TagParser et Technolangue-Easy

Gil Francopoulo
Tagmatica
126 rue de Picpus 75012 Paris
gil.francopoulo@wanadoo.fr

Mots-clés : chunking, analyse syntaxique à large couverture
Keywords: chunking, large scale coverage syntactic parsing

Résumé TagParser est une chaîne d'analyse à stratégie montante dont les ressources ont été conçues afin de permettre un développement incrémental de la grammaire. TagParser combine TagChunker avec un module de calcul de relations syntaxiques.

Abstract TagParser is a 'bottom up' parser whose linguistic resources have been created in order to allow an incremental development. TapParser combines TagChunker with a module for syntactic relations computation.

1 Introduction

Nous présentons ici l'analyseur que nous avons utilisé pour participer au programme d'évaluation Technolangue-Easy organisé par le CNRS-LIMSI et ELDA sous l'égide du Ministère de la Recherche. TagParser est un analyseur de français[1] robuste, rapide et à large couverture. Il appartient à la famille des analyseurs dits « montants » dans le sens où les éléments constitutifs de l'analyse sont agrégés de proche en proche et par strates successives afin de former un résultat syntaxique.

2 Démarche

Notre démarche n'est pas purement scientifique mais plutôt technologique. Il ne s'agit pas d'établir ou de tester une théorie. Il s'agit d'élaborer le meilleur système en tenant compte des paramètres suivants :

- Qualité du résultat

[1] La version anglaise est en cours de développement selon les mêmes principes.

- Rapidité d'analyse
- Coût de développement
- Coût de la maintenance
- Couverture et robustesse

Notons que la couverture et la robustesse (bien que quelque fois confondues) sont, d'après nous, des critères légèrement différents même s'ils sont liés. La couverture, c'est le nombre de phénomènes linguistiques que le programme est capable d'analyser. Dans un système qui applique plusieurs stratégies d'analyse, on distingue la couverture globale de la couverture de chacune des stratégies d'analyse. La robustesse, c'est la faculté du système de pouvoir choisir une stratégie de remplacement quand un échec avec une certaine stratégie a été détectée.

De plus, TagParser n'est pas conçu avec des « bouts de ficelle » mais selon la conception objet et le développement itératif, avec un emploi généralisé d'UML et de Java. Tout en étant innovant, ce n'est pas un prototype de laboratoire.

3 Evaluation

Il est tentant de « sur-simplifier » l'évaluation d'un système au couple précision / rappel. En fait, ce couple ne fournit qu'une indication quantitative qu'il faut prendre pour ce qu'elle est : c'est-à-dire un couple d'indicateurs, rien de plus.

Loin de nous l'idée de dénigrer la campagne d'évaluation Technolangue-Easy. Elle a le mérite pour la première fois en France, en plus du calcul de la précision-rappel de 18 analyseurs :

- D'avoir permis la confrontation intellectuelle de toutes les équipes françaises du domaine lors des réunions de la mise en place de la campagne d'évaluation[2] ;
- D'avoir motivé de nombreux développements ;
- De fournir un guide d'annotation ;
- De fournir la première version d'un corpus annoté, bien qu'à ce propos, il semble assez dangereux de le figer une fois pour toute car cela aurait pour biais que les développeurs d'analyseurs se focalisent sur le corpus annoté plutôt que sur la tâche prise dans son ensemble. Dans cette optique, le corpus annoté devrait plutôt être considéré comme un ensemble de départ qui serait augmenté (ou modifié) périodiquement, plutôt que comme un corpus de référence immuable.

Mais d'autres critères sont tout aussi intéressants pour évaluer un système. L'analyseur n'est pas un but en soi : **il doit rendre des services à des utilisateurs et pouvoir évoluer dans un**

[2] Personnellement, nous avons beaucoup appris lors de ces réunions.

contexte industriel. Que vaut le couple précision-rappel si l'analyse d'une phrase prend plus de 5 minutes ? Que vaut ce couple si le coût d'évolution et de maintenance est prohibitif ?

