

FRMG: Évolutions d'un analyseur syntaxique TAG du français

Éric de la Clergerie, Benoît Sagot,
Lionel Nicolas, Marie-Laure Guénot

<http://alpage.inria.fr>

Journée ATALA "Quels analyseurs syntaxiques pour le français"
Paris, 10 Octobre 2009



FRMG est un analyseur TAG du français

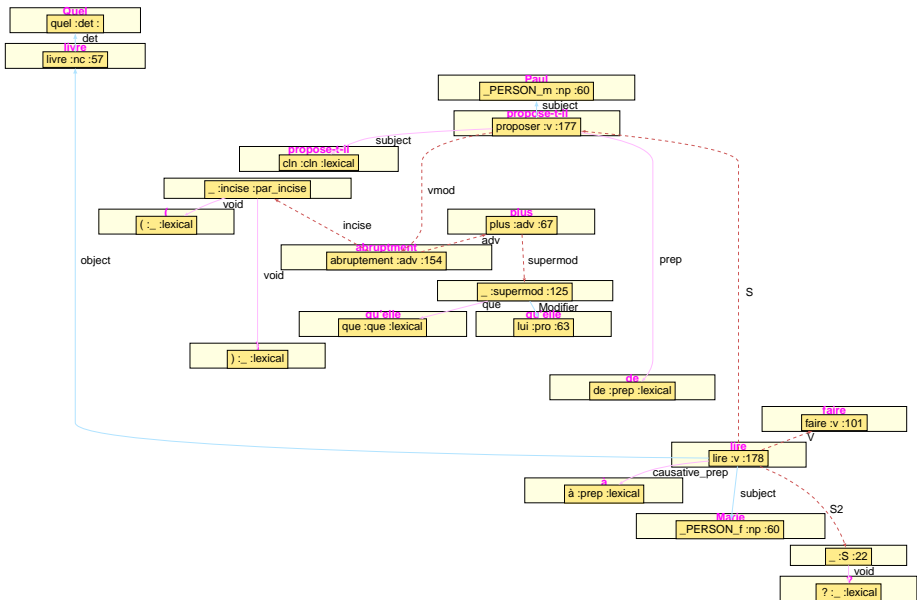
- issu de la compilation d'une méta-grammaire
- très compacte grâce à la factorisation des arbres
- exploitant les fonctionnalités de **DYALOG** environnement de programmation en logique (langage, compilateur, machine virtuel)

FRMG s'intègre dans une chaîne de traitement

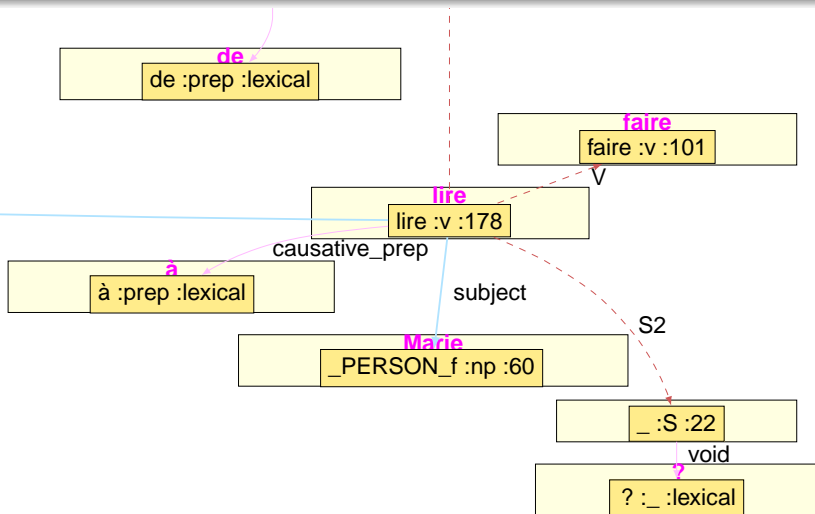
- en amont, avec **SXPIPE** et le lexique **LEFFF**
 - ▶ **SXPIPE** : segmentation, token, corrections, entités nommées retourne un DAG (treillis de mots)
 - ▶ **LEFFF** : lexique morphosyntaxique et syntaxique du français
- en aval, avec un module de désambiguïsation

FRMG et ses compagnons soumis à un processus de retour pour améliorer les performances.

