

FRMG: évolutions d'un analyseur syntaxique TAG du français

Éric de la Clergerie*, Benoît Sagot*, Lionel Nicolas[◇] et Marie-Laure Guénot*

* ALPAGE, INRIA & Paris 7

[◇] Équipe RL, Laboratoire I3S, UNSA+CNRS

Abstract

Nous présentons FRMG, un analyseur syntaxique du français à large couverture. Nous mettons en avant les méthodes qui ont permis d'améliorer ses performances depuis sa naissance, en 2004, initiée dans le cadre de la première campagne **EASy** d'évaluation des analyseurs syntaxiques.

Introduction

FRMG (*F*rench *M*eta*G*rammar) est un analyseur syntaxique profond à large couverture pour le français. Une description grammaticale de haut niveau sous forme de *métagrammaire* (MG) sert de point de départ pour la génération d'une grammaire d'arbres adjoints (TAG, *Tree Adjoining Grammar*) (Joshi, 1987; Abeillé, 2002) [Section 1]. Cette grammaire est transformée par le système DYALOG (Villemonde de La Clergerie, 2005; Villemonde de la Clergerie, 2002a) en un analyseur syntaxique.

À la différence d'autres grammaires TAG à large couverture comprenant plusieurs milliers d'arbres, FRMG se caractérise par un très faible nombre d'arbres (189 en Septembre 2009). Ce résultat découle de bonnes synergies entre les possibilités descriptives des MG et les mécanismes de factorisation offerts par DYALOG [Section 2]. Ces synergies permettent de plus facilement améliorer la couverture linguistique sans nécessairement sacrifier les performances en terme de temps d'analyse [Section 3].

Précédemment mentionné, DYALOG est un environnement de programmation en logique permettant, en autres choses, la compilation de grammaires en analyseurs efficaces [Section 4]. En particulier, DYALOG réalise une analyse préalable de la grammaire FRMG pour déterminer quels en sont les arbres qui peuvent être compilés en tant qu'arbres TIG (*Tree Insertion*

Grammar) (Schabes and Waters, 1995) de manière à offrir une meilleure complexité. Il permet aussi la construction d'analyseurs robustes pouvant rendre, si besoin, un ensemble d'analyse partielles couvrant une phrase. C'est également en DYALOG qu'est développé l'outil MGCOMP qui transforme la MG en grammaire TAG.

L'analyseur syntaxique FRMG découlant des phases de compilation de la métagrammaire FRMG ne peut bien sûr fonctionner seul. Il s'intègre dans une chaîne complète de traitement comprenant, en amont, le lexique syntaxique LEFF et les nombreux modules de SXPIPE en charge de la segmentation, correction orthographique et détection des entités nommées [Section 5]. De même, les résultats bruts d'analyse fournis par FRMG sous forme d'une forêt complète de dérivations TAG sont post-traités pour obtenir une sortie sous forme de dépendances, éventuellement désambiguïsée et éventuellement convertie aux formats **EASy** ou **PASSAGE** [Section 6]. Ces vues par dépendances servent de point d'entrée pour les campagnes d'évaluation **EASy** (Paroubek et al., 2008) et maintenant **PASSAGE**, mais aussi pour des tâches d'acquisition de connaissance ou d'extraction d'information. Un exemple de sortie désambiguïsée en dépendances est fournie Figure 1 pour la phrase "*Quel livre Paul propose-t-il (plus abruptement qu'elle) de faire lire à Marie ?*" (avec volontairement une faute d'orthographe !). Elle nous servira de fil conducteur le long de ce document.

Historiquement, la métagrammaire FRMG et l'environnement de compilation MGCOMP (Thomasset and Villemonde de La Clergerie, 2005) sont nés courant 2004, motivés par la première campagne **EASy** (fin 2004). Leur développement initial, conduit dans le cadre du projet INRIA ATOLL, a requis moins de 6 mois, démontrant le potentiel des métagrammaires pour la description de phénomènes syntaxiques. Néanmoins, le développement d'une grammaire à large couverture

reste une tâche de longue haleine. C’est également vrai pour l’ensemble des composants de la chaîne. Ce développement doit être guidé par des retours d’utilisation sur corpus permettant de mettre en relief les manques et erreurs des divers composants ainsi que les problèmes de performances. Ce travail a été fait en combinant diverses techniques supervisées (avec une référence) ou non, conduisant à des améliorations très sensibles de FRMG et du reste de la chaîne [Section 7].

Cet effort d’amélioration, maintenant conduit dans le cadre du projet ALPAGE (INRIA & Université Paris 7), reste un objectif permanent, tant il vrai qu’on ne peut jamais considérer comme achevé un outil de traitement linguistique.

1 Métagrammaires

Introduites par (Candito, 1999) et depuis plusieurs fois reprises (Clément and Kinyon, 2003; Gaiffe et al., 2002; Duchier et al., 2005), les métagrammaires sont une des réponses apparues dans la communauté TAG pour faire face aux problèmes de développement de larges grammaires TAG (Doran et al., 1994). En effet, le domaine de localité étendu des arbres TAG entraîne une explosion combinatoire du nombre d’arbres, ceux-ci partageant de nombreux sous-arbres. Ainsi, la structure du noyau verbal va se retrouver dans tous les arbres ancrés par des verbes. Modifier la description d’un de ses sous-arbres implique *a priori* la modification de tous les arbres le contenant, posant des problèmes de maintenance. Les métagrammaires apportent une solution élégante à ces questions de développement et de maintenance, mais pas directement à celle de l’explosion combinatoire.

