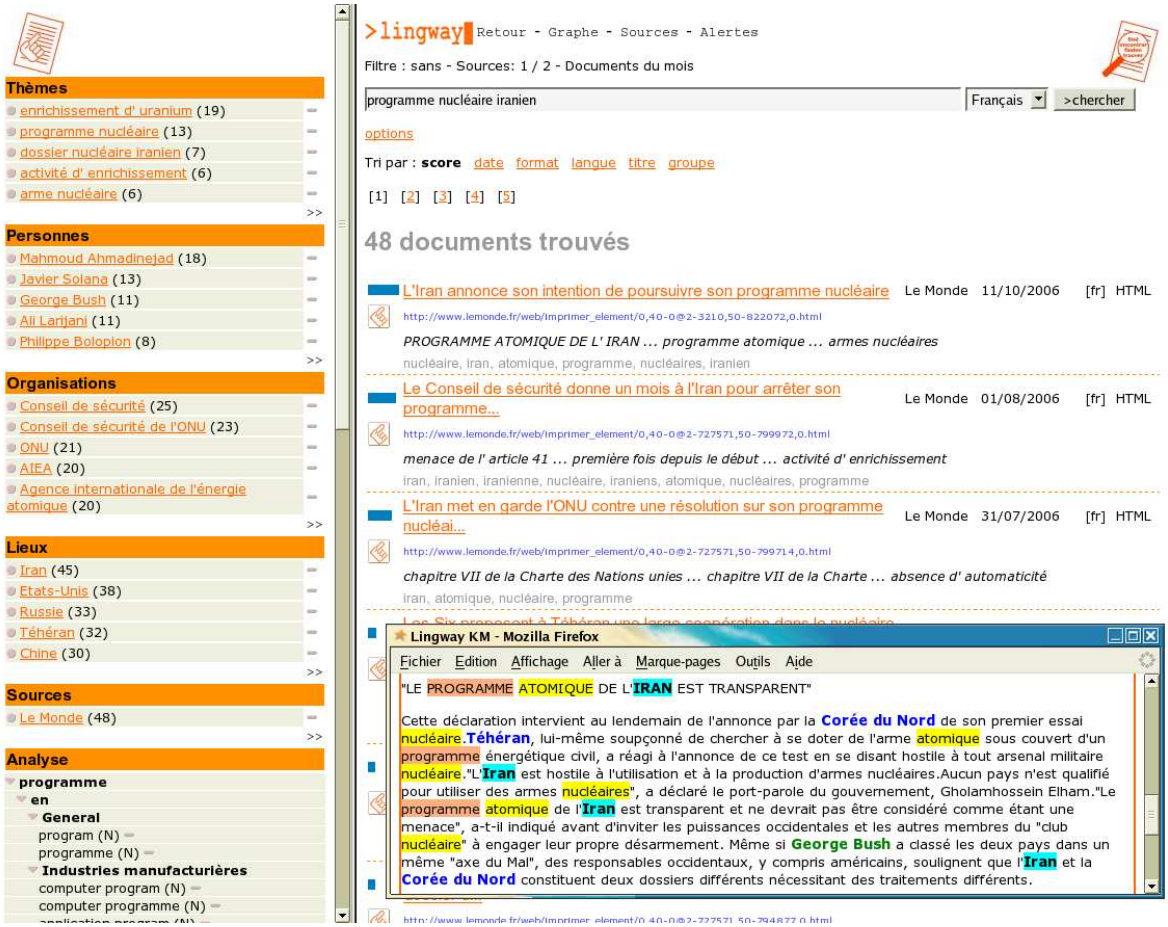


FIG. 1 – « programme nucléaire iranien »

(& pour ET logique, | pour OU logique, et "..." pour garder un mot à l'identique). Les parties séparées par un opérateur logique ne sont pas dégradées en sous clauses, mais l'expansion est en revanche réalisée.
Dans ce mode, « veille & concurrentiel & "en ligne" » gardera toujours les 3 termes, mais on pourra transformer par exemple en « veille & concurrent & "en ligne" ».
Ce mode est notamment bien adapté à la mise au point de profils de veille pour la gestion des alertes.

1.1.3 Paramétrage des stratégies de recherche

De nombreux paramétrages sont disponibles pour agir sur le comportement du moteur et le contenu des interprétations : privilégier les sens de domaines donnés, fixer des distances maximales pour les mots proches, parcourir ou non certaines relations du réseau sémantique, etc.

Ces stratégies de recherche peuvent être différentes selon les divers champs à interroger. Typiquement, on peut vouloir élargir le champ de la requête (c'est-à-dire utiliser des mots plus génériques) pour le titre, alors que l'on peut privilégier la recherche de mots plus précis

pour le corps du document.

1.1.4 « Cross-Language »

La fonction de CLIR (pour « Cross-Language Information Retrieval ») permet de générer des interprétations dans une langue différente de la langue de la requête. Cela permet d'interroger un fonds documentaire à partir d'une langue différente de la langue utilisée par les utilisateurs pour leurs requêtes. Cette fonction peut s'utiliser avec plusieurs langues comme cibles.

Par exemple, on pourra interroger en français une base contenant des documents en français et en anglais. LKM interrogera alors les documents français avec des mots français, les documents anglais avec des mots anglais de même sens.

L'exemple de la figure 2 montre un exemple de recherche dans une base de documents en anglais à partir d'une question posée en français.

