Trouver le coupable : Fouille d'erreurs sur des sorties d'analyseurs syntaxiques

Benoît Sagot, Éric Villemonte de la Clergerie

Projet ATOLL, INRIA
{benoit.sagot ; eric.de_la_clergerie}@inria.fr

Résumé

Nous présentons une méthode de fouille d'erreurs pour détecter automatiquement des erreurs dans les ressources utilisées par les systèmes d'analyse syntaxique. Nous avons mis en œuvre cette méthode sur le résultat de l'analyse de plusieurs millions de mots par deux systèmes d'analyse différents qui ont toutefois en commun le lexique syntaxique et la chaîne de traitement pré-syntaxique. Nous avons pu identifier ainsi des inexactitudes et des incomplétudes dans les ressources utilisées. En particulier, la comparaison des résultats obtenus sur les sorties des deux analyseurs sur un même corpus nous a permis d'isoler les problèmes issus des ressources partagées de ceux issus des grammaires.

Mots-clés : analyse syntaxique, fouille d'erreurs.

Abstract

We introduce an error mining technique for automatically detecting errors in resources used in parsing systems. We applied this technique on parsing results produced on several millions of words by two distinct parsing systems, which share a common syntactic lexicon and pre-parsing processing chain. We were thus able to identify errors and missing elements in the resources. In particular, by comparing both systems' results, we were able to differentiate between problems stemming from shared resources and those resulting from grammars.

Keywords: parsing, error mining.

1. Introduction

L'analyse syntaxique des langues est une tâche complexe, en partie à cause de la richesse et du volume des informations à prendre en compte sur les mots et sur les constructions syntaxiques. Il est pourtant indispensable de disposer de ces informations, sous la forme de ressources telles que des lexiques et des grammaires, en essayant de minimiser les incomplétudes et les erreurs présentes dans ces ressources. L'utilisation à grande échelle de ces ressources dans des analyseurs est à ce titre très intéressante (van Noord, 2004), et en particulier l'étude des cas conduisant à un échec de l'analyse : comme le dit le dicton, on apprend de ses erreurs.

Nous proposons un modèle probabiliste permettant de repérer les formes potentiellement sources d'erreurs à partir d'un corpus de phrases soumis à analyse. Pour exploiter au mieux les formes détectées par le modèle et en particulier pour pouvoir identifier la source de l'erreur, un environnement de visualisation a été mis en place. L'ensemble a été testé sur les résultats d'analyse produits sur plusieurs corpus de plusieurs centaines de milliers de phrases et deux systèmes

d'analyse syntaxique distincts, à savoir FRMG et SXLFG.

2. Principes

2.1. Idée générale

L'idée que nous avons mise en œuvre est la suivante, inspirée de (van Noord, 2004). Pour identifier les incomplétudes et les incorrections d'un système d'analyse syntaxique, on peut analyser un corpus de taille conséquente et étudier à l'aide d'outils statistiques ce qui différencie les phrases pour lesquelles l'analyse a réussi de celles pour lesquelles elle a échoué.

L'application la plus simple de cette idée consiste à chercher les formes, dites *suspectes*, qui se retrouvent fréquemment dans des phrases qui n'ont pu être analysées. C'est ce que fait (van Noord, 2004), sans toutefois chercher à identifier une forme suspecte pour chaque phrase non analysable, et donc sans prendre en compte le fait qu'il y a une cause d'erreur dans toute phrase non analysable[1].

Au contraire, nous allons chercher, pour chaque phrase dont l'analyse a échoué, la forme qui a le plus de chances d'être la cause de cet échec : c'est le *suspect principal* de la phrase. Il se peut que cette forme soit renseignée dans le lexique de façon incorrecte ou incomplète, qu'elle participe à des constructions non couvertes par la grammaire, ou qu'elle illustre des imperfections de la chaîne de traitements pré-syntaxiques. Ce principe de base sur les formes s'étend naturellement aux lemmes et aux séquences de formes (ou de lemmes)[2].

2.2. Modèle probabiliste

On suppose donc que le corpus utilisé est découpé en phrases, elles-mêmes découpées en formes. On note p_i la i-ième phrase. On note $o_{i,j},\ (1 \leq j \leq |p_i|)$ les occurrences des formes constituant p_i, et on désigne par $F(o_{i,j})$ les formes correspondantes. Enfin, on note erreur la fonction qui à une phrase p_i associe 1 si l'analyse de p_i a échoué, et 0 sinon.