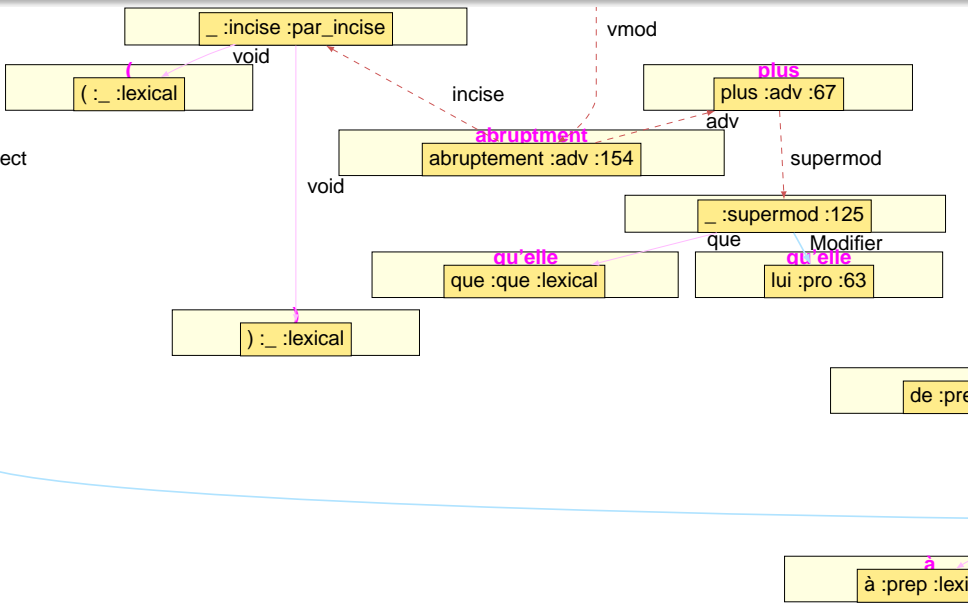
Quel livre Paul propose-t-il (plus abruptement qu'elle) de faire lire à Marie ?



Quel livre Paul propose-t-il (plus abruptement qu'elle) de faire lire à Marie ?



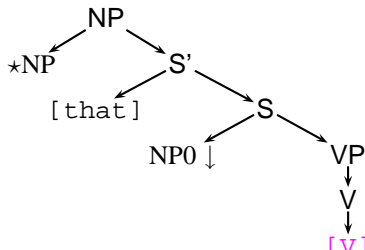
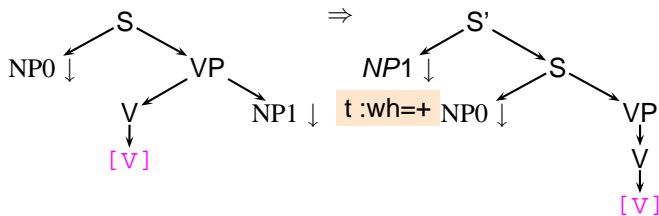
Quel livre Paul propose-t-il (plus abruptement qu'elle) de faire lire à Marie ?



Pb avec les grammaires TAGs

Le domaine étendu de localité des TAG entraîne

- une explosion du nombre d'arbres (plusieurs milliers)
⇒ pb d'efficacité de l'analyse
- beaucoup de sous-arbres en commun
⇒ pb de maintenance



Pour **FRMG**, le choix est :

- utilisation de *Métagrammaire*
description modulaire et factorisée de phénomènes syntaxique
- utilisation d'opérateurs de factorisation de **DYALOG**

Métagrammaire :