Plutôt que de métagrammaire, il serait plus juste de parler d’une grammaire source descriptive servant à produire une grammaire cible opérationnelle. Une MG est organisée en une hiérarchie multiple de classes. Chaque classe contient des éléments de description exprimés sous forme de contraintes sur et entre nœuds. Parmi ces contraintes, citons la dominance (strict ou immédiate) et la précédence linéaire, ainsi que des contraintes de décoration (par des structures de traits) sur les nœuds ou sur la classe.

Les classes peuvent aussi indiquer qu’elles fournissent ou requièrent une certaine ressource, par exemple la ressource `sujet`. Ainsi, dans FRMG, beaucoup de modifieurs au niveau de

la phrase (adverbes, groupes prépositionnels, adverbess temporels, ...) s’appuient sur la fonctionnalité `s_modifieur` qui s’appuie elle-même sur une ressource `x_modifieur` (modifieur générique, possiblement incisé) qui s’appuie sur la ressource `shallow_auxiliary` définissant la base d’un arbre auxiliaire. Une même ressource peut être demandée plusieurs fois en se plaçant dans des espaces de noms différents pour éviter des conflits. Ainsi, une ressource “d’accord en genre et nombre” entre un nœud et son père est requise plusieurs fois dans certaines classes. Le listing 1 montre la description de l’apposition comme modifieur post-posé du groupe nominal.

```
class noun_apposition {
- x_modifieur;      %% modifieur
  Foot < Incise;    %% post-posé
  node(Root).cat = value(N2); %% de GN
- Foot :: full_agreement; Foot = Foot :: N;
  ...
}
```

Listing 1 – Exemple de classe

Les classes sont ensuite croisées pour mettre en relation fournisseurs et demandeurs tout en accumulant progressivement leurs contraintes. Seules sont finalement conservées les classes neutres dont les contraintes sont satisfiables. Les contraintes des classes neutres survivantes sont alors utilisées pour produire des schémas d’arbres minimaux.

La modularité des descriptions permet de construire des arbres en combinant les contraintes de plusieurs classes, chacune se focalisant sur un point particulier. D’un autre côté, une même classe peut participer à plusieurs classes neutres, apportant ainsi une factorisation dans le développement des grammaires. Ainsi la notion de sujet peut être liée à une classe de base, portant la ressource `+sujet`, qui va être utilisée dans de très nombreuses classes neutres. L’ensemble des arbres produits à partir des classes neutres constitue la grammaire cible résultant de la compilation de la métagrammaire source. Ainsi, la MG de français (CRABBÉ, 2005) écrite avec XMG (Duchier et al., 2005) produit une grammaire d’environ 8 000 arbres.

2 Factorisation

Plusieurs approches ont été proposées pour réduire le nombre d’arbres dans les grammaires TAG, soit par compilation d’un ensemble d’arbres

en un automate (Carroll et al., 1998), soit en décorant un arbre par un automate de parcours (Harbusch and Woch, 2004) de manière à représenter de fait plusieurs arbres. En un sens, DIALOG offre une forme restreinte de cette dernière approche en permettant l’emploi d’opérateurs réguliers sur des sous-arbres. Il est ainsi possible de parler de la disjonction de deux sous-arbres, de la répétition d’un sous-arbre (par une étoile de Kleene) ou de l’entrelacement de deux séquences de sous-arbres¹. L’optionnalité d’un sous-arbre découle de l’opération de disjonction et est raffinée grâce à la notion de garde positive (resp. négative) permettant de poser des conditions sur la présence (resp. l’absence) d’un sous-arbre. Présents dans DIALOG, ces opérateurs sont rendus disponibles au niveau descriptif des MG dans MG-COMP, en particulier pour les gardes qui s’expriment comme des expressions booléennes sur des équations entre chemins de traits. Ces opérateurs ne changent pas la nature du formalisme grammatical mais permettent une réduction exponentielle du nombre d’arbres.

Ainsi, une expérience de *défactorisation* d’une ancienne version de FRMG² a produit 1 986 621 arbres, quasiment exclusivement ancrés par des verbes (1 986 300, soit 99,98%), le reste ne représentant que 321 arbres. La factorisation est de fait concentrée au sein de quelques arbres : seuls 6 arbres ont un taux d’expansion supérieur à 10^5 , dont l’arbre #110 (constructions verbales canoniques) donnant 700 660 arbres à lui seul.