Soit $\mathcal{O}_f$ l'ensemble des occurrences d'une forme f dans le corpus : $\mathcal{O}_f = \{o_{i,j} | F(o_{i,j}) = f\}$. Le nombre d'occurrences de f dans le corpus est donc $|\mathcal{O}_f|$.

Définissons tout d'abord le *taux de suspicion moyen global* $\overline{S}$, qui est la probabilité moyenne qu'une occurrence donnée d'une forme soit la cause d'un échec d'analyse. Nous faisons l'hypothèse que l'échec de l'analyse d'une phrase est dû à une cause unique, et donc ici à une forme unique. Cette hypothèse, qui n'est pas nécessairement vérifiée, simplifie le modèle et donne des résultats pertinents. En notant $\mathrm{occ}_{\mathrm{total}}$ le nombre total de formes dans le corpus, on a donc :

$$\overline{S} = \frac{\Sigma_i \mathrm{erreur}(p_i)}{\mathrm{occ}_{\mathrm{total}}}$$

Soit une forme f qui est la j-ième forme de la phrase p_i, c'est-à-dire que $F(o_{i,j}) = f$. Supposons que l'analyse de p_i a échoué : $\mathrm{erreur}(p_i) = 1$. On appelle *taux de suspicion* de la j-ième forme $o_{i,j}$ de la phrase p_i la probabilité, notée $S_{i,j}$, que l'occurrence $o_{i,j}$ de la forme f soit responsable de l'échec de l'analyse de p_i. Si au contraire l'analyse de la phrase p_i a réussi, ses occurrences ont un taux de suspicion nul.

[1] Il définit taux de suspicion d'une forme f par le taux de phrases inanalysables parmi celles contenant f.

[2] Des expériences préliminaires incluant à la fois les formes et les séquences de deux formes ont donné des résultats très intéressants, montrant que certaines séquences causent une erreur sans que cela ne soit le cas pour leurs composants dans d'autres contextes.

On définit alors le *taux de suspicion moyen* S_f de la forme f comme étant la moyenne des taux de suspicion de ses occurrences (toutes phrases confondues) :

$$S_f = \frac{1}{|\mathcal{O}_f|} \cdot \sum_{o_{i,j} \in \mathcal{O}_f} S_{i,j}$$

Nous utilisons un algorithme de recherche de point fixe, en itérant un certain nombre de fois les calculs suivants. Supposons que l'on vient de terminer la n-ième itération : nous connaissons pour chaque phrase p_i et pour chaque occurrence $o_{i,j}$ de cette phrase l'estimation de son *taux de suspicion* $S_{i,j}$ par la n-ième itération, estimation notée $S_{i,j}^{(n)}$. On peut en déduire l'estimation de rang $n+1$ pour le taux de suspicion moyen de chaque forme f, noté $S_f^{(n+1)}$:

$$S_f^{(n+1)} = \frac{1}{|\mathcal{O}_f|} \cdot \sum_{o_{i,j} \in \mathcal{O}_f} S_{i,j}^{(n)}$$

Ce taux[3] nous permet de calculer une nouvelle estimation des taux de suspicion des occurrences, en attribuant à chaque occurrence d'une phrase p_i un taux de suspicion $S_{i,j}^{(n+1)}$ égal à cette estimation $S_f^{(n+1)}$ du taux de suspicion moyen S_f de la forme correspondante, puis normaliser à l'échelle de la phrase. Ainsi :

$$S_{i,j}^{(n+1)} = \mathrm{erreur}(p_i) \cdot \frac{S_{F(o_{i,j})}^{(n+1)}}{\sum_{1 \leq j \leq |p_i|} S_{F(o_{i,j})}^{(n+1)}}$$

À ce stade, la $n+1$-ième itération est terminée, et on peut recommencer à nouveau ces calculs, jusqu'à convergence sur un point fixe. L'amorçage de cet algorithme se fait en posant, pour une occurrence $o_{i,j}$ dans la phrase p_i, $S_{i,j}^{(0)} = \mathrm{erreur}(p_i)/|p_i|$. Autrement dit, si l'analyse de p_i a échoué, on part d'une estimation où toutes ses occurrences ont une probabilité égale d'être la cause de l'échec.

Après quelques dizaines d'itérations de ce processus, on obtient donc des estimations du taux de suspicion moyen de chaque forme, ce qui permet

– de repérer les formes les plus vraisemblablement responsables d'erreurs,
– pour chaque forme f, d'identifier les phrases non analysables p_i où une de ses occurrences $o_{i,j} \in \mathcal{O}_f$ est un des principaux suspects et où $o_{i,j}$ a un taux de suspicion parmi les plus élevés parmi les occurrences de f.