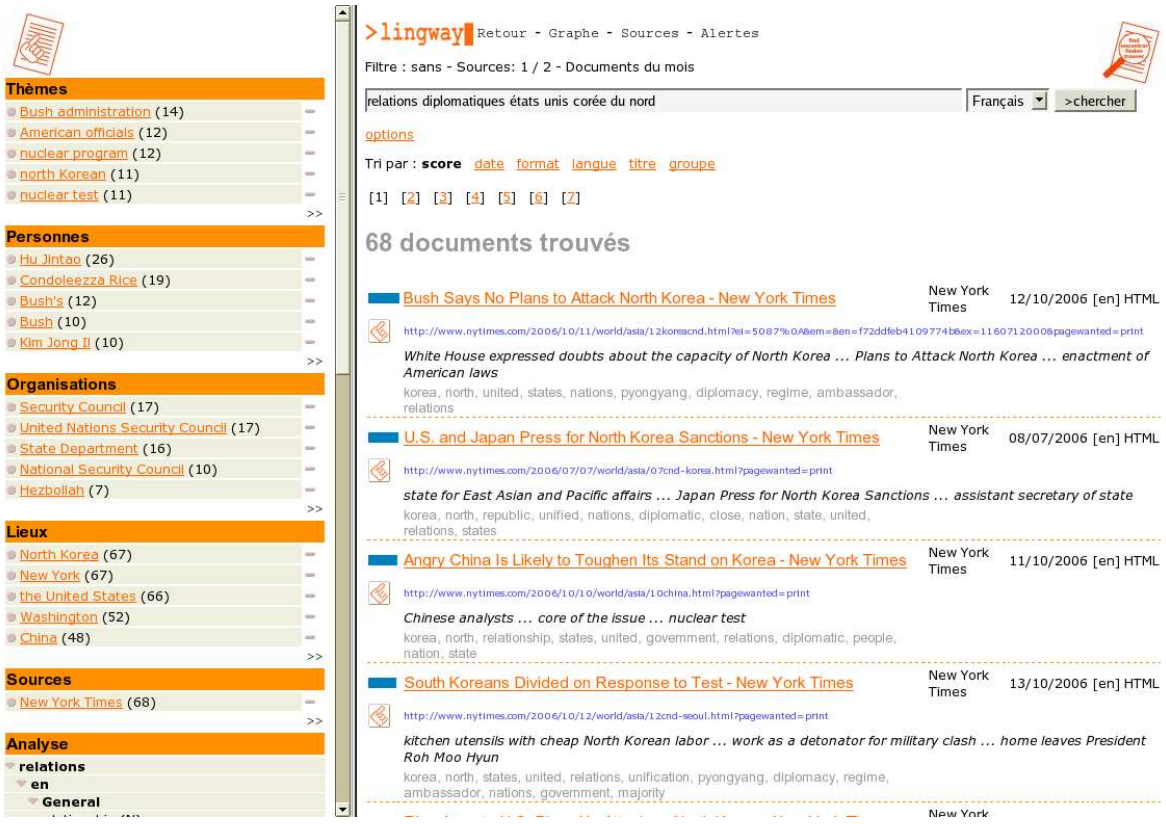


FIG. 2 – « relations diplomatiques États Unis Corée du nord »

1.2 Indexation et catégorisation

L'indexation et la catégorisation sont basées sur une structuration XML permettant le marquage de la structure générale d'un texte, l'extraction d'entités nommées (personnes,

organisations, produits, lieux, etc.), le marquage des termes importants, et des phrases importantes sous un point de vue donné.

1.2.1 Balisage de la structure

Ce module reconnaît les titres, sous-titres, sections, paragraphes et phrases, et produit une version XML du document avec des balises correspondant à la structure du document.
Cette reconnaissance n'est pas uniquement basée sur les balises du document en entrée (<title>, <h1>, etc.) ou autres marques de formatage, mais aussi sur la position, le format des caractères, et un certain nombre d'autres éléments.

1.2.2 Extraction des entités nommées

Lingway KM permet d'extraire certaines entités nommées :

- personnes (en isolant nom, prénom et fonction quand cela est possible) ;
- lieux (typés en pays, villes, régions, etc.) ;
- organisations (avec extraction du nom complet et du sigle quand cela est possible) ;
- dates et autres marqueurs temporels.

Des jeux de règles plus spécifiques peuvent rapidement être développés pour des besoins plus précis. A titre d'exemple, ont été développées des règles de marquage de :

- types et caractéristiques de matériels informatiques ;
- niveau de formation ;
- événements scientifiques et techniques (workshops, conférences, ateliers,...) ;
- autres événements (culturels, sportifs, ...).

1.2.3 Extraction des descripteurs thématiques (termes)

Un extracteur de terminologie retrouve les termes présents dans le document sur la base d'une analyse linguistique (structures Nom-Adjectif, Nom-Préposition-Nom, ...), etc. Ces termes deviennent des descripteurs potentiels du contenu du document. Le choix effectif des termes retenus comme descripteurs associés au document se fait sur la base de critères statistiques complémentaires.
Cette extraction statistique repose sur une première phase d'analyse linguistique du document où les mots simples sont étiquetés morphologiquement, les mots composés sont reconnus et le texte est découpé en bloc appelés « chunks » considérés comme des unités syntaxico-sémantiques constituant des « termes candidats » que la phase statistique sélectionnera ou éliminera. La phase de chunking repose sur les mêmes fonctionnalités Outilex que la reconnaissance des mots composés, elle est décrite à la section 3 page 11.