- description par un ensemble de classes, prises dans un réseau d'héritage
- un classe exprime des contraintes sur les noeuds de l'arbre d'analyse
dominance, précedence, décorations
- mécanisme de gestion de ressources : fournisseur/consommateur

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

Compiler la méta-grammaire

Compilateur **MGCMP**, développé avec **DYALOG**

Compilateur **MGCMP**, développé avec **DYALOG**

Étape 1 : Classes terminales

Héritage des contraintes par les classes terminales (+ vérif contraintes)

Compilateur **MGCMP**, développé avec **DYALOG**

Étape 1 : Classes terminales

Héritage des contraintes par les classes terminales (+ vérif contraintes)

Étape 2 : Classes neutres

- Croisement des classes terminales pour neutraliser ressources & besoins
 - ▶ $C_1[-R \cup \mathcal{K}_1] \times C_2[+R \cup \mathcal{K}_2] = (C_1 \times C_2)[=R \cup \mathcal{K}_1 \cup \mathcal{K}_2]$
 - ▶ (Espace de nom) \Rightarrow import classe productrice avec renommage
 $C_1[-N::R \cup \mathcal{K}_1] \times C_2[+R \cup \mathcal{K}_2] = (C_1 \times N::C_2)[=N::R \cup \mathcal{K}_1 \cup N::\mathcal{K}_2]$
- Réduction des gardes (quand possible)
- Vérification des contraintes

Compilateur **MGCMP**, développé avec **DYALOG**

Étape 1 : Classes terminales

Héritage des contraintes par les classes terminales (+ vérif contraintes)

Étape 2 : Classes neutres

- Croisement des classes terminales pour neutraliser ressources & besoins
 - ▶ $C_1[-R \cup \mathcal{K}_1] \times C_2[+R \cup \mathcal{K}_2] = (C_1 \times C_2)[=R \cup \mathcal{K}_1 \cup \mathcal{K}_2]$
 - ▶ (Espace de nom) \Rightarrow import classe productrice avec renommage
 $C_1[-N::R \cup \mathcal{K}_1] \times C_2[+R \cup \mathcal{K}_2] = (C_1 \times N::C_2)[=N::R \cup \mathcal{K}_1 \cup N::\mathcal{K}_2]$
- Réduction des gardes (quand possible)
- Vérification des contraintes

Étape 3 : Arbres TAG/TIG

Utilisation des contraintes des classes neutres pour construire les arbres

Idea : putting more in a single tree, because the trees share many common subparts

- defining more than one traversal path per tree (Harbush)
- using regular operators on trees :

disjunctions $T[t_1; t_2] \equiv T[t_1] \cup T[t_2]$

repetitions (Kleene Stars) $t@* \equiv \text{kleene}_t(\epsilon) \cup \text{kleene}_t(t, \text{kleene}_t)$

interleaving (free ordering between node sequences)

$(t_1, t_2)##t_3 \equiv (t_1, t_2, t_3; t_1, t_3, t_2; t_3, t_1, t_2)$

optionality (optional node) $t? \equiv (t; \epsilon)$

guards (guarded nodes) $T[G_+, t; G_-] \equiv T[t].\sigma_+ \cup T[\epsilon].\sigma_-$

guards : boolean formula over equation between feature structure paths

La grammaire en quelques tables

Classes 191	Arbres 133=126+7	Init. 44	Aux. 89	Aux. Env. 12	Aux. Gauches 29	Aux. Droits 48
-----------------------	----------------------------	--------------------	-------------------	------------------------	---------------------------	--------------------------

Distribution par catégories d'arbres

non ancrés 50	v 27	coo 12	adv 10	adj 8	csu 4	prep 3	aux 2	np 2	nc 1	det 1	pro 1
-------------------------	----------------	------------------	------------------	-----------------	-----------------	------------------	-----------------	----------------	----------------	-----------------	-----------------

Distribution par la catégorie de l'ancre

Canonique 7	Extr. 19	Actif 19	Passif 6	Quest. 4	Rel. 4	Clivées 11	Coord 12	Adv 14	Adj 11
-----------------------	--------------------	--------------------	--------------------	--------------------	------------------	----------------------	--------------------	------------------	------------------