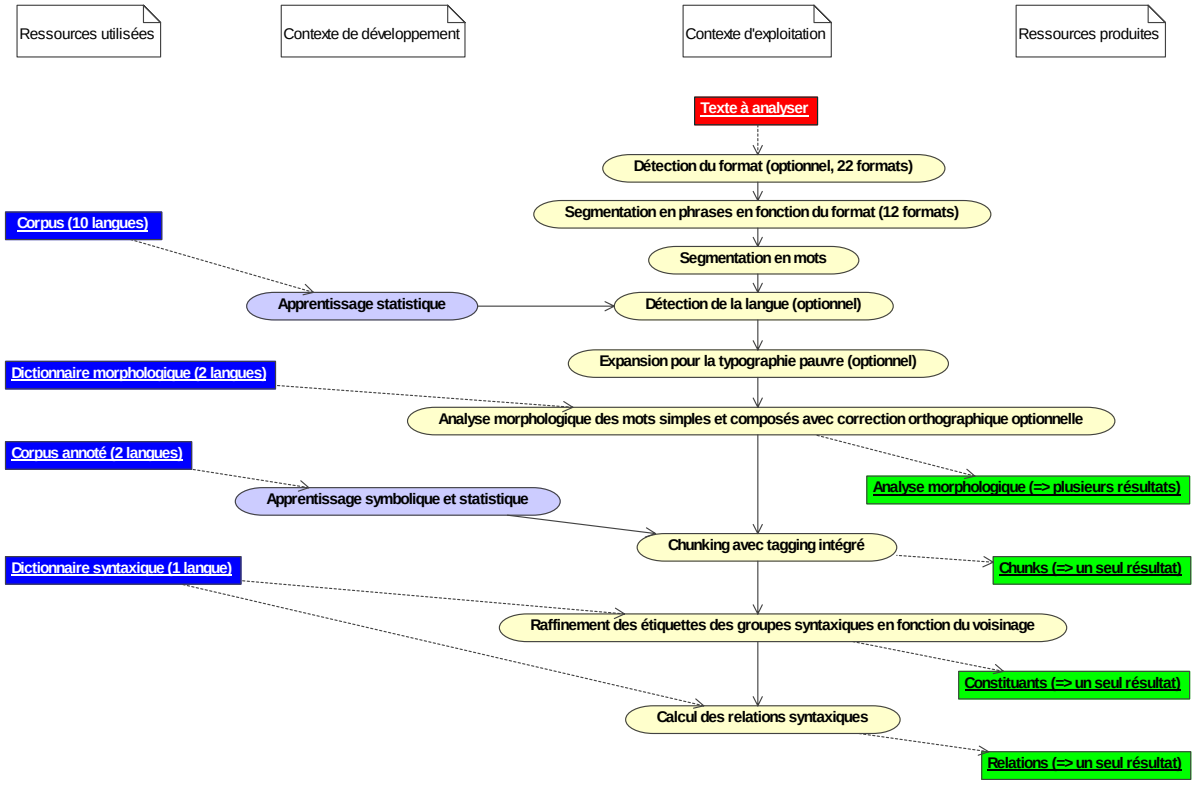
4 Chaîne d'analyse

4.1 Présentation

TagParser est une chaîne d'analyse complète associant segmenteur, analyseur morphologique, chunker et calculateur de relations syntaxiques. Un certain nombre d'outils auxiliaires peuvent lui être associés selon les circonstances comme la détection automatique des formats (est-ce un fichier Word ou HTML ?), la détermination de la langue, la correction orthographique ou le traitement de la typographie pauvre (absence d'accent et de casse).

La chaîne d'analyse prend un texte en entrée et produit un résultat syntaxique au format PEAS [Gendner]. Les deux parties les plus importantes sont le chunker [Francopoulo] et le calcul des relations syntaxiques.

4.2 Diagramme d'activité UML



4.3 Chunking

4.3.1 Objectif

4.3.1.1 étiquetage des parties du discours (i.e. Part of Speech Tagging)

Par exemple, dans les deux phrases suivantes : "Le comité d'experts table sur une croissance de 1%." et "La table est à 3 mètres du mur." Il s'agit, pour le mot "table", de le catégoriser en tant que verbe dans la première phrase et en tant que nom dans la seconde.

4.3.1.2 délimitation des groupes syntaxiques

Il s'agit de fixer les frontières des groupes syntaxiques. Ainsi, dans le premier exemple, le découpage suivant est construit : [Le comité][d'experts][table][sur une croissance][de 1%].

4.3.1.3 étiquetage des groupes

Les groupes constitués sont étiquetés et produisent le résultat suivant : GN GP GV GP GP.

4.3.2 Stratégie d'analyse

Le chunker appartient à la famille des analyseurs dits "hybrides" qui combinent des informations symboliques (via un dictionnaire et un automate) et un modèle probabiliste (via une matrice de pondération). La stratégie d'analyse repose sur trois principes simples : a) une analyse par automate est tentée, b) si elle échoue, des fragments d'automate sont combinés et tentés, c) quand des solutions multiples sont trouvées, la "meilleure" solution est choisie en fonction de la matrice de pondération.