Ces descripteurs sont validables via l'interface standard. Les descripteurs retenus sont automatiquement considérés comme une indexation du document sous forme de méta-données.

1.3 Aide à la lecture

Les fonctions d'aide à la lecture permettent d'appréhender rapidement soit le contenu d'un document donné (fonctions de résumé et de colorisation du texte), soit le contenu d'un ensemble de documents trop important pour être lu intégralement (fonctions de clustering et de cartographie).

2 Module Outilex de reconnaissance des mots composés

La reconnaissance des mots composés lors de l'analyse de la requête est un point essentiel participant à l'amélioration de la qualité du moteur : par exemple, dans la requête « politiques économique et agricole européennes » l'analyseur détecte la présence du mot composé « politique agricole » et l'étend sémantiquement vers « PAC » et « politique agricole commune » (figure 3 en bas à gauche).
Cette reconnaissance « floue » des composés constitue la principale contribution d'Outilex à LKM.

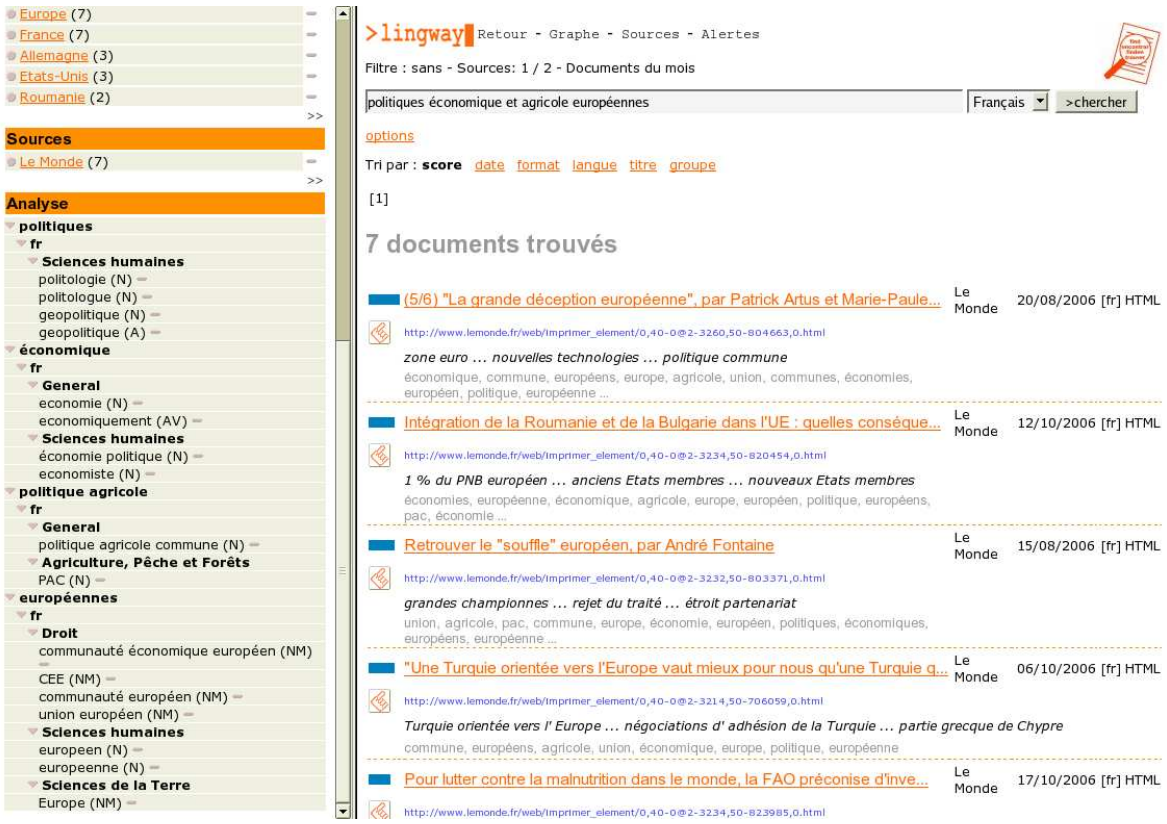


FIG. 3 – « politiques économique et agricole européennes »

2.1 Définition

La notion de « mot composé » est une notion assez vague en traitement des langues. Un mot composé est une suite de mots simples (peut-être vaudrait-il mieux parler de « tokens » à ce niveau) qui forment une unité de sens lorsqu'ils sont lus ensemble. En suivant cette définition, n'importe quelle phrase peut prétendre au statut de mot composé.

D'un point de vue du Traitement Automatique, il est nécessaire d'ajouter une contrainte opératoire à cette définition, traduisant le fait que l'unité de sens en question est tangible et exploitable par une machine. On dira donc qu'un mot composé est une suite d'éléments lexicaux, enregistrée dans un dictionnaire (les raisons de cet enregistrement ne sont pas du ressort de cette spécification).

Si le mot composé est enregistré dans un dictionnaire, se pose la question de le reconnaître dans un énoncé donné. Il est d'usage de dire qu'il existe un continuum dans les comportements de composés, entre le composé figé, qui n'admet aucune insertion, modification de l'ordre des composantes, et peu voire pas du tout de variations morphologiques d'une part, et les composés terminologiques pour lesquels on rencontre toutes sortes de variations.