Comparés aux 8000 arbres précédemment mentionnés (CRABBÉ, 2005), nos chiffres sont au moins 2 ordres de grandeur au-dessus. La question de la surgénérativité de FRMG se pose donc. Il est clair que FRMG produit des arbres non factorisés associés à des cadres de sous-catégorisation n’existant pas en français. Des mesures plus précises sur les cadres en trop suggèrent qu’il doit être possible, avec plus de contraintes, de descendre en dessous de 100 000 arbres mais très difficilement en dessous de 10 000 arbres. Cependant, la sur-générativité de FRMG n’est pas réellement un problème si, en pratique, elle ne sur-génère pas trop et si la factorisation n’a pas d’impact sur les temps d’analyse. Sur le premier point, l’emploi de nombreuses gardes fait que FRMG est une

grammaire ambiguë mais pas plus, semble-t-il, que d’autres grammaires, et dont l’ambiguïté n’est pas principalement liée aux mécanismes de factorisation. Concernant l’efficacité, le simple fait de ne pas avoir à gérer des milliers d’arbres apporte un gain important à la compilation et ne semble pas pénalisant à l’exécution, même en présence de nombreuses gardes.³

3 Couverture linguistique

Il est difficile de fournir une liste exhaustive des phénomènes syntaxiques couverts par FRMG, en assurant de plus une représentation correcte. L’exemple de la Figure 1 illustre quelques phénomènes avec le traitement (a) d’une causative (avec *Marie* comme sujet inversé introduit par *à* et *faire* vu comme un modal modifiant *lire*) ; (b) un objet interrogatif extrait ; (c) un verbe à contrôle (*proposer*) adjoit sur *lire* avec sujet plein et sujet clitique inversé ; (d) un adverbe (*abruptement*) flottant et incisé entre parenthèses ; et (e) modification de l’adverbe par une comparative (*que * lui*).

Pour être plus complet, FRMG permet le traitement de la valence pour les verbes (jusqu’à trois arguments, sujet inclus) et, dans une moindre mesure, pour les noms et les adjectifs (sur les arguments phrastiques) et sur les adverbes et prépositions. Il est possible d’extraire des arguments des verbes dans des constructions relatives, interrogatives, clivées et topicalisées (comme *rare sont les amis fidèles*). Il est à noter que l’extraction *profonde* au sein d’un argument d’un verbe n’est pas directement possible dans le cadre des TAG. On ne peut donc gérer une phrase comme “*un livre [dont]_i vous êtes [le héros ϵ_i]*”. Une évolution vers les Multi-Component TAGs semble nécessaire à terme (en s’appuyant sur les Thread Automata (Villemonte de la Clergerie, 2002b)). Les constructions difficiles à décrire que sont les coordinations, superlatives et comparatives sont partiellement présentes (mais limitées sur les ellipses). Un effort a été fait sur la gestion des ponctuations, en particulier pour les incisives.

Bien que cet inventaire ne soit pas complet, de très nombreux phénomènes échappent encore à FRMG. C’est un problème classique des grammaires développées manuellement qui ont du mal à augmenter leur couverture à cause de très nom-

¹L’entrelacement de deux séquences s_1 et s_2 retourne toute les séquences comprenant l’ensemble des sous-arbres de s_1 et s_2 qui préservent les ordres respectifs de s_1 et s_2 .

²Version svn #377 du 27 Février 2007

³Des expériences ont été menées avec DIALOG en compilant (difficilement) une grammaire de 7000 arbres. Mais l’analyseur a été uniquement testé avec un tout petit lexique ce qui ne permet pas de tirer des conclusions.

breux phénomènes à taux d'occurrence très faible. Néanmoins, en comparaison avec des approches statistiques, les métagrammaires permettent plus facilement à un expert humain d'abstraire un phénomène ou une classe de phénomènes à partir de quelques exemples.

4 Quelques caractéristiques de l'analyseur

DYALOG permet la compilation de la grammaire TAG FRMG en un analyseur tabulaire pouvant rendre l'ensemble des analyses grâce à du partage de calculs. Pour chaque arbre élémentaire de la grammaire, la compilation produit une méta-transition qui regroupe l'ensemble des transitions d'un automate à 2 piles (2SA) nécessaires pour l'ensemble des traversées possibles de l'arbre. Un parcours d'arbre se modélise par une série de suspensions/reprises au niveau des nœuds de substitution, nœuds d'adjonction et nœuds pied (Villemonde de la Clergerie, 2001). Bien que ce modèle ne soit pas optimal pour les TAGs⁴, il est plus efficace en pratique pour une grammaire *réaliste* comme FRMG.

L'efficacité de l'analyse est également améliorée grâce à des analyses statiques de la grammaire.

La première analyse identifie les arbres TIG de la grammaire. Pour être TIG (Schabes and Waters, 1995), un arbre auxiliaire doit (1) avoir sa *dorsale* joignant la racine au nœud pied en frontière gauche ou droite et (2) n'avoir que des arbres TIG gauches (resp. droites) adjoinables sur une dorsale gauche (resp. droite). En ne prenant pas en compte les décorations sur les nœuds, les TIG assurent une complexité cubique équivalente à celle des CFG, au lieu de $O(n^6)$ pour les TAG. C'est tout l'intérêt des TIG, d'autant que FRMG est quasi entièrement TIG. Néanmoins, quelques phénomènes, comme la modification d'un comparateur par la seconde partie d'une comparative (*plus X que Y*, illustrée Figure 1) restent TAG et tendent, par propagation, à rendre non-TIG d'autres arbres.

DYALOG utilise aussi des techniques d'interprétation abstraite pour déterminer quels traits morphosyntaxiques sont susceptibles d'être modifiés par adjonction (entre le pied et la racine d'un arbre auxiliaire). Les traits non modifiables requièrent une variable au lieu de deux en terme de représentation, ce qui réduit assez notablement

⁴En effet, il n'assure pas une complexité en $O(n^6)$, n dénotant la longueur de la phrase

les coûts d'unification.