[3] Nous avons également fait des expériences où S_f est estimé à l'aide d'un autre estimateur, le *taux de suspicion moyen lissé*, noté $\tilde{S}_f^{(n)}$, qui prend en compte le nombre d'occurrences de f. En effet, la confiance que l'on peut accorder à l'estimation $S_f^{(n)}$ est d'autant plus faible que le nombre d'occurrences occ_f est faible. D'où l'idée de lisser $S_f^{(n)}$ en le remplaçant par un barycentre $\tilde{S}_f^{(n)}$ de $S_f^{(n)}$ et de $\overline{S}$ dont le coefficient de pondération λ dépend de $|\mathcal{O}_f|$: si $|\mathcal{O}_f|$ est grand, $\tilde{S}_f^{(n)}$ sera proche de $S_f^{(n)}$; s'il est petit, il sera plus proche de $\overline{S}$:

$$\tilde{S}_f^{(n)} = \lambda(|\mathcal{O}_f|) \cdot S_f^{(n)} + (1 - \lambda(|\mathcal{O}_f|)) \cdot \overline{S}.$$

Dans ces expériences, nous avons utilisé la fonction de lissage $\lambda(|\mathcal{O}_f|) = 1 - e^{-\beta|\mathcal{O}_f|}$ avec $\beta = 0.1$. Mais ce modèle couplé au classement selon $M_f = S_f \cdot \ln|\mathcal{O}_f|$ (voir plus bas) donne des résultats similaires au modèle sans lissage. Nous présentons donc le modèle non lissé, qui a l'avantage de ne pas faire intervenir de fonction de lissage choisie empiriquement.

L'algorithme a été implémenté en *perl*, en optimisant les structures de données pour réduire les coûts en mémoire et en temps. En particulier, les phrases stockent une liste d'occurrences qui pointent directement sur la structure associée à chaque forme.

2.3. Extensions du modèle

Ce modèle donne déjà de très bons résultats, comme nous le verrons à la section 4. Cependant, on peut l'étendre de différentes façons, parmi lesquelles nous avons déjà implémenté certaines.

Tout d'abord, il est possible ne pas se restreindre aux formes. De fait, nous ne travaillons pas sur les formes seules, mais sur des couples formés d'une forme (une entrée du lexique) et du ou des tokens qui leur correspondent dans le texte brut (un token est une portion de texte délimitée par des espaces ou des tokens de ponctuation).

Par ailleurs, on peut vouloir rechercher la cause de l'échec de l'analyse d'une phrase pas seulement dans la présence d'une forme dans cette phrase, mais aussi dans la présence d'un bigramme voire d'un trigramme de formes. Il suffit pour cela de généraliser la notion d'occurrence : on peut dire que les occurrences associées à une phrase sont un ensemble de faits simultanés caractéristiques (mais non spécifiques) de cette phrase. Par exemple, en définissant les occurrences associées à une phrase comme l'ensemble des formes et des bigrammes de formes de cette phrase, on obtient des résultats très intéressants.

L'autre généralisation possible consisterait à savoir prendre en compte des faits qui ne sont pas simultanés, mais qui sont des hypothèses concurrentes, qu'il faut par conséquent probabiliser aussi. Nous n'avons pas encore mis en œuvre un tel mécanisme. Il serait pourtant très intéressant, car il permettrait de dépasser le stade de la forme ou des n-grammes de formes et d'obtenir des généralisations, par exemple au niveau des lemmes.

3. Mise en œuvre

Nous avons appliqué ces principes pour rechercher les causes d'erreurs dans les sorties de deux systèmes d'analyse syntaxique profonde pour le français, FRMG et SXLFG, sur de gros corpus.

3.1. Analyseurs

Les deux systèmes d'analyse que nous avons utilisés reposent sur des analyseurs syntaxiques profonds non-probabilistes. Ils ont en commun :

– le lexique syntaxique Le*fff* 2 (Sagot *et al.*, 2005), qui comporte 600 000 entrées (couvrant 400 000 formes distinctes) ; une entrée regroupe informations morphologiques, cadres de sous-catégorisation syntaxique (lorsque pertinent) et informations syntaxiques complémentaires, en particulier pour les formes verbales (contrôles, attributifs, impersonnels,...) ;
– la chaîne de traitement pré-syntaxique SXPipe (Sagot et Boullier, 2005), qui convertit un texte brut en suite de treillis de mots présents dans le Le*fff* ; SXPipe comprend entre autres des modules de segmentation en phrases, de tokenization et correction orthographique, de détection d'entités nommées, et d'identification non-déterministe des mots composés.