2.2 Types de variations

La reconnaissance des composés figés (de type « pomme de terre », « garde-barrière », etc.) ne sont pas du ressort de la présente spécification, même si un certain nombre des mécanismes décrits plus loin peuvent leur être appliqués. On s'attache plutôt à définir les outils permettant la reconnaissance de composés dits « terminologiques », c'est-à-dire admettant tout ou partie des variations suivantes :

- omission de certains mots (déterminants, prépositions) : « direction ressources humaines » → direction_des_ressources_humaines
- factorisation : « antenne hertzienne ou parabolique » → antenne_parabolique ou antenne_hertzienne,
- insertion : « chariot électrique de golf » → chariot_de_golf électrique
- modification de l'ordre : « enlargement of the liver » → liver_enlargement
- dérivation : « hepatic enlargement » → liver_enlargment

2.2.1 Imbrication des composés

L'objet du document étant le processus de reconnaissance, nous ne nous intéresserons pas aux phénomènes d'imbrication de composés, comme dans l'exemple « direction des ressources humaines », où ressources_humaines est à soi seul un composé. Le statut de ce dernier composé dans un composé plus vaste (« direction des ressources humaines ») relève de la lexicographie ou plus pragmatiquement de la construction du dictionnaire, pas de son utilisation.

2.2.2 Représentation des types de variation

Formellement, chaque composé peut présenter tel ou tel type de variation, dans des mesures différentes. Pour des besoins de simplification et de représentation des données lexicales,

nous adopterons la partition suivante en distinguant deux types de composés :

- les composés figés, qui sont reconnaissables à l'identique (moyennant des variations morphologiques),
- les composés dits terminologiques, reconnaissables moyennant certaines approximations.

Ces approximations concernent une certaine classe de mots-outils qui peuvent se trouver dans la chaîne d'entrée ou dans le composé et sont " transparents " vis-à-vis du processus de reconnaissance (c'est-à-dire qu'ils ne sont pas pris en compte), et moyennant des transformations lexicales autorisées par deux indicateurs booléens sur le composé :

- ORDRE : l'ordre des composantes (dans le dictionnaire) doit-il être respecté lors de la reconnaissance ?
- SLEX : des substitutions lexicales peuvent-elles être effectuées lors du processus de reconnaissance (utilisation de dérivation nom/adjectif, ...) ?

Ces indicateurs sont globaux sur le composé et ne permettent pas de moduler l'indication des types de transformation ou du composant sur lequel ils peuvent agir. Il s'agit évidemment d'une représentation grossière mais qui, d'expérience, permet de représenter la majorité des phénomènes.

Le tableau ci-contre donne un exemple de transformation entre la chaîne d'entrée et le composé reconnu pour chacun des cas de figure :

	ORDRE = oui	ORDRE = non
SLEX = non	« direction ressources humaines » → direction_des_ressources_humaines	« enlargement of the liver » → liver_enlargement
SLEX = oui	« salaire encadrement » → salaire_des_cadres	« activité du volcan » → volcan_actif

2.2.3 Types d'éléments « transparents »

Les éléments qui ne sont pas pris en compte lors de la reconnaissance de composés teerminologiques sont :

- les déterminants,
- les prépositions à sémantique faible, telles que « de », « à », « pour ».

Ce processus est considéré comme global et applicable à tous les composés terminologiques.

2.2.4 Types de transformation

Les transformations que peut subir une chaîne d'entrée pour permettre la reconnaissance terminologique peuvent être de tous ordres, même si, le plus souvent, il s'agit :

- de décoordination de dépendants,
- d'extraposition d'un ou plusieurs mots (adjectifs ou adverbes la plupart du temps).

Ce processus est fortement dépendant à la fois du contexte et aussi de la présence ou non de composés terminologiques à reconnaître. En d'autres termes, les transformations ne seront considérées comme licites que si elles ont permis la reconnaissance d'un composé. Il est donc nécessaire de pouvoir exprimer ces transformations dans un système à base de règles, qui soit dirigé par les données, comme présenté dans la section suivante.

2.2.5 Substitutions lexicales

La substitution d'un élément par un autre implique qu'il existe une base de données de formes sémantiquement proches qui puisse servir de ressource à ce processus. Nombre de bases de données lexicales (CELEX, WordNet, Frantext, etc.) fournissent des réseaux lexicaux qu'il est possible d'utiliser pour un tel système. Nous considèrerons simplement ici qu'il existe une telle base.

2.3 Règles de transformation

Les règles sont exprimées comme réécritures d'une partie de la séquence. On peut donc les formaliser comme des transducteurs. Le principe est de proposer, via un opérateur donné, la confrontation d'une suite d'éléments au dictionnaire de termes. Si le terme est présent, la règle réussit et la réécriture est effectuée ; sinon elle échoue.
Les règles sont donc des motifs d'expressions régulières pour lesquelles l'alphabet est constitué des éléments issus de l'analyse morphologique, sous la forme `forme . lemme : catégorie : traits` (par exemple `tables.table:N:NH`)

2.3.1 Règle simple

Soit la séquence « carte internationale de crédit », analysée en

- carte :N
- internationale :A
- de :P
- crédit :N

La règle permettant de tester la reconnaissance du terme carte_de_crédit sera :

```
(_N_) (_A_) _P_ (_N_)
-> (1,3) 2
```

Elle se lit : si un terme peut être reconnu à partir des captures 1 et 3 du motif en première ligne, alors réécrire ce terme suivi de l'élément contenu dans la capture 2. Les symboles _ sont des « wildcards » pour les parties non-spécifiées des tokens (forme, lemme, traits). Notons que la préposition n'est pas considérée comme significative et n'a donc pas été reprise dans le test de termes. L'opérateur `( )` de reconnaissance de termes met en œuvre les mécanismes évoqués ci-dessus, à savoir :