Distribution par phénomènes syntaxiques

Gardes 820	Disjonctions 92	Entrelacement 26	Étoiles de Kleene 13
----------------------	---------------------------	----------------------------	--------------------------------

Utilisation des opérateurs de factorisation

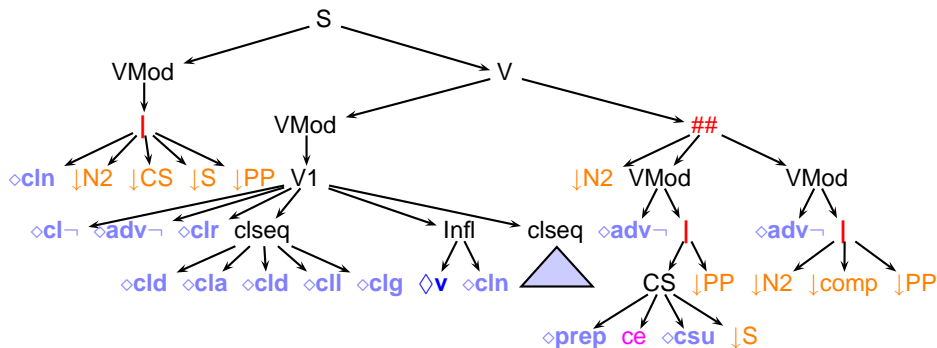
Arbre #111

Arbre #111 : pour la construction canonique du verbe

- croisement de 25 classes terminales ;
- 43 nœuds plus 3 alternatives et 1 entrelacement ;
- 35 gardes ;
- +2000 lignes XML

Arbre #111 : pour la construction canonique du verbe

- croisement de 25 classes terminales ;
- 43 nœuds plus 3 alternatives et 1 entrelacement ;
- 35 gardes ;
- +2000 lignes XML



Factorisation et Familles

- Arbres défactorisés : 1 986 300 dont 1 986 300 pour les verbes (99,98%)
- Arbre #111 = 700 660 arbres défactorisés

taux d'expansion	1	2-9	10 - 99	$\geq 10^2$	$\geq 10^3$	$\geq 10^4$	$\geq 10^5$
#arbres factorisés	76	29	2	1	6	14	6

familles de la forme v_subj_obj_prepobj_active_ -

	LEFFF \cap FRMG	FRMG/LEFFF	LEFFF/FRMG
familles	58	71	32
arbres non-factorisés	775 636	1 210 664	-
sans doublons	149 890	172 662	-
sans que_restr	75 630	77 106	-

En réduisant les familles, sans décorations, sans doublons, sans 'que de restriction' : $\sim 75\,630$ arbres

- compilation d'un analyseur tabulaire, décrivant l'ensemble des parcours d'un arbre sous forme d'une méta-transition 2SA
- modèle d'analyse par suspension-reprise au niveau des noeuds
- analyse un DAG en entrée (SxPIPE+ info LEFFF)
- retourne l'ensemble des analyses complètes
possibilité de retourner des analyses partielles couvrant la phrase
- mode 'just-in-time' en flux
- exploitation lexicalisation des arbres, quand disponible
⇒ filtrage d'une partie de la grammaire
- utilisation d'information coin-gauche
- ...

Grammaire hypertag #111

arg0	arg0	[extracted	-]
			kind	subj	
			pcas	-	
			real	real0 - CS N2 PP S cln prel pri	
arg1	arg1	[extracted	-]
			kind	kind1 - acomp obj prepacomp prepobj	
			pcas	pcas1 + - apres à avec de par ...	
			real	real1 - CS N N2 PP S V adj cla ...	
arg2	arg2	[extracted	-]
			kind	kind2 - prepacomp prepobj prepscomp prepvcomp scomp vcomp wh- comp	
			pcas	pcas2 - + apres à ...	
			real	real2 - CS N N2 PP S ...	
cat	v				
diathesis	active				
refl	refl				