Un gain important provient enfin de l'utilisation d'une stratégie d'analyse exploitant des informations coin gauche (*left-corner*). Ces informations sont calculées statistiquement par DYALOG et sont ensuite exploitées, au tout début de l'analyse syntaxique, pour déterminer, en fonction des mots de la phrase et des arbres activés, quelles sont les positions gauches possibles pour chaque arbre. Pendant l'analyse, ces contraintes de position réduisent largement l'espace de recherche.

L'analyse s'effectue sur un treillis de mots fourni par SXPIPE, qui permet de conserver les ambiguïtés lexicales et d'éviter le recours à un étiqueteur. Chaque transition dans un tel treillis est décoré (parfois de façon ambiguë) par les informations morphosyntaxiques et syntaxiques données par LEFFF pour le mot correspondant, informations rassemblées sous forme d'un *hypertag*. D'autre part, les mots inconnus ou partiellement spécifiés sont gérés, assurant une meilleure robustesse⁵.

DYALOG n'impose pas de contraintes de lexicalisation sur les arbres, permettant ainsi à certains arbres de FRMG de ne pas être ancrés lexicalement⁶. Néanmoins, DYALOG tire profit des arbres ancrés lexicalement en ne les activant que si au moins un des hypertags du treillis d'entrée est compatible avec les informations requises par l'ancre.

Pendant l'analyse, certains mots comme les *eah* ou les redoublements de mots à l'oral sont ignorés par FRMG, tout en laissant une trace dans les sorties d'analyse.

Enfin, par défaut, FRMG essaie de produire des analyses complètes pour une phrase et retourne l'ensemble de celles-ci. Mais, en l'absence d'analyse complète, l'analyseur bascule en mode robuste et retourne un ensemble d'analyses partielles et disjointes couvrant au mieux la phrase.

Pour toujours plus de robustesse et d'efficacité, FRMG peut analyser en mode *just-in-time*. À la fin d'un délai (*timeout*) imposé, les réponses trouvées sont émises même si les calculs ne sont pas terminés. Enfin, notons que l'analyseur peut fonctionner en mode flux, sans avoir à redémarrer pour chaque phrase, économisant ainsi le temps de la-

⁵Cette fonctionnalité est également un atout pour la recherche de corrections pour les mots erronés (dans le cadre de la fouille d'erreurs, Section 7).

⁶Ce qui fournit un moyen supplémentaire de réduire légèrement le nombre d'arbres.

tence de démarrage.

5 Intégration dans une chaîne de traitement

FRMG est utilisé au sein d'une chaîne de traitement complète. En amont, divers pré-traitements sont effectués par la chaîne SXPIPE (Sagot and Boullier, 2008), un ensemble de composants qui assurent la segmentation des textes en phrases et formes, avec reconnaissance de formes complexes et agglutinées, correction orthographique⁷ (*abruptment* en *abruptement*, Figure 1) et détection des entités nommées. Comme indiqué plus haut, SXPIPE retourne un treillis de mots sous formes de transitions d'un automate à états finis.

SXPIPE et FRMG exploitent les informations lexicales fournies par le lexique LEFFF (Sagot et al., 2006). Le LEFFF est un lexique morphologique et syntaxique à large couverture, développé dans le cadre du formalisme lexical Alexina. Outre les informations morphosyntaxiques, les informations les plus pertinentes pour FRMG portent sur la sous-catégorisation des verbes, et plus récemment sur celle des adverbes, adjectifs et noms.

6 Sorties d'analyse

La sortie brute de l'analyseur FRMG est une forêt partagée de dérivations, indiquant que telle opération TAG (substitution, adjonction, ancrage) a pris place sur tel nœud de tel arbre pour tel constituant. Des modules *Perl* permettent une première conversion de cette forêt en une forêt partagée de dépendances : les ancres des arbres mis en relation par une opération TAG sur un nœud de label *L* sont mis en relation de dépendance avec le label *L*. Les labels des nœuds dans FRMG sont généralement choisis pour refléter soit leur fonction grammaticale (*subject*, *object*, ...), soit leur catégorie syntaxique, ce qui donne d'office des labels plus parlants pour les dépendances. Pour les arbres non ancrés, des pseudo-ancres sont ajoutées pour un traitement uniforme, comme par exemple la pseudo-ancre *incise* en Figure 1.

Une forêt de dépendance est représentée dans le format **DEP XML** qui reprend les éléments graphiques de la Figure 1. À ce jour, **DEP XML** s'appuie essentiellement sur⁸ :

⁷Qui peut être non-déterministe, délégrant ainsi dans ce cas le choix de la bonne correction à l'analyseur.

⁸Nous omettons la description complète des composants **op** et **hypertag** pour des raisons de place.

- des **clusters** associés aux mots de la phrase ;
- des **nodes** pointant vers un cluster et décorés d'un lemme, d'une catégorie syntaxique et d'un ensemble de dérivations. Il est à noter qu'au travers des différentes alternatives, un nœud peut être la tête de plusieurs constituants **op** différents avec des emfans différents ;
- des **edges** mettant en relation un nœud source avec un nœud cible et décorés par un label. Plus finement, un arc *e* indique avec des sous-éléments **deriv** quelles sont les dérivations associées au nœud source qui utilisent *e*, en précisant de plus les emfans des constituants **op** sources et cibles.