- masquage des éléments « transparents »,
- substitutions lexicales si le composé atteint permet cette opération,
- modification de l'ordre des composantes (même type de conditions)

2.3.2 Règle multiple

Dans la mesure où les règles sont déjà de la forme « si...alors », une extension simple du formalisme est de prévoir des alternatives ordonnées à la reconnaissance (de type « si..alors..sinon »), comme la règle suivante qui permet de traiter les séquences de type N et N de N.

```
 (_N_) (_CC_) (_N_) (_P_) (_N_)
->(1,5) 2 (3,5)
->(1,5) 2 3 4 5
->1 2 (3,5)
```

Par exemple, sur une entrée « fabricant et réparateur de cycles », la règle va tester l'existence des composés fabricant_de_cycles et réparateur_de_cycles, puis en cas d'échec, du premier de ces termes (deuxième réécriture) en considérant la factorisation. Enfin, en cas de nouvel échec, du second, mais sans traiter la séquence comme une factorisation.

2.3.3 Interaction avec la syntaxe locale

Ainsi que le montre clairement ce dernier exemple, les règles exprimées sont très proches de celles des grammaires locales. De plus, la confrontation à la terminologie permet d'élaborer des hypothèses sur la syntaxe locale de la phrase, hypothèses qu'il peut être souhaitable de conserver pour la suite de l'analyse (dans l'exemple précédent, présence ou non de factorisation). C'est pourquoi on adjoint au formalisme celui de pouvoir créer des « chunks », c'est-à-dire des groupes de mots marqués d'une catégorie. Ces groupes sont considérés comme des îlots de confiance pour un traitement ultérieur. Ce « chunking » est effectué au moyen d'un opérateur `[ ]`, suivi de la catégorie morpho-syntaxique du chunk à créer, et englobant les éléments capturés. Ainsi, la précédente règle pourra être réécrite en :

```
(_N_) (_CC_) (_N_) (_P_) (_N_)
->(1,5) 2 (3,5)
->(1,5) 2 [NP 3, 4, 5]
->[NP 1] 2 (3,5)
```

3 Chunking

Le chunking constitue la première phase de l'extraction des thèmes par LKM. La deuxième phase, statistique, sélectionne les thèmes les plus pertinents pour chaque document. Sur l'exemple de la figure 4, la colonne de gauche contient les thèmes jugés les plus pertinents pour la sous-collection de documents sélectionnée grâce à la requête « Liban et ONU » .
Ces termes peuvent aussi être exploités dans l'outil graphique de clustering de termes où un graphe des entités nommés et des termes pertinents pour la sous-collection de documents représentes par des arcs plus ou moins long le degré de coocurrence desdits termes dans la collection (figure 5 page 13).

3.1 Définition

Le « chunking » est un traitement linguistique intervenant après la segmentation en phrase et destiné à découper une phrase en blocs constituant des unités sémantiques atomiques en

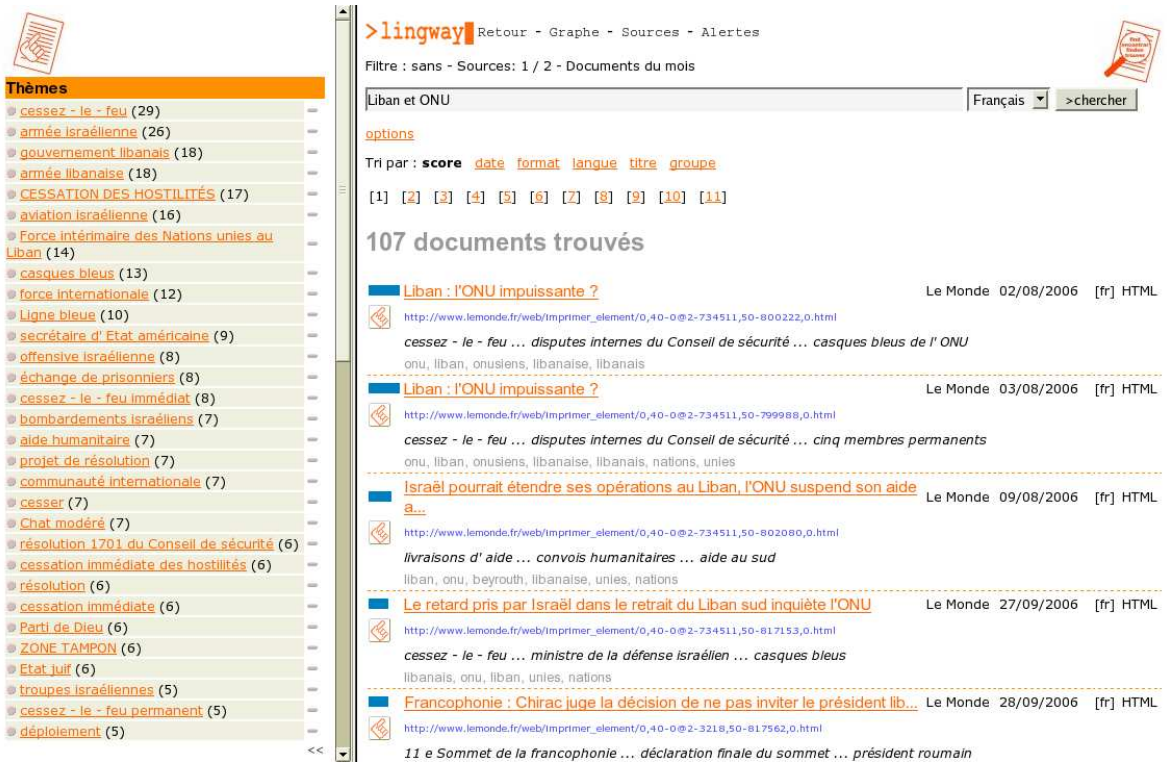


FIG. 4 – « Liban et ONU »

se basant sur les propriétés morphologiques et syntaxiques des mots. Il s'agit principalement de reconnaître des groupes nominaux (du type adjectif + nom), verbaux (verbe + adverbe) ou adjectivaux. Pour nous, le but est de constituer une liste de termes susceptibles d'être sélectionnés par la phase statistique d'extraction des thèmes et donc d'être considérés comme représentatifs du document analysé.