En revanche, FRMG et SXLFG utilisent des analyseurs complètement différents, ne reposant ni sur le même formalisme, ni sur la même grammaire, ni sur le même générateur d'analyseurs. La comparaison des résultats de fouille d'erreurs sur les sorties de ces systèmes nous permet donc de distinguer les erreurs dues à Le*fff* ou à SXPipe d'une part, et celles dues à l'une ou l'autre

des grammaires d'autre part. Voyons plus en détails les caractéristiques de ces deux analyseurs.

L'analyseur FRMG (Thomasset et Villemonte de la Clergerie, 2005) s'appuie sur une grammaire compacte TAG du français générée à partir d'une méta-grammaire. La compilation et l'exécution de l'analyseur se déroule dans le cadre du système DYALOG (Villemonte de la Clergerie, 2005).

L'analyseur SXLFG (Boullier et Sagot, 2005) est un analyseur LFG efficace et robuste. L'analyse est effectuée en deux phases. Tout d'abord, un analyseur à la Earley construit une forêt partagée représentant l'ensemble des analyses en constituants qui satisfont le squelette non-contextuel de la grammaire. Puis les structures fonctionnelles sont construites de bas en haut, éventuellement en plusieurs passes. L'efficacité de l'analyseur repose sur diverses techniques de représentation compacte de l'information, sur l'emploi systématique de méthodes de partage de calcul et de structures, sur des techniques d'évaluation paresseuse, et sur l'élagage heuristique et quasiment non-destructif en cours d'analyse. Enfin, la grammaire utilisée est un descendant de la grammaire LFG utilisée par Lionel Clément pour son système XLFG.

Nos deux analyseurs implémentent aussi également des techniques évoluées de rattrapage et de tolérance d'erreurs, mais elles sont inutiles dans le cadre des travaux décrit ici, puisque nous ne cherchons qu'à distinguer les phrases qui ont une analyse complète (sans utilisation de techniques de rattrapage) de celles qui n'en ont pas.

3.2. Corpus

Nous avons analysé, à l'aide de ces deux systèmes, les corpus suivants :

Corpus MD : Il s'agit de plusieurs centaines de milliers de phrases d'un corpus d'articles du *Monde diplomatique* dont le style est naturellement exclusivement journalistique.

Corpus EASy : Il s'agit du corpus d'environ 40 000 phrases constitué pour la campagne d'évaluation des analyseurs syntaxiques EASy. Nous n'avons utilisé que les corpus bruts, sans prendre en compte le fait que pour environ 10% des phrases une annotation manuelle est disponible. Le corpus EASy regroupe des sous-corpus de styles variés : journalistique, littéraire, législatif, médical, transcription d'oral, courrier électronique, questions, etc.

Les deux corpus sont relativement bruts dans le sens où aucun nettoyage n'a été effectué pour éliminer certaines séquences de caractères ne pouvant pas être considérées comme des phrases.

La table 1 donne quelques informations générales sur les corpus utilisés et sur les résultats obtenus par les deux analyseurs. Il est à noter que les deux analyseurs n'ont pas exploité exactement le même jeu ni le même nombre de phrases pour MD, et qu'ils n'utilisent pas exactement la même notion de phrase.

corpus	#phrases	#succès (%)	#formes	#occ	$\overline{S}$ (%)	Date
MD/FRMG	330 938	136 885 (41.30%)	255 616	10 422 926	1.86%	Juil. 05
MD/SXLFG	630 000	337 437 (53.56%)	373 986	15 840 364	1.85%	Déc. 05
EASy/FRMG	39 872	16 477 (41.32%)	61 135	878 156	2.66%	Déc. 05
EASy/SXLFG	39 872	21 067 (52.84%)	61 135	878 156	2.15%	Déc. 05

Tableau 1. Informations générales sur les corpus et les résultats d'analyse

3.3. Environnement de visualisation des résultats

Nous avons développé un outil de visualisation des résultats de la fouille d'erreurs, qui permet de les parcourir et de les annoter. Il s'agit d'une page WEB qui fait usage de techniques de génération dynamique, en particulier via *javascript*. Un exemple est montré figure 1.

Pour cela, les suspects sont triés selon une mesure M_f modélisant pour une forme f l'intérêt qu'a l'utilisateur à tenter de corriger dans ses ressources l'erreur correspondante. Un utilisateur souhaitant privilégier les erreurs presque certaines sur les erreurs fréquentes peut visualiser les suspects en fonction de $M_f = S_f$. Un utilisateur souhaitant privilégier la fréquence d'une erreur potentielle au détriment de sa crédibilité peut les visualiser en fonction de[4] $M_f = S_f|\mathcal{O}_f|$. Une position intermédiaire, adoptée ici, consiste à les trier en fonction de $M_f = S_f \cdot \ln|\mathcal{O}_f|$.

