Desiging and improving FRMG, a wide coverage French meta-grammar

http://alpage.inria.fr

Éric de la Clergerie
<Eric.De_La_Clergerie@inria.fr>

INRIA Paris-Rocquencourt / Univ. Paris Diderot

A Coruña – 9 de octubre 2012

Introduction

FRMG is a wide coverage French meta-grammar

- fast initial development in 2005 (EASy campaign)
- continuous improvement since then
- now usable at large scale with good coverage and accuracy
- online demo at `http://alpage.inria.fr/parserdemo`
- **note:** existence of SPMG for Spanish (Victoria project)

Objectives of this talk:

1. provide some background on TAGs, tree factoring and meta-grammars
2. present FRMG
3. illustrate the descriptive power of meta-grammars on a class of complements
4. present some results on the improvement of FRMG

Outline

1. Designing
2. Using
3. Improving

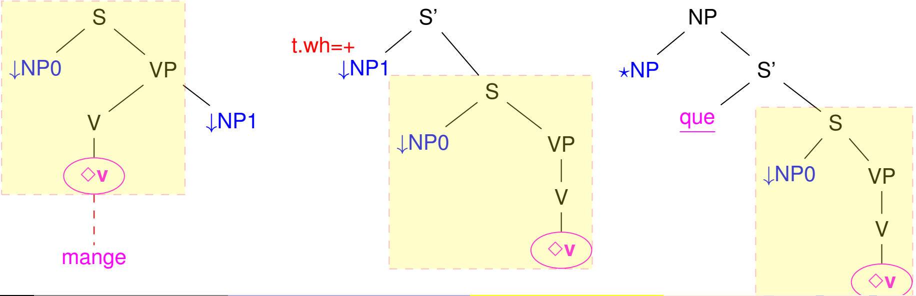
Pro and cons of TAGs

Tree Adjoining Grammars provide the advantage of extended domain of locality:

- subcategorization frames
- long distance dependencies (extractions/movements)

but the drawback is

- an explosion of the number of trees: several (tens of) thousands
  $\Longrightarrow$ efficiency problems during parsing
- many common sub-trees
  $\Longrightarrow$ development and maintenance problems



Solutions

For FRMG, the choice is

- to use factoring operators within trees
  but difficult to directly write complex factorized trees
- to use a Metagrammar
  modular and factorized description of syntactic phenomena
  $\rightsquigarrow$ generation of factorized trees from the descriptions

Tree factoring

Principe : combining several trees into a single one, sharing common subparts

- several traversal paths in a tree (Harbush)
- or using regular operators within trees (DyALog):

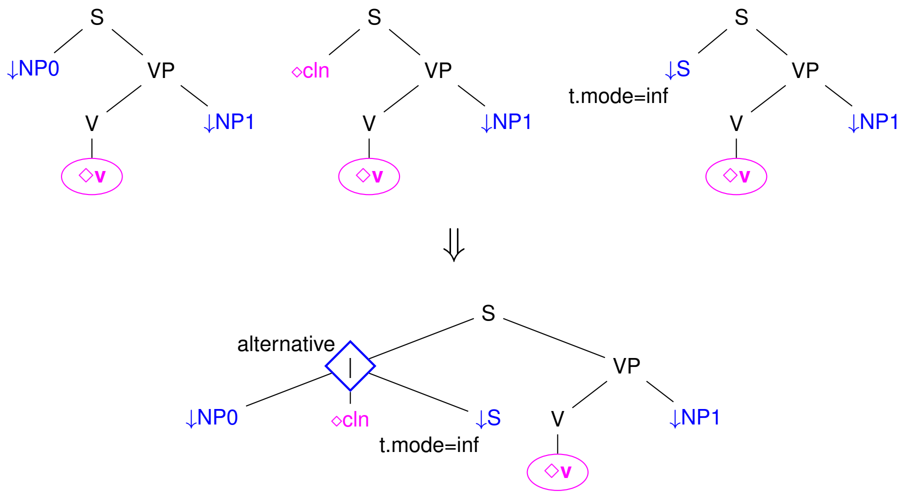
disjunctions	$T[t_1;t_2] \equiv T[t_1] \cup T[t_2]$
repetitions	(Kleene Stars) $T[t@*] \equiv \{T[\epsilon], T[t], T[(t,t)], \ldots\}$
interleaving	(free ordering within sequences) $(t_1, t_2)\#\#t_3 \equiv (t_1, t_2, t_3; t_1, t_3, t_2; t_3, t_1, t_2)$
optionality	$t? \equiv (t; \epsilon)$
guards	(node with guards) $T[G_+, t; G_-] \equiv T[t].\sigma_+ \cup T[\epsilon].\sigma_-$ guards: Boolean formulae on equations on feature path and values