3.2 Règles

De même que les mots composés, les chunks sont reconnus par des motifs syntaxiques et ajoutés à l'automate d'analyse grâce à des règles de réécriture. À la différence des règles des mots composés, aucun accès à un dictionnaire n'est effectué, les blocs sont ajoutés à l'automate s'ils ne se chevauchent pas et en respectant l'ordre de priorité, en l'occurrence l'ordre de déclaration.

Par exemple, la règle suivante nommée `NP` (*NounPhrase*) reconnaît les groupes nominaux et les ajoute à l'automate avec l'étiquette `CHUNK` :

```
::NP
--(_D_? _NB_|_D_ _NB_?|_P_ _D_? _NB_?) (_A_* (_N_|_NM_) (_N_|_A_|_NM_)*)
>>[1] {CHUNK 2}
```

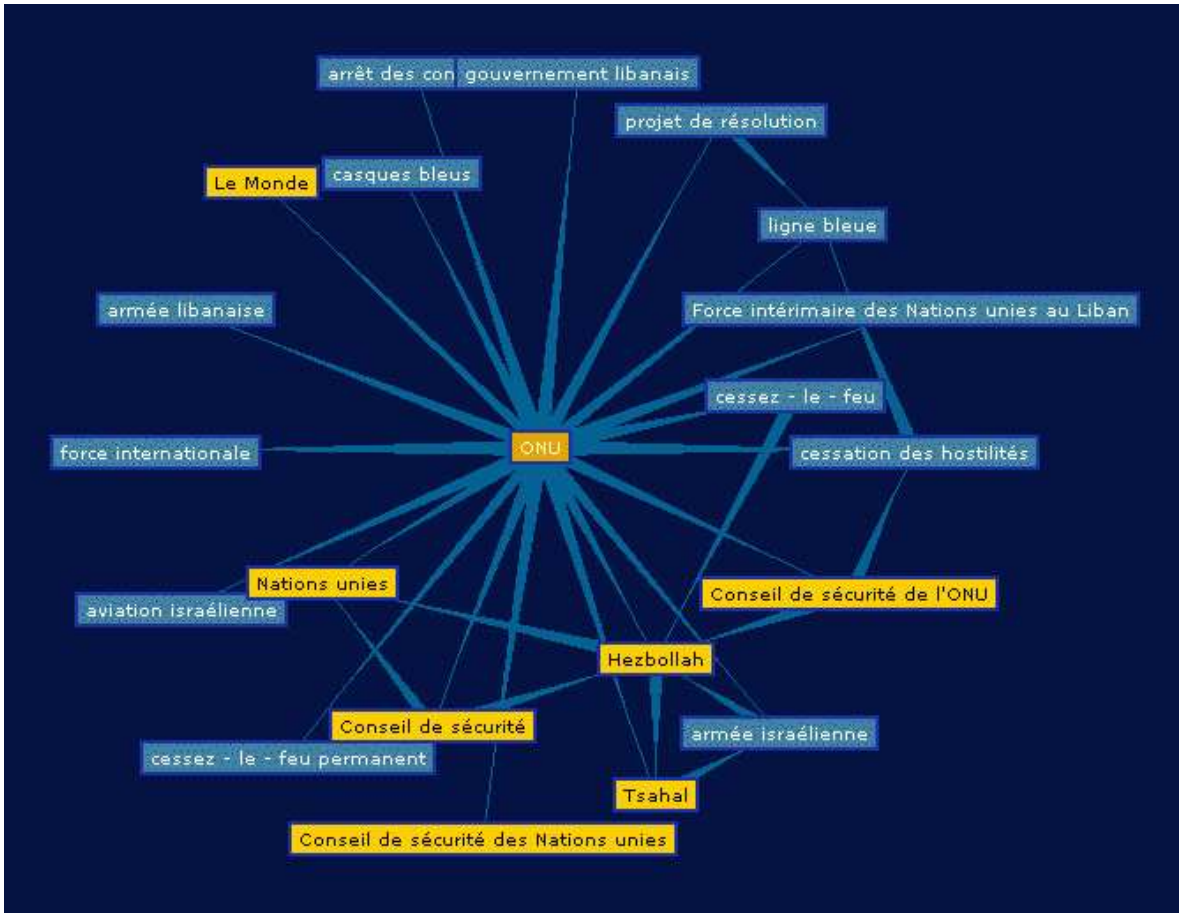


FIG. 5 – « Liban et ONU », clustering de termes

Appliquée à la première phrase de cette section, cette règle marque les blocs :

- *les mots composés*
- *les chunks*
- *des motifs syntaxiques*
- *l'automate*
- *analyse*
- *des règles*
- *réécriture*

Les réécriture se composent, c'est-à-dire qu'une règle peut définir un motif syntaxique contenant des références à des blocs ajoutés par une règle précédente, permettant ainsi par inclusions successives de constituer des termes plus intéressants que des mots simples ou des mots composés de base comme « élargissement du conseil de sécurité » ou « opération de maintien de la paix ».