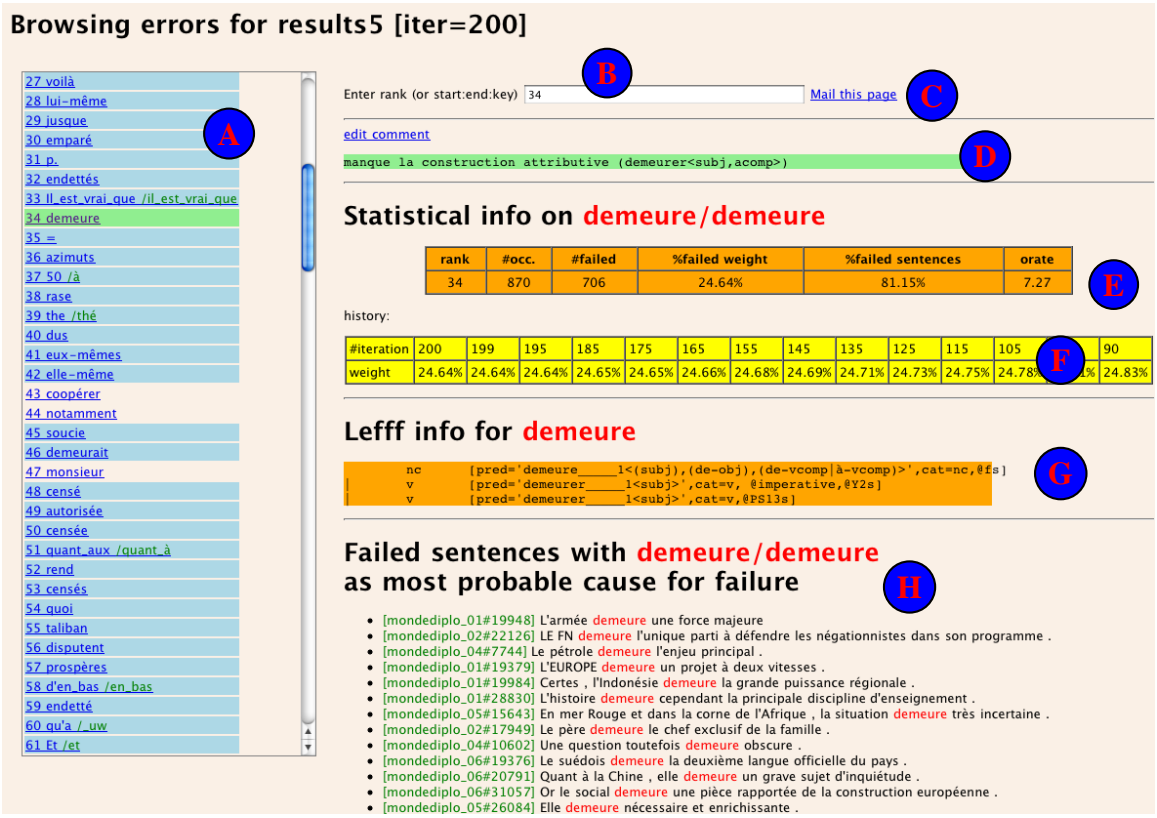

*Figure 1. Visualisation du résultat de la fouille d'erreurs (illustré pour MD/*FRMG*).*

L'environnement de visualisation permet de naviguer entre les suspects (triés) dans un menu à gauche de la page (**A**). Lorsque le token suspect est égal à la forme, seul le token est indiqué. Sinon, le token est séparé de la forme par «/». Si l'on sélectionne un suspect, la partie droite de la page indique toutes sortes d'informations à son sujet. Après avoir indiqué son rang dans le classement selon la mesure M_f choisie (**B**), un champ est disponible pour ajouter ou éditer un commentaire associé au suspect (**D**). Ces commentaires, destinés au dépouillement des résultats par des linguistes ou par les développeurs des analyseurs et des grammaires, sont stockés dans une base de données (SQLITE). Des informations statistiques sont également fournies pour

[4] Soit une forme f. Le taux de suspicion S_f peut être vu comme la probabilité qu'une occurrence particulière de f induise une erreur. Par conséquent, $S_f|\mathcal{O}_f|$ modélise le nombre d'occurrences de f en tant que source d'erreur.