Grammaire hypertag #111

arg0	arg0	[extracted	-]
			kind	subj	
			pcas	-	
			real	real0 - CS N2 PP S cln prel pri	
arg1	arg1	[extracted	-]
			kind	kind1 - acomp obj prepacomp prepobj	
			pcas	pcas1 + - apres à avec de par ...	
			real	real1 - CS N N2 PP S V adj cla ...	
arg2	arg2	[extracted	-]
			kind	kind2 - prepacomp prepobj prepscomp prepvcomp scomp vcomp wh- comp	
			pcas	pcas2 - + apres à ...	
			real	real2 - CS N N2 PP S ...	
cat	v				
diathesis	active				
refl	refl				

Lexique hypertag «promettre»

arg0	[kind	subj -]
		pcas	-	
arg1	[kind	obj scomp -]
		pcas	-	
arg2	[kind	prepobj -]
		pcas	à -	
refl	-			

Grammaire hypertag #111

arg0	arg0	extracted - kind subj pcas - real real0 - CS N2 PP S cln prel pri
arg1	arg1	extracted - kind kind1 - acomp obj prepacomp prepobj pcas pcas1 + - apres à avec de par ... real real1 - CS N N2 PP S V adj cla ...
arg2	arg2	extracted - kind kind2 - prepacomp prepobj prepscomp prepvcomp scomp vcomp wh-comp pcas pcas2 - + apres à ... real real2 - CS N N2 PP S ...
cat	v	
diathesis	active	
refl	refl -	

Lexique hypertag «promettre»

arg0	[kind subj -] [pcas -]
arg1	[kind obj scomp -] [pcas -]
arg2	[kind prepobj -] [pcas à -]
refl	-

Les variables propagent les valeurs des hypertags aux noeuds et gardes

FRMG retourne l'ensemble des analyses sous forme des *forêts partagées de dérivations* :

- chaque étape de dérivation (subst, adj) applique un arbre τ_1 sur un noeud N d'un autre arbre τ_2

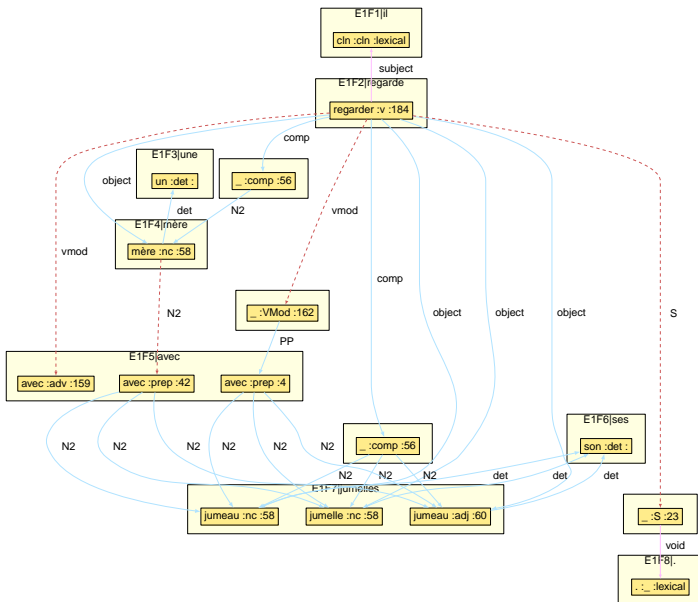
Conversion en une forêt partagée de dépendances

- arcs de la forme $\text{anchor}(\tau_1) \longrightarrow_N \text{anchor}(\tau_2)$
- introduction de pseudo-ancres vides pour les arbres non-lexicalisés

Représentation DEP XML s'appuyant sur

- *cluster* pour les formes
- *node* pour lemmes, pos, arbres ancrés, dérivations associées
- *edge* liant les nodes, associés à un sous-ensemble des dérivation *deriv* du noeud gouverneur
- d'autres informations dont constituants *op*

Forêts partagée de dépendances (exemple)