4 Annexe : mise en œuvre

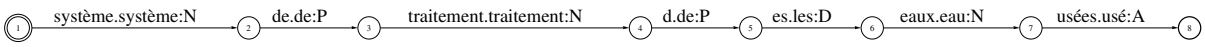
4.1 Introduction

La suite de ce document aborde la mise en œuvre du module de reconnaissance de composés dits terminologiques (par opposition aux composés figés) admettant un certain nombre de variations linguistiques (entre autres : omission de composants, factorisation, insertion, modification de l'ordre, dérivation) les rendant plus difficilement reconnaissables.

Le but est d'obtenir une reconnaissance de composés souple dirigées par des règles comprenant des motifs morphologiques à reconnaître et des mécanismes de réécriture contraints par le dictionnaire des composés. Le but est aussi d'obtenir une implémentation efficace en temps et en espace, c'est pourquoi le langage utilisé est le C++.

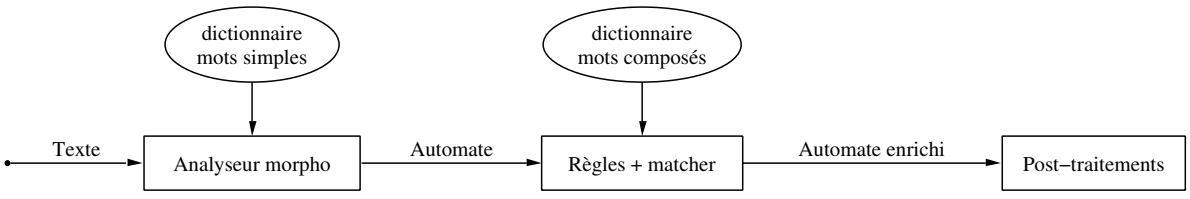
4.2 Principe

Les résultats de l'analyse morphologique se présentent sous forme d'automate fini déterministe étiqueté par des mots. Ci-dessous le résultat de l'analyse de la phrase "système de traitement des eaux usées" :



Les transitions portent des triplets (forme fléchie, forme canonique, catégorie morphologique). La reconnaissance des composés qui suit ajoute des interprétations possibles à la phrase, c'est-à-dire de nouveaux chemins dans l'automate.
Sur la figure 6, la reconnaissance des mots composés "traitement de l'eau" et "eau usée" ajoute deux transitions à l'automate de départ.
L'automate est prêt à subir les post-traitements classiques qui suivent l'analyse morphologique : sélection d'une interprétation particulière grâce à un algorithme du meilleur chemin, analyse syntaxique, analyse sémantique, etc.

4.3 Architecture générale



Ce module correspond à la boîte « Règles + matcher », il inclut le parser/compilateur de règle et le « matcher », c'est-à-dire le composant responsable du calcul de l'intersection entre

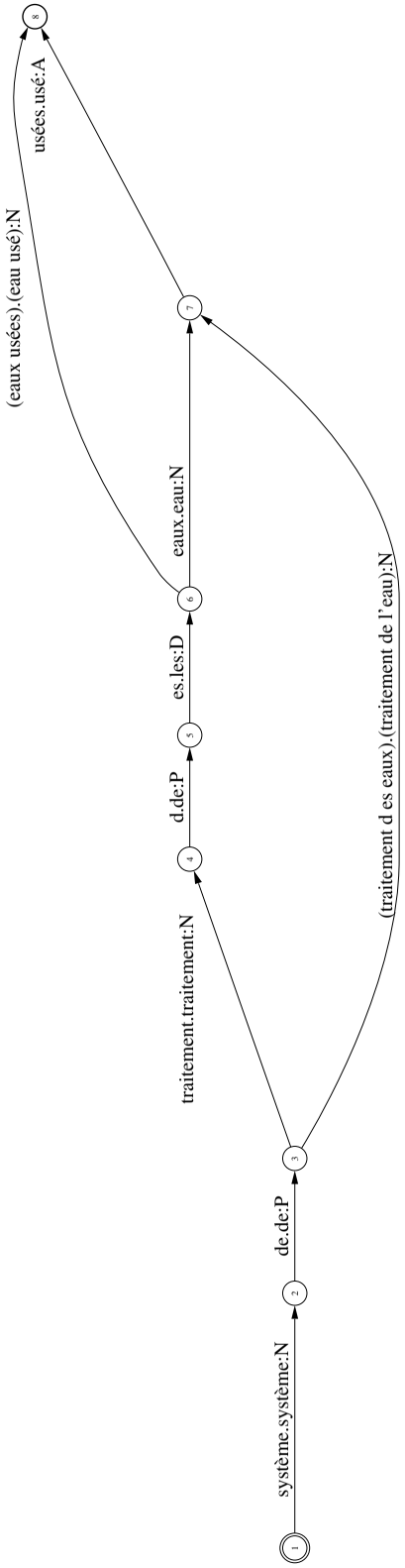


FIG. 6 – L'analyse après reconnaissance des mots composés

l'automate d'analyse et ceux des règles mais il ne comprend pas l'implémentation du dictionnaire des mots composés qui est externe et doit être branché au matcher par l'utilisateur du module.