f (**E**), dont son nombre d'occurrences occ_f, le nombre d'occurrences de f appartenant à des phrases non analysables, l'estimation finale de son taux de suspicion moyen S_f, et le pourcentage $\mathrm{err}(f)$ de phrases non analysables parmi celles où f apparaît. Ces indications sont complétées par un bref historique (**F**) montrant la convergence du S_f. La partie inférieure de la page donne le moyen d'identifier les causes d'erreurs provenant de f en montrant d'une part (**G**) les entrées de f dans le lexique Lefff, et d'autre part (**H**) les phrases non analysables où f est le suspect principal et où une de ses occurrences a un taux de suspicion spécialement élevé.

La page avec commentaires peut être envoyée par mail, par exemple au gestionnaire de Lefff ou au développeur de tel ou tel analyseur (**C**).

4. Résultats

Pour MD/FRMG, l'algorithme a effectué 200 itérations en environ 2700 s sur un PC à 3.2 Ghz avec 1 Go de mémoire vive. Il a proposé 18 492 formes dites *pertinentes* (sur les 255 616 possibles), une forme pertinente étant une forme f qui vérifie les contraintes (arbitraires) de seuil suivantes[5] : $S_f^{(200)} > 1,5 \cdot \overline{S}$ et $|\mathcal{O}_f| > 5$.

Nous ne pouvons pas encore assurer la convergence théorique de l'algorithme[6]. Mais sur les 1000 premières formes retournées pour MD/FRMG, la dernière itération fait varier le taux de suspicion de moins de 0,01% en moyenne (le maximum est de 0,06%, et seules 10 formes varient de plus de 0,01%). Et en pratique, il n'est pas nécessaire de réaliser 200 itérations et on obtient déjà de bons résultats avec une cinquantaine d'itérations seulement.

Pour un « petit » corpus comme EASy/FRMG, les 200 itérations prennent 260s, retournent 2678 formes (sur les 61 125 possibles). Les indications de convergence sur les 1000 premières formes sont équivalentes à celles obtenues pour MD/FRMG.

La table 2 donne une idée de la répartition des suspects par rapport à leur fréquence, montrant que les formes rares ont proportionnellement plus de chances d'être suspectes. La forme suspecte la plus fréquente est « " » avec $S_f = 9\%$, résultant en partie de problèmes de segmentation.

#occ	$>$ 100 000	$>$ 10 000	$>$ 1000	$>$ 100	$>$ 10
# formes	13	84	947	8345	40 393
# formes suspectes (%)	1 (7.6%)	13 (15.5%)	177 (18.7%)	1919 (23%)	12 022 (29.8%)

*Tableau 2. Répartition des suspects pour MD/*FRMG

4.1. Causes d'erreurs identifiées

Les tables 3 et 4 montrent des extraits des résultats obtenus en appliquant notre méthode sur les résultats d'analyse du corpus MD par FRMG. Elles donnent respectivement les 20 couples token(s)/forme les mieux classés, et ceux classés 100 à 109, pour montrer que les résultats restent pertinents bien au-delà des premiers rangs. Outre le taux de suspicion et le nombre d'oc-

[5] Ces seuils éliminent des formes à la sortie des résultats mais il est à noter que toutes les formes et leurs occurrences sont utilisées lors des itérations.

[6] Mais l'algorithme présente des ressemblances troublantes avec des algorithmes itératifs et convergents proposés pour la recherche de distributions de probabilités à entropie maximale sous un ensemble de contraintes (Berger *et al.*, 1996).