Dans un contexte applicatif, il est souvent nécessaire de ne considérer qu'une analyse par phrase (si possible la bonne !). Cette nécessité a mené au développement d'un désambiguisateur, écrit en DIALOG. Il s'appuie sur un algorithme en programmation dynamique de recherche de la meilleure analyse en sommant les poids des arcs (et, dans une bien moindre mesure, des nœuds) participant d'une analyse. Le poids d'un arc résulte de l'accumulation des poids donnés par des règles élémentaires exprimées sous forme de motifs prenant en compte l'arc courant (nœuds source et cible, type, label) et éventuellement les arcs frères, fils ou parents, voire des arcs en compétition. Les poids sont choisis de manière heuristique et ne résultent pas, pour l'instant, d'un processus d'apprentissage. À ce jour, le désambiguisateur comporte environ 130 règles élémentaires. Par exemple, citons l'existence de règles favorisant les arcs remplissant la valence d'un verbe (sujet, objet, ...), la présence d'un sujet avant son verbe, l'inversion du sujet si certaines conditions sont remplies, etc. D'autres règles pénalisent les dépendances à longue distance, les transcatégorisations non nécessaires, certaines constructions possibles mais improbables, etc.

Progressivement, le désambiguisateur est devenu un composant important de la chaîne de traitement. Malheureusement, le coût théorique de l'algorithme est polynomial, conduisant en pratique à des temps de désambiguisation très longs sur certaines phrases très ambiguës. En moyenne, les temps de désambiguisation sont du même ordre de grandeur que les temps d'analyse et donc non négligeables. Par ailleurs, la désambiguisation est relativement instable, pouvant dramatiquement

dépendre de variations sur les poids. Ainsi, dans la Figure 1, le rattachement de l’objet extrait (bonus) est fortement contrebalancé par la distance importante du rattachement (malus), ce qui donne alors des analyses très différentes.

Enfin, dans le cadre des campagnes **EASy** et maintenant **PASSAGE**, le désambiguïsateur est complété par un convertisseur (écrit en DYALOG) vers les formats attendus par les organisateurs. Ces deux formats, très proches, proposent une annotation syntaxique sous forme de groupes non récursifs (GN, GA, GR, GP, NV, PV) et de relations (SUJ-V, AUX-V, COD-V, ATB-SO, CPL-V, MOD-V, MOD-N, MOD-A, MOD-R, MOD-P, COORD, APPOS, JUXT, COMP) entre formes et/ou groupes. Ces formats sont prévus pour des analyses plus surfaciques que celles rendues par FRMG, amenant à des pertes d’information et à des erreurs pendant la conversion.

7 Améliorer les performances

Les améliorations de FRMG se portent sur quatre grands axes :

- augmenter la **couverture** en terme d’analyses complètes. Même si FRMG peut retourner des analyses partielles, divers indicateurs montrent que les analyses complètes sont souvent de meilleure qualité ;
- augmenter la **qualité** des analyses, ce qui ne découle pas automatiquement de l’augmentation de la couverture et dépend également de la phase de désambiguïsation ;
- réduire le taux d’**ambiguïté** des analyses, pour rendre plus efficaces l’analyseur et le désambiguïsateur ;
- réduire les **temps d’analyse** (et de désambiguïsation). Ces temps ont malheureusement tendance à augmenter avec l’augmentation de la couverture, sauf à savoir bien contraindre l’emploi des diverses constructions syntaxiques. Néanmoins, si les temps d’exécution ont de l’importance dans certains contextes, l’utilisation de fermes⁹ de machines fournit une alternative viable pour le traitement *offline* de grands corpus.

Après une première génération d’outils de visualisation de résultats (comme la vue graphique Figure 1), nous avons développé de nouveaux outils fournissant des informations plus précises

⁹dans notre cas, GRID 5000 (<https://www.grid5000.fr>).

pour identifier des problèmes et indiquer des zones d’améliorations potentielles sur ces quatre axes.

En premier lieu, nous contrôlons régulièrement l’**efficacité**, l’**ambiguïté** et l’absence de perte de couverture sur les jeux de phrases test EUROTRA et TSNLP (Table 1). Néanmoins, on constate vite les limites des jeux de tests si on compare avec les résultats de couverture obtenus sur de vrais corpus (de plusieurs millions de phrases).

Jeux	#phrases	Couv.	t. moy. (s)	amb.
EUROTRA	334	100%	0.15	0.63
TSNLP	1161	95.07%	0.07	0.46
EasyDev	3879	64.73%	0.93	1.04
JRCacquis	1.1M	51.26%	1.41	1.1
Europarl	0.8M	70.19%	1.69	1.36
EstRep	1.6M	67.05%	0.69	0.92
Wikipedia	2.2M	69.11%	0.49	0.87
Wikisource	1.5M	61.08%	0.71	0.89
AFP	1.6M	52.15%	0.51	1.06

TAB. 1 – Informations sur quelques corpus (2009)

Une échelle plus réaliste est donnée par le corpus de développement EasyDev d’environ 4 000 phrases, pour lequel nous avons en outre des annotations de référence (au format EASy). Début 2008, suite à diverses modifications de FRMG, les temps d’analyses étaient montés assez haut. En testant diverses pistes¹⁰, plusieurs causes furent identifiées conduisant à un facteur d’accélération d’environ 10¹¹.