Au lieu d'opposer les méthodes symboliques aux méthodes statistiques, nous les combinons pour en tirer le meilleur de chacune d'elle. Autrement dit : quand la situation est prévue lors du développement, on l'applique sans autre forme de procès, car elle correspond à une situation totalement maîtrisée par le développeur de l'analyseur. En revanche, si la situation n'est pas prévue, il faut faire appel à des variables cachées que seul un corpus est capable de procurer. On observe d'ailleurs que la frontière entre les deux méthodes varie en fonction de l'état de développement du système. Ainsi, pour un système peu développé, une méthode probabiliste est parfaitement adéquate. Mais les performances en terme de qualité plafonnent assez vite, même en raffinant le modèle mathématique. Si la part des connaissances symboliques s'accroît, plus la méthode symbolique est applicable car plus les situations sont prévues à l'avance, et de ce fait, moins les pondérations sont nécessaires.

Les informations linguistiques qui pilotent l'analyse sont issues d'un dictionnaire et d'un entraînement sur un corpus. **Aucune règle de grammaire n'est jamais écrite**. Si une phrase pose problème, il suffit de l'ajouter avec son annotation dans le corpus d'apprentissage, puis de relancer le programme d'apprentissage. Ce dernier vérifie d'une part que la phrase annotée ne contredit pas les annotations pré-existantes et d'autre part produit un nouvel automate et une nouvelle matrice de pondération. Son développement est de ce fait complètement incrémental. Il est de plus facilement adaptable à un certain type de texte en fonction d'une application particulière. Vous trouverez les détails dans l'article publié au congrès TALN-2003. Le chunker a été écrit en 2002 (et employé dans diverses applications) et les principes

de base exposés en 2003 sont toujours valables, mais depuis lors, le corpus d'apprentissage est passé de 18 000 à 34 000 mots et sa couverture améliorée d'autant.

4.4 Raffinement des étiquettes syntaxiques

Il reste quelques problèmes localisés à résoudre. Ainsi, par exemple, un certain type de groupe qui commence par "du" en français est ambigu entre GN ou un GP. Il n'est pas possible en se fondant uniquement sur les constituants de déterminer son étiquette. Prenons par exemple les phrases: "Il fait du ski" comparativement à "il arrive du ski". Dans ce cas précis, le chunking marque le groupe comme étant "indéterminé entre GN et GP". L'indétermination est levée en croisant deux types d'information : le voisinage du groupe en question et un dictionnaire syntaxique. Une petite grammaire désambiguïsation a été écrite explicitement dans ce but. Dans toutes les autres circonstances où il possible de déterminer si l'on a affaire à un GN ou un GP, l'étiquette est fixée dans la phase de chunking.

4.5 Calcul des relations syntaxiques

Le chunking produit un seul résultat constitué d'une liste de groupes syntaxiques (i.e. liste de constituants). Il s'agit maintenant de calculer les relations qui font que ces groupes occupent certaines fonctions dans la phrase. Au contraire du chunker, qui est dérivé d'un corpus annoté, les règles de grammaire sont écrites explicitement. En effet, l'apprentissage à partir d'un corpus de taille moyenne (i.e. de l'ordre de quelques milliers de mots) se prête mal aux relations syntaxiques. Il faudrait un corpus de l'ordre du million de mots pour en capter la généralité. Les relations sont catégorisées en 14 types comme Sujet-Verbe, Modifieur-Nom, Coordination décrites dans [Gendner]. Ces relations sont subdivisés en relations externes quand elles connectent des groupes entre eux et en relations internes quand elles connectent des mots à l'intérieur d'un même groupe. Le calcul des relations est en réalité l'application en séquence de 14 grammaires locales spécialisées pour tel ou tel type de relation. Ces grammaires exploitent deux critères différents: d'une part, une description (sommaire, il est vrai actuellement) du régime verbal représenté dans le dictionnaire syntaxique et d'autre part la position linéaire du groupe relativement aux autres groupes dans la phrase.

5 Conclusion

TagParser est un logiciel qui a été construit par étapes. Une fois le chunker élaboré et testé, avec l'objectif d'extraire des termes techniques, le besoin s'est fait sentir de distinguer les actants du verbe des modifieurs du verbe, dans cette optique le calcul des relations syntaxiques a été ajouté.

Références

FRANCOPOULO G. (2003) TagChunker : mécanisme de construction et évaluation, Actes TALN.

GENDNER V., VILNAT A. (2004) Les annotations syntaxiques de référence PEAS V-1.6. www.limsi.fr/Recherche/CORVAL/easy/