Rang	Token(s)/forme	$S_f^{(200)}$	$\lvert\mathcal{O}_f\rvert$	$\mathrm{err}(f)$	M_f	Origine de l'erreur
1	;	65%	13379	100%	6.16	FRMG : « ; » ne peut terminer les phrases dans FRMG
2	*	83%	199	100%	4.40	corpus : phrases comportant le seul mot « * »
3	(...)	58%	1835	100%	4.37	SXPipe : on devrait en faire un mot ignorable
4	coll/coll.	74%	257	93%	4.09	FRMG : ne sait pas analyser «L'Harmattan , coll.»
5	jour	44%	3005	100%	3.49	Lefff (ancienne version) : manquait la forme « jour »
6	[...]	61%	173	100%	3.16	SXPipe : on devrait en faire un mot ignorable
7	feu	47%	643	100%	3.01	Lefff : « faire long feu » ; FRMG : adj. pré-déterminant
8	date	45%	749	100%	2.97	FRMG (ancienne version) : pb sur les verbes support
9	confiance	44%	666	100%	2.89	FRMG(ancienne version) : pb sur les verbes support
10	80/à	57%	138	100%	2.83	SXPipe : bug (idem aux rangs 12 : 70/à, 13 : 60/à et 14 : 30/à)
11	etc/etc.	44%	493	100%	2.74	Lefff : ce n'est pas une ponctuation...
15	voici	43%	269	93%	2.43	FRMG : grammaire incomplète
16	qu'elle/quelle	30%	3052	96%	2.37	SXPipe (ancienne version) : bug (id. 17 :qu'elles/quelles)
17	contrairement	39%	313	100%	2.23	Lefff : Manque la notion de sous-cat adverbiale (ici *à-obj*)
18	fiche	63%	34	94%	2.23	Lefff : manque comme nom commun (!)
19	emparé	66%	29	100%	2.22	Lefff : Manquent les constr. pronominales (*s'emparer (de)*)
20	demeurent	37%	398	87%	2.21	Lefff : Manque la construction attributive

Tableau 3. Analyse des 20 premières formes (classées selon $M_f = S_f \cdot \ln |\mathcal{O}_f|$)

currences, nous donnons le taux $\mathrm{err}(f)$ (utilisé par (van Noord, 2004)) qui est le pourcentage d'échecs d'analyse parmi les phrases comportant une occurrence de la forme f, ainsi qu'une analyse manuelle de la cause de l'erreur identifiée automatiquement.

On constate que les résultats sont très bons : les deux tableaux identifient correctement des erreurs, réparties en quatre sources : les erreurs dans le lexique Lefff, les erreurs dans la chaîne de traitement pré-syntaxique SXPipe, les imperfections de la grammaire, mais aussi les problèmes issus du corpus en raison du fait qu'il s'agit d'un corpus brut.

Rang	Token(s)/forme	$S_f^{(200)}$	$\lvert\mathcal{O}_f\rvert$	$\mathrm{err}(f)$	M_f	Origine de l'erreur
100	investir	30%	136	86%	1.48	Lefff : l'objet direct n'est pas obligatoire
101	clôt	47%	23	96%	1.48	Lefff : Manquent les constr. pronominales (*se clore*)
102	demain	25%	378	81%	1.48	Lefff et FRMG : Peut former un GN saturé (*pour demain*)
103	Seuls/seuls	29%	169	95%	1.48	FRMG : adj. pré-déterminant (id. au rang 104)
105	autoproclamé	50%	19	100%	1.47	Lefff : Manque la construction attributive
106	renchérit	45%	25	92%	1.46	Lefff et FRMG : Traiter les constructions narratives
107	emparée	50%	18	100%	1.46	Lefff : Manquent les constr. pronom. (id. rang 109, cf. rang 19)
108	Mille/_NUMBER	56%	13	100%	1.45	SXPipe : *Mille et une* pas reconnu comme un nombre

Tableau 4. Analyse des formes de rang 100 à 109 (classées selon $M_f = S_f \cdot \ln |\mathcal{O}_f|$)

Pour le corpus EASy, les résultats sont également pertinents, mais parfois plus délicats à interpréter, suite à la taille limitée du corpus, à sa grande hétérogénéité et à la présence de sous-corpus de courriers électroniques et de transcription de corpus oraux qui introduisent beaucoup de bruit. L'importance des erreurs de segmentation (dues à la fois à SXPipe et au corpus lui-même, déjà segmenté) s'en trouve renforcée.

4.2. Comparaison entre les deux analyseurs

Nous avons complété l'étude des résultats séparés de nos deux systèmes d'analyse par un couplage des deux résultats. Nous avons calculé pour cela pour chaque forme la moyenne har-

monique des mesures $M_f = S_f \cdot \ln|\mathcal{O}_f|$ obtenus pour chaque système d'analyse. Les résultats sont très intéressants, car ils identifient des erreurs provenant en majorité des ressources partagées entre les deux systèmes (le lexique Le*fff* et la chaîne de traitement pré-syntaxique SXPipe). Bien que certaines erreurs sont issues d'incomplétudes communes aux deux grammaires, c'est néanmoins un moyen efficace d'obtenir une première répartition entre sources d'erreurs différentes.