Améliorer la **couverture** revient avant tout à explorer les échecs et déterminer leurs causes, qu’elles soient lexicales, grammaticales ou autres. Des techniques récentes ne nécessitant pas de données de référence (van Noord, 2004; Sagot and de la Clergerie, 2008; Sagot and Villemonte de La Clergerie, 2006), basées sur l’analyse de grand corpus, accélèrent ce travail en permettant de suspecter automatiquement des entrées possiblement erronées ou incomplètes dans le lexique sur lequel repose l’analyseur. Ces techniques partent de l’idée qu’une forme mal décrite se retrouvera plus que de raison dans des échecs d’analyse : il s’agit donc de techniques de *fouille d’erreurs* dans les sorties d’analyseurs. D’autres techniques (Nicolas et al., 2008; Yi and Kordoni, 2006; van de Cruys, 2006), vont plus avant dans l’assistance en ajoutant une étape de génération de corrections potentielles. Pour ce faire, elles exploitent le fait que

¹⁰Dont beaucoup se révélèrent infructueuses !

¹¹Depuis, les temps d’exécution ont recommencé à légèrement monter suite à des extensions de couverture, illustrant ce jeu de va-et-vient entre les 4 axes.

le lexique et la grammaire seront rarement erronés pour un même élément donné. Les propositions de corrections lexicales sont donc générées en étudiant les attentes de la grammaire vis à vis de formes suspectées lors de l'analyse des phrases non-analysables associées. Cette étude est réalisée en remplaçant une forme suspecte par d'autres formes n'imposant pas les contraintes lexicales originelles de la forme suspectée. Cette opération est simple dans le cadre de FRMG qui accepte des formes lexicales sous-spécifiées. Couplée à d'autres (Nicolas et al., 2009), cette approche nous a déjà permis de corriger plus de 2800 formes¹². De plus, elle possède l'avantage d'identifier des sous-corpus représentatifs des manques de la grammaire, c-a-d. des corpus de phrases non analysables et sans suggestion de corrections.

Améliorer la couverture n'assure pas automatiquement l'amélioration de la **qualité** des analyses, qui dépend à la fois de la grammaire (et des autres composants en amont) et de la désambiguïsation. Améliorer la qualité repose sur la visualisation d'analyses mais aussi surtout sur l'utilisation de données de référence, en l'occurrence fournies par le corpus EasyDev qui couvre divers styles de documents (journalistique, littéraire, médical, questions, mails, oral spontané transcrit). Concrètement, il s'agit de permettre au linguiste de sonder les résultats sur les données de référence pour y détecter des séries d'erreurs récurrentes, afin de les analyser, d'en détecter la cause et de suggérer une solution. À cette fin ont été élaborés divers scripts, les premiers fournissant des évaluations de la qualité des résultats, les seconds ordonnant les résultats selon différents critères possibles. Pendant cette phase de correction manuelle nous effectuons un va-et-vient entre (i) les ressources qui servent à l'analyse, (ii) les résultats fournis et (iii) le corpus de référence EasyDev.

Les premiers scripts évaluent la qualité des résultats en fournissant des statistiques globales et détaillées par type de corpus, groupe ou relation. On y voit apparaître les erreurs les plus marquées, les groupes et les relations sur lesquelles il faut se concentrer plus particulièrement. À la vue de ces chiffres, on a pu constater que les erreurs se situaient plus souvent dans l'introduction de groupes erronés (rappel) que dans l'omission de bonnes constructions (précision). On a vu égale-

ment, par exemple, que concernant les groupes il fallait travailler en priorité sur la précision des GN (qui était de 73,07% avant les corrections) et le rappel des GR (72,77% avant les corrections) parce représentant les résultats les plus bas.

Pour plus de finesse, d'autres scripts, plus récents, produisent des matrices de confusion par groupes (Table 2) et par relations. Elles font apparaître les confusions les plus fréquentes entre une construction et une autre, et sont reliées aux énoncés correspondants, ce qui permet de se focaliser sur les descriptions grammaticales en jeu. Pour reprendre les exemples précédents, les matrices de confusion font apparaître que la plupart des GN manquants ont été soit mal segmentés soit étiquetés GP, et que la plupart des GR en trop auraient dû être étiquetés GA ou GP.

L'examen de ces énoncés donne lieu à une classification synthétique des erreurs. La différence constatée entre les résultats produits par FRMG et ceux provenant de l'annotation de référence peuvent venir de plusieurs origines différentes, qu'il est essentiel de distinguer afin de savoir à quel niveau de la chaîne de traitement une correction est nécessaire. Un nombre encore non négligeable d'erreurs proviennent du corpus de référence et ont été corrigés avec EasyRef (Éric Villemonte de la Clergerie, 2008). D'autres erreurs proviennent de SXPIPE, ou de désambiguïsations lexicales erronées (comme le choix d'une étiquette adjectif au lieu de nom). Ensuite viennent les erreurs lexicales liées à LEFFF, avec certaines étiquettes manquantes ayant échappé à la fouille d'erreur ou au contraire certaines en trop induisant en erreur la désambiguïsation. Enfin, nous trouvons les erreurs provenant de la grammaire FRMG. Ici, il peut s'agir de constructions qui n'ont pas été décrites suffisamment en détail (comme les phénomènes de coordination), ou alors, à l'opposée, de constructions trop contraintes pour prendre en compte certaines occurrences et pour lesquelles les contraintes sont alors relâchées.