- algorithme de type 1-best en programmation dynamique, écrit en **DYALOG**
- sommes de poids sur les dépendances
- poids fournis par des règles portant sur la dépendance et ses voisines
- poids manuellement définis
quelques tentatives d'apprentissage
- temps de traitement du même ordre que pour l'analyse

```
%% Penalize inverted subjects (especially for robust mode)
edge_cost_elem( Name:: '-INVERTED_SUBJ',
edge{ label => subject,
      source => node{ cluster => cluster{ right => R } },
      target => node{ cluster => cluster{ left => L } }
},
W
) :-
rule_weight(Name,W,-1000),
R =< L
.
```

- vers formats EASy et Passage :
6 types de *chunks* et 14 types de dépendance
- plus “superficiel” que sorties FRMG
- erreurs dues à la conversion

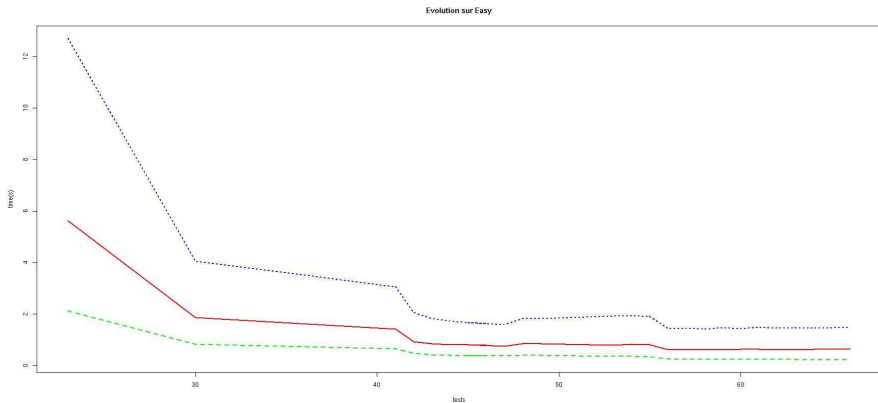
Cela porte sur 4 axes

- augmenter la **couverture** en terme d'analyses complètes
- améliorer la **qualité** des analyses
- réduire le taux d'**ambiguïté** des analyse
- réduire les **temps d'analyse** et de désambiguïsation.

Ces axes sont en partie contradictoires

Jeux	#phrases	Couv.	t. moy. (s)	amb.	couv. 09/09
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	59.46%
Europarl	0.8M	70.19%	1.69	1.36	78.33%
EstRep	1.6M	67.05%	0.69	0.92	75.06%
Wikipedia	2.2M	69.11%	0.49	0.87	79.48%
Wikisource	1.5M	61.08%	0.71	0.89	66.79%
AFP	1.6M	52.15%	0.51	1.06	

Évolutions vitesse FRMG



Easydev : 3868 phrases, 58.20% couverture, timeout (20s) : 0.4%

- Pour améliorer la couverture, recherche des manques dans le lexique, la grammaire, ...
- traitement de gros corpus suivi de fouille d'erreurs
 - ▶ identifier les mots trop souvent présents dans des phrases non analysables
 - ▶ surtout si en co-occurrence avec des mots sans problèmes
 - ▶ processus itératif de type EM
 - ▶ fournit des phrases où le mot est le principal suspect
- possibilité de suggérer des corrections :
 - ▶ ré-analyse les phrases en sous-spécifiant le suspect
 - ▶ fait ressortir les analyses les plus fréquentes devenues possibles au niveau du suspect

Browsing errors for results5 [iter=200]

27 voilà
28 lui-même
29 jusque
30 emparé
31 p.
32 endettés
33 il est vrai que /il est vrai que
34 demeure
35 =
36 azimuts
37 50 /à
38 rase
39 the /thé
40 dus
41 eux-mêmes
42 elle-même
43 coopérer
44 notamment
45 soucie
46 demeuraît
47 monsieur
48 censé
49 autorisée
50 censée
51 quant aux /quant à
52 rend
53 censés
54 quoi
55 taliban
56 disputent
57 prospères
58 d'en bas /en bas
59 endetté
60 qu'à /uw
61 Et /et