Tree factoring

- does not change the expressive power or the complexity of TAGs
- but unfactoring $\Longrightarrow$ exponential increase of the grammar
- used directly by DyALog

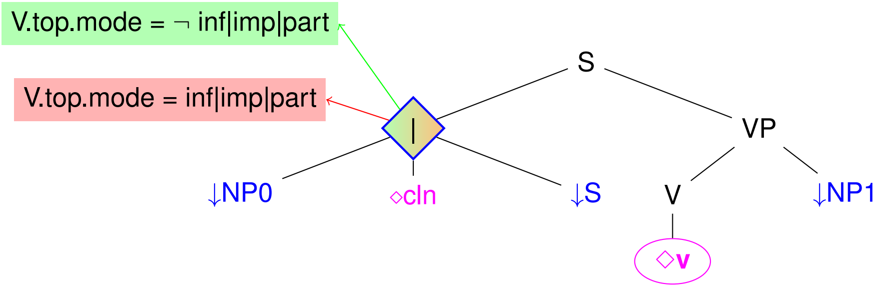
Disjunction

Several possible realizations for the subject (NP, cln, S, . . .)



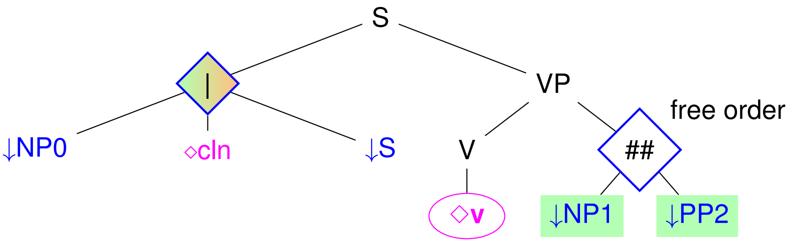
Guards

In French, the subject may be missing, under conditions



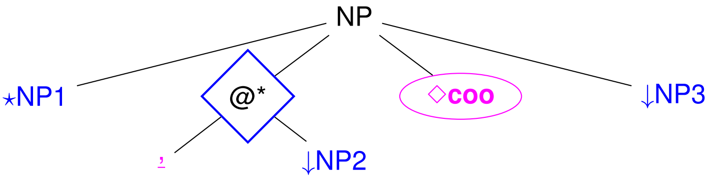
Free ordering

The verbal arguments are not ordered (rough approximation)



Unfactoring: $(1_{\text{no subj}} + 3_{\text{subj}}) * (1_{\text{no arg}} + 2_{\text{1 arg}} + 2_{\text{2 args}}) = 20$ trees

Natural description of coordination by repetition:



Meta-Grammars

Definition (Meta-grammar)

Modular Description with classes grouping constraints, and with inheritance

Meta-Grammars

Definition (Meta-grammar)

Modular Description with classes grouping constraints, and with inheritance

```
class collect_real_subject_canonical {
 <: collect_real_subject;
 $arg.extracted = value(~cleft);
 S >> VSubj; V >> psubj;
 VSubj < V; VMod < psubj;
 node psubj: [cat:N2, id:subject,
              top:[wh:-,sat:+]];
 - psubj::agreement; psubj = psubj::N;
 psubj =>
  node(Infl).bot.inv = value(+),
  $arg.extracted = value(-),
  $arg.real = value(N2),
  desc.extraction = value(~-),
  node(V).top.mode= value(~inf|imp|...);
 ~psubj=> node(Infl).bot.inv = value(~+);
}
```