Entre le début d'utilisation de cette méthode et les derniers résultats obtenus à ce jour, on note une progression générale, pour les groupes, de 2,51 points en précision, 6,82 points en rappel et 4,99 points en f-mesure ; pour les relations, on a gagné 1,52 points en précision, 5,18 en rappel et 3,55 en f-mesure. Par rapport aux exemples cités plus haut, la précision des GN a augmenté de 7,75 points et

¹²En sus des corrections trouvées manuellement par simple examen des suspects fournis par la première étape.

référence classée en hypothèse \implies	B_GN	GN	E_GN	BE_GN	B_GP
B_GN	5459 (91%)	52 (0.88%)	14 (0.28%)	115 (1.94%)	118 (1.99%)
GN	50 (4.65%)	725 (67%)	149 (13.86%)	19 (1.77%)	12 (1.12%)
E_GN	4 (0.07%)	68 (1.15%)	5345 (90.04%)	140 (2.36%)	10 (0.17%)
BE_GN	106 (3.22%)	45 (1.37%)	78 (2.37%)	2663 (80.97%)	7 (0.21%)
B_GP	166 (2.02%)	30 (0.37%)	2 (0.02%)	30 (0.37%)	7760 (94.61%)

TAB. 2 – Fragment de matrice de confusion sur les groupes (pour R139) avec BE_X pour un groupe de type X de taille 1, B_X pour une ouverture, E_X une fermeture, et X pour un composant interne.

le rappel des GR de 8,88 points. Les corpus qui profitent le plus des corrections sont d'abord les littéraires et médicaux ; ceux qui nécessitent encore des corrections sont les corpus oraux et de mail. Ces évolutions sont visibles dans la Table 3 entre les *runs* R101 et R139 sur EasyDev, les autres lignes donnant des indications sur le plus long terme.¹³ On constate que les évolutions ne sont pas toujours monotones (R101), des modifications quelque part dans la chaîne de traitement pouvant avoir des effets inattendus (d'où l'importance de mesures de contrôle). Néanmoins, l'évolution sur le long terme est manifestement très positive. Cette tendance est confortée, Table 4, par les résultats de FRMG lors des campagnes d'évaluation **EASy** (2004) et **PASSAGE** (2007).

Campagne	f-mesure groupes	f-mesure relations
2004	69%	41%
2007	89%	63%

TAB. 4 – Résultats des campagnes d'évaluation

Conclusion

La chaîne de traitement autour de FRMG est encore jeune. Même si beaucoup d'améliorations ont pu déjà y être apportées, nous comptons sur les derniers outils de retour d'information dont l'exploitation ne fait que commencer. Nous réfléchissons aussi à une prochaine génération d'outils d'acquisition de connaissances linguistiques travaillant sur gros corpus, par exemple pour l'acquisition de restrictions de sélection et de poids pour la désambiguïsation.

Par ailleurs, diverses pistes pour étendre FRMG sont envisagées :

1. En tant que grammaire développée manuellement, FRMG est moins sensible aux variations de domaine avec, par exemple, des

¹³Ces mesures sont calculées avec nos propres outils d'évaluation qui donnent généralement des valeurs plus faibles de quelques points que les résultats officiels. Enfin, il est à noter que les données de références ont été en partie corrigées au cours des années.

résultats sur de l'oral transcrits qui ne sont pas déshonorants. Néanmoins, l'utilisation de FRMG dans des tâches d'extraction d'information où l'on recherche une couverture très large et une désambiguïsation très précise nous amène à réfléchir à des mécanismes d'adaptation à de nouveaux domaines, en amont avec des phases d'acquisition (incluant, en outre, de la fouille d'erreurs) et en aval par des systèmes de configuration.

2. Passage aux Multi-Component TAGs pour gérer des extractions profondes (comme les génitifs) et des phénomène de portée (comme la négation) en s'appuyant sur les *Thread Automata* dans DYALOG (Villemonte de la Clergerie, 2002b).
3. Intégration d'informations probabilistes pendant la phase d'analyse syntaxique de manière à favoriser la recherche des analyses les plus probables en priorité dans l'optique d'analyses *just-in-time*.

Enfin, précisons que FRMG est un logiciel libre, tout comme SXPipe et le LEFFF. FRMG est accessible sous la GForge de l'INRIA.¹⁴ Il est également possible de jouer avec la chaîne de traitement et de visualiser la grammaire FRMG sur <http://alpage.inria.fr/demos.fr.html>.

Bibliographie

- Anne Abeillé. 2002. *Une grammaire électronique du français*. CNRS Editions, Paris.
- Marie-Hélène Candito. 1999. *Organisation modulaire et paramétrable de grammaires électroniques lexicalisées*. Ph.D. thesis, Université Paris 7.
- J. Carroll, N. Nicolov, M. Smets, O. Shaumyan, and D. Weir. 1998. Grammar compaction and computation sharing in automata-based parsing. In *Proceedings of Tabulation in Parsing and Deduction (TAPD'98)*, pages 16–25, Paris (FRANCE).