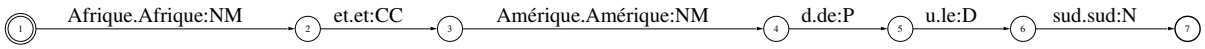
4.4 Exemples

4.4.1 Factorisation des composants

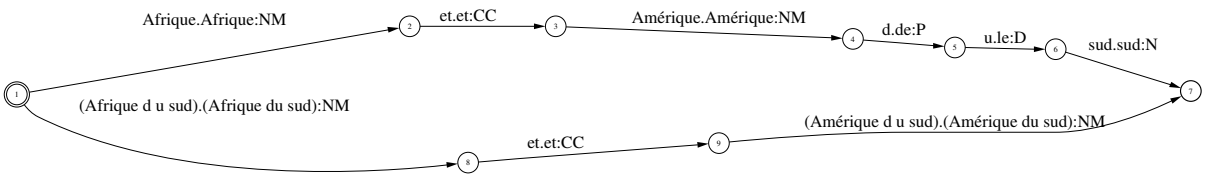
Dans la règle suivante, on définit un motif (ligne commençant par `--`) reconnaissant deux mots composés du type *Nom Prep Nom* (les noms propres sont autorisés grâce à l'étiquette morphologique `NM`) et réalisant leur « dé-coordination » grâce à des réécritures (lignes commençant par `->`) :

```
::Exemple Outilex Factorisation
--(_N_|_NM_) (_CC_) (_N_|_NM_) (_P_) (_D_)? (_N_|_NM_)
->(1,4,5,6) [2] (3,4,5,6)
->(1,6) [2] (3,4,5,6)
->(1,6) [2] (3,6)
->(1,4,5,6) [2] (3,6)
->(1,6) [2] [3] [4] [5]
->[1] [2] (3,6)
```

La phrase suivante :



est reconnue par le motif. Elle est réécrite en regroupant les mots différemment avant l'accès au dictionnaire des mots composés : par exemple `(1,4,5,6)` correspond à tester si le mot (Afrique :NM, de :P, le :D, sud :N) appartient au dictionnaire. Si une des réécritures « fonctionne », c'est-à-dire, si le ou les mots composés ainsi formés existent, les transitions définies par ces regroupement sont ajoutées à l'automate. Ici, c'est la troisième réécriture qui autorise l'enrichissement de l'automate comme suit :



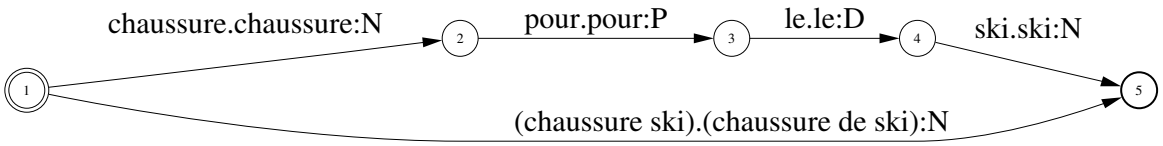
Les mots réécrits sous la forme `[x]` le sont tels qu'ils apparaissent à l'origine, c'est-à-dire même forme, lemme et étiquette morpho/syntaxique.

4.4.2 Élements transparents

Cet exemple illustre le fait que l'accès au dictionnaire des composés « à la préposition près » permet une reconnaissance souple diminuant le silence (occurrences ratées) tout en entraînant un minimum de bruit (occurrences erronées) car des contraintes fortes sont imposables sur les atomes du motif.

Dans l'expression suivante, les prépositions sont complètement spécifiées (à, de, pour) et servent de garde-fou à l'accès au dictionnaire :

```
::Exemple Outilex Eléments Transparents
--(_N_) (à:P_|de:P_|pour:P_) (_D_)? (_N_)
->(1,4)
->><4,1>
```

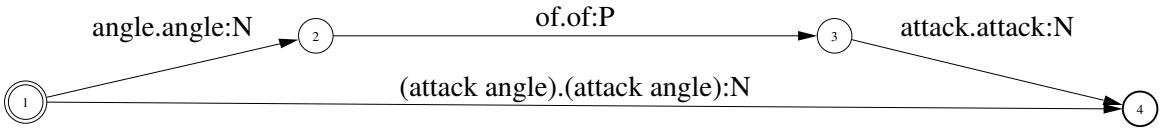


4.4.3 Permutation des composants

L'opérateur `< >` permet un accès au dictionnaire des composés vérifiant, outre la présence du mot, la validité de la permutation des composants. Il trouve son utilité par exemple en anglais, où la plupart des mots composés du type *Nom Prep Nom* ont des composants permutables :

```
::Exemple Outilex Permutations
--(_N_) (_P_)? (_N_)
->(1,3)
->><3,1>
```

Sur l'exemple suivant, le dictionnaire ne contient que la forme *attack angle* mais la deuxième réécriture `<3,1>` permute les deux noms avant l'accès et permet ainsi la reconnaissance du composé :



4.5 ASTL

Ce module repose sur ASTL (Automata Standard Template Library) une librairie C++ de composants génériques de manipulation d'automate sous licence Lesser GPL (disponible

sur `http://astl.sourceforge.net`). Elle est basée sur STL, la librairie standard de patrons du C++, et utilise les techniques de programmation générique par template les plus efficaces. Elle fournit les fonctionnalités bas-niveau de gestion des containers d'automates (automates d'analyse et de règle, dictionnaire des composés) et de reconnaissance des expressions rationnelles (compilation des règles et intersection entre analyse et règle).