Enter rank (or start:end:key) [Mail this page](#)

[edit comment](#)

manque la construction attributive (demeurer<subj,acomp>)

Statistical info on **demeure/demeure**

rank	#occ.	#failed	%failed weight	%failed sentences	orate
34	870	706	24.64%	81.15%	7.27

history:

#iteration	200	199	195	185	175	165	155	145	135	125	115	105	95	90
weight	24.64%	24.64%	24.64%	24.65%	24.65%	24.66%	24.68%	24.69%	24.71%	24.73%	24.75%	24.78%	24.81%	24.83%

Lefff info for **demeure**

```
nc [pred='demeure _____ l<{subj},(de-ob)},(de-vcomp[à-vcomp]>',cat=nc,#fs]
v [pred='demeurer _____ l<sub>>',cat=v,#imperative,#Y2s]
v [pred='demeurer _____ l<sub>>',cat=v,#PS13a]
```

Failed sentences with **demeure/demeure** as most probable cause for failure

- [mondediplo_01#19948] L'armée **demeure** une force majeure
- [mondediplo_02#22126] LE FN **demeure** l'unique parti à défendre les négationnistes dans son programme .
- [mondediplo_04#7744] Le pétrole **demeure** l'enjeu principal .
- [mondediplo_01#19379] L'EUROPE **demeure** un projet à deux vitesses .
- [mondediplo_01#19984] Certes , l'Indonésie **demeure** la grande puissance régionale .
- [mondediplo_01#28830] L'histoire **demeure** cependant la principale discipline d'enseignement .
- [mondediplo_05#15643] En mer Rouge et dans la corne de l'Afrique , la situation **demeure** très incertaine .
- [mondediplo_02#17949] Le père **demeure** le chef exclusif de la famille .
- [mondediplo_04#10602] Une question toutefois **demeure** obscure .
- [mondediplo_06#19376] Le suédois **demeure** la deuxième langue officielle du pays .
- [mondediplo_06#20791] Quant à la Chine , elle **demeure** un grave sujet d'inquiétude .
- [mondediplo_06#31057] Or le social **demeure** une pièce rapportée de la construction européenne .
- [mondediplo_05#26084] Elle **demeure** nécessaire et enrichissante .

- Améliorer la qualité en exploitant des données de référence (treebank) données EASy
- résultats globaux, par type de corpus, groupes et relations
- visualisation des résultats (**EASYREF**)

Matrice de confusion

réf \implies hyp	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%)

Nouveau : Matrices de différences entre runs.

Évolution des performances

	% analyse totales	Groupes			Relations		
		prec.	rappel	f	prec.	rappel	f
R6 (05/07)	42.16%	78.12%	71.27%	74.54%	62.29%	46.63%	53.34%
R27 (12/07)	56.06%	83.66%	82.90%	83.28%	64.27%	55.66%	59.65%
R76 (02/09)	59.47%	84.23%	82.91%	85.56%	63.36%	55.62%	59.24%
R101 (06/09)	59.56%	83.24%	79.63%	81.40%	63.1%	53.40%	57.85%
R139 (09/09)	64.73%	87.41%	86.00%	86.70%	65.10%	59.03%	61.92%
R157 (10/09)	67.03%	87.71%	86.84%	87.28%	65.62%	60.26%	62.82%

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

Note : Nos outils d'évaluation donnent des valeurs plus faibles que les valeurs officielles.

FRMG encore jeune (création en 2004)

- forte évolutions des performances
- de nouveaux outils d'amélioration envisagés
dont utilisation méthodes d'apprentissage non-supervisées
- (futur) utilisation de proba pendant l'analyse
- (futur) passage aux MC-TAGs (extractions profondes, portées)

- Librement disponible sur le site d'ALPAGE
- Utilisable en ligne
- La grammaire est aussi accessible en ligne