¹⁴<http://alpage.inria.fr/software.fr.html>

	% analyse	Groupes			Relations		
	totales	prec.	rappel	f	prec.	rappel	f
R6 (05/2007)	42.16%	78.12%	71.27%	74.54%	62.29%	46.63%	53.34%
R27 (12/2007)	56.06%	83.66%	82.90%	83.28%	64.27%	55.66%	59.65%
R76 (02/2009)	59.47%	84.23%	82.91%	85.56%	63.36%	55.62%	59.24%
R101 (06/2009)	59.56%	83.24%	79.63%	81.40%	63.1%	53.40%	57.85%
R139 (09/2009)	64.73%	87.41%	86.00%	86.70%	65.10%	59.03%	61.92%
R157 (10/2009)	67.03%	87.71%	86.84%	87.28%	65.62%	60.26%	62.82%

TAB. 3 – Évolution des résultats sur EasyDev

- Lionel Clément and Alexandra Kinyon. 2003. Generating LFGs with a metaGrammar. In *Proc. of Lexical Functional Grammars (LFG'03)*. CSLI Publications.
- Benoît CRABBÉ. 2005. *Représentation informatique de grammaires fortement lexicalisées*. Ph.D. thesis, Université Nancy 2, June.
- Christy Doran, Dania Egedi, Beth Ann Hockey, B. Srinivas, and Martin Zaidel. 1994. XTAG system — a wide coverage grammar for English. In *Proc. of the 15th International Conference on Computational Linguistics (COLING'94)*, pages 922–928, Kyoto, Japan.
- D. Duchier, J. Le Roux, and Y. Parmentier. 2005. XMG : un Compilateur de Métagrammaire Extensible. In *Conférence Traitement Automatique des Langues Naturelles (TALN'2005)*, Dourdan.
- Bertrand Gaiffe, Benoît Crabbé, and Azim Roussanaly. 2002. A new metagrammar compiler. In *Proc. of TAG+6*, Venice, Italy.
- Karin Harbusch and Jens Woch. 2004. Integrated natural language generation with schema-tree adjoining grammars. In Christopher Habel and editors Thomas Pechmann, editors, *Language Production*. Mouton De Gruyter.
- Aravind K. Joshi. 1987. An introduction to tree adjoining grammars. In Alexis Manaster-Ramer, editor, *Mathematics of Language*, pages 87–115. John Benjamins Publishing Co., Amsterdam/Philadelphia.
- Lionel Nicolas, Benoît Sagot, Miguel A. Molinero, Jacques Farré, and Eric Villemonte de La Clergerie. 2008. Computer aided correction and extension of a syntactic wide-coverage lexicon. In *Proceedings of Coling 2008*, Manchester.
- Lionel Nicolas, Benoît Sagot, Miguel A. Molinero, Jacques Farré, and Eric Villemonte de La Clergerie. 2009. Trouver et confondre les coupables : un processus sophistiqué de correction de lexique. In *Conférence Traitement Automatique des Langues Naturelles (TALN'20059)*, Senlis.
- Patrick Paroubek, Isabelle Robba, Anne Vilnat, and Christelle Ayache. 2008. Easy, evaluation of parsers of french : what are the results ? In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC)*, Marrakech, Morocco.
- Benoît Sagot and Pierre Boullier. 2008. SxPipe 2 : architecture pour le traitement pré-syntaxique de corpus bruts. *Traitement Automatique des Langues (TAL)*, 49(2) :155–188.
- Benoît Sagot and Éric de la Clergerie. 2008. Fouille d'erreurs sur des sorties d'analyseurs syntaxiques. *Traitement Automatique des Langues (TAL)*, 49(1).
- Benoît Sagot and Éric Villemonte de La Clergerie. 2006. Trouver le coupable : Fouille d'erreurs sur des sorties d'analyseurs syntaxiques. In *Proc. of TALN'06*, pages 287–296. Prix du meilleur papier.
- Benoît Sagot, Lionel Clément, Éric Villemonte de la Clergerie, and Pierre Boullier. 2006. The Lefff 2 syntactic lexicon for french : architecture, acquisition, use. In *Proc. of LREC'06*.
- Yves Schabes and Richard C. Waters. 1995. Tree insertion grammar : a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Fuzzy Sets Syst.*, 76(3) :309–317.
- François Thomasset and Éric Villemonte de La Clergerie. 2005. Comment obtenir plus des métagrammaires. In *Proceedings of TALN'05*, Dourdan, France, June. ATALA.
- Tim van de Cruys. 2006. Automatically extending the lexicon for parsing. In *Proceedings of the eleventh ESSLLI student session*.
- Gertjan van Noord. 2004. Error mining for wide-coverage grammar engineering. In *Proc. of ACL 2004*, Barcelona, Spain.
- Éric Villemonte de la Clergerie. 2001. Refining tabular parsers for TAGs. In *Proceedings of NAACL'01*, pages 167–174, CMU, Pittsburgh, PA, USA, June.
- Éric Villemonte de la Clergerie. 2002a. Construire des analyseurs avec DyALog. In *Proc. of TALN'02*, June.
- Éric Villemonte de la Clergerie. 2002b. Parsing MCS languages with thread automata. In *Proc. of TAG+6*, May.
- Éric Villemonte de La Clergerie. 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, October.
- Zhang Yi and Valia Kordoni. 2006. Automated deep lexical acquisition for robust open texts processing. In *Proceedings of LREC-2006*.
- Éric Villemonte de la Clergerie. 2008. A collaborative infrastructure for handling syntactic annotations. In *proc. of The First Workshop on Automated Syntactic Annotations for Interoperable Language Resources*, Hong-Kong, January.