Rang	Token(s)/forme	$\bar{M}_f$	Origine de l'erreur
1	(...)	3,55	SXPipe : en faire un mot que l'on peut ignorer (_EPSILON)
2	[...]	2,85	SXPipe : comme pour *[...]*
3	demeurent	2,23	Le*fff* : Manque la construction attributive
4	Premières/Premiere	2,09	SXPipe : bug
5	emparé	2,05	Le*fff* : Manquent les constr. pronominales (*s'emparer*)
6	endettés	2,04	Le*fff* : Manque en tant qu'adjectif
7	lui-même	1,99	Le*fff* : À mettre comme un adjectif spécial
8	endetté	1,87	Le*fff* : comme pour *endettés*
9	elle-même	1,81	Le*fff* : À mettre comme un adjectif spécial
10	d'en_bas/en_bas	1,79	Le*fff* et grammaires : Constr. 'prep+adv' pour certains adv
11	larvée	1,78	Le*fff* : Manque comme adjectif
12	–/-	1,75	grammaires : gestion du tiret parenthétique

Tableau 5. Couples token(s)/forme maximisant la moyenne harmonique $\bar{M}_f$ des mesures M_f^{FRMG} et M_f^{SXLFG} obtenues par FRMG *et* SXLFG

À titre d'illustration, la table 5 montre quels sont les 10 couples token(s)/forme dont la moyenne harmonique des taux de suspicion obtenus par FRMG et SXLFG est la plus grande (sur les résultats d'analyse du corpus MD).

5. Conclusions et perspectives

L'analyse de corpus importants permet donc de mettre en place des techniques de fouille d'erreurs, pour identifier les incomplétudes et les inexactitudes dans les différentes ressources utilisées par un système d'analyse syntaxique complet. La technique décrite ici et mise en œuvre sur les formes (ou les couples token(s)/forme) nous a déjà permis de détecter nombre d'erreurs et de manques dans notre lexique Le*fff*, de révéler des comportements inappropriés dans notre chaîne de traitement pré-syntaxique SXPipe, et de mettre en avant le manque de couverture de nos grammaires pour certains phénomènes.

Nous souhaitons mettre en place différentes améliorations et extensions de ce travail. Tout d'abord, l'interface est améliorable, de même que l'implémentation de l'algorithme itératif.

Ensuite, nous pensons intégrer au modèle la possibilité que les faits pris en compte (ici les occurrences) ne soient pas nécessairement tous certains, mais puissent être parfois en concurrence les uns avec les autres. Par exemple, pour une forme donnée, plusieurs lemmes sont souvent possibles. Les probabiliser permettrait donc de rechercher les *lemmes* les plus suspects.

Nous comptons également développer un module permettant non seulement de détecter des erreurs, par exemple dans le lexique, mais également de proposer une correction. Pour cela, nous

envisageons, pour chaque phrase dont l'analyse a échoué, de relancer l'analyse en remplaçant les informations lexicales associées au suspect principal de la phrase par des informations sous-spécifiées. Ces informations pourraient être obtenues par exemple par généralisation, selon certaines règles, des informations associées au suspect dans le lexique. On pourrait aussi traiter le suspect principal comme s'il était un mot inconnu, à qui l'on associerait des informations très peu contraignantes. L'étude statistique des analyses ainsi obtenues permettraient alors de proposer des corrections pour le mot en question.

Références

BERGER A., DELLA PIETRA S. et DELLA PIETRA V. (1996). « A maximun entropy approach to natural language processing ». In *Computational Linguistics*, 22 (1), pp. 39–71.

BOULLIER P. et SAGOT B. (2005). « Efficient and robust LFG parsing : SxLfg ». In *Proceedings of IWPT'05*. Vancouver, Canada.

SAGOT B. et BOULLIER P. (2005). « From raw corpus to word lattices : robust pre-parsing processing ». In *Proceedings of L&TC 2005*. Poznań, Pologne.

SAGOT B., CLÉMENT L., VILLEMONTE DE LA CLERGERIE E. et BOULLIER P. (2005). « Vers un méta-lexique pour le français : architecture, acquisition, utilisation ». Journée d'étude de l'ATALA sur l'interface lexique-grammaire et les lexiques syntaxiques et sémantiques.

THOMASSET F. et VILLEMONTE DE LA CLERGERIE E. (2005). « Comment obtenir plus des Méta-Grammaires ». In *Proceedings of TALN'05*. ATALA, Dourdan, France.

VAN NOORD G. (2004). « Error Mining for Wide-Coverage Grammar Engineering ». In *Proc. of ACL 2004*. Barcelona, Spain.

VILLEMONTE DE LA CLERGERIE E. (2005). « DyALog : a Tabular Logic Programming based environment for NLP ». In *Proceedings of 2nd International Workshop on Constraint Solving and Language Processing (CSLP'05)*. Barcelona, Spain.