- Inheritance (<:)
- Constraints
  - dominance (>> et >>+)
  - precedence (<)
  - equality (=)
  - Decorations (FS)
    - nodes
    - class
  - Eq. between pathes (.)
    - node (node psubj)
    - class (desc)
    - var ($arg)
- Resources + / Needs −
  - Namespace (::)
- Guards (=>)

Compiling Meta-Grammars

MGCOMP, developed with DYALOG

Step 1: Terminal classes

Constraint inheritance by the terminal classes (+ constraint checking)

Step 2: Neutral classes

- Crossing of terminal classes to neutralize resources & needs
  - $C_1[-R \cup \mathcal{K}_1]xC_2[+R \cup \mathcal{K}_2] = (C_1xC_2)[=R \cup \mathcal{K}_1 \cup \mathcal{K}_2]$
  - (Namespace) $\Longrightarrow$ import producing classe with renaming
    $C_1[-N{::}R \cup \mathcal{K}_1]xC_2[+R \cup \mathcal{K}_2] = (C_1xN{::}C_2)[=N{::}R \cup \mathcal{K}_1 \cup N{::}\mathcal{K}_2]$
- Guard reduction (whenever possible)
- Constraint checking

Step 3: TAG Trees

Use of constraints of neutral classes to build trees

- underspecified precedence between sibling nodes $\Longrightarrow$ interleaving

FRMG: A French meta-grammar

- Verbe subcategorization : subject `subj`, attribute `acomp`, `object`, `vcomp`, `scomp`, `wh-comp`, `prep-vcomp`, `prep-scomp` `prep-object`, `prep-acomp`
  at most 3 arguments (subject incl.)
- Aux. verbs, control verbs
- Various realizations (NP, clitics, infinitive, completive, . . .)
  and subject position (pre, post, post-clitics)
- extraction of arguments and adjuncts (wh, relatives, clefted, topicalisation)
- active and passive voices
- coordination (without ellipses), comparatives, superlatives
- verb/sentence modifiers (with incises) at various positions
  (participle sentences, PP, adv, . . .),
- «support» verbs (prendre conscience de)
- punctuation

An use case: handling clause complements

A large and heterogeneous class of complements, difficult to characterize:

- they are **modifiers** bringing information on many aspects (tense, duration, space, manner, cause, intensity, quantity, . . .)
- they have many realizations: adverbs, prep. phrases, conjonctive subordinates, participials, adjectival phrases, (idiomatic) clauses, concessives, . . .
- they are **mobiles** in a clause, with several possible anchoring points (clauses, coordination, prepositions, event nouns, . . .)
- they are **parenthesables** (parenthesis, coma, dash, . . .), in a more or less mandatory way depending of the complement and position
- they include idiomatic constructions, and are often semantically restricted (tense, space, bodypart, . . .)
  $\Longrightarrow$ importance of semantic features provided by the lexicon (LEFFF)

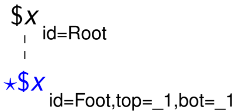
A few illustrative sentences covered by FRMG

Il a **parfois** envie de partir.
Désormais, il veut partir.
Avec son ami, il a décidé de partir **sans tarder**.
Il est arrivé **pendant que tu parlais**.
Sa société n'allant pas bien, il doit la vendre.
Il prend le train, **soucieux des deniers publics**.
Il a, **le premier**, fini l'exercice.
Mains sur la tête, il recule contre le mur.
Couloir de droite, vous avez la classe de Mr Louis.
Vous trouverez, **chapitre 22**, les explications nécessaires à ce devoir.
Il attend, **rue des Bourdonnais**, que ses amis arrivent.
Il a, **lui aussi**, décidé de partir.
Il est parti, **il n'y a pas deux jours**, avec des amis.
Il est parti, **voici deux semaines**, avec des amis.
Service oblige, je dois vous quitter.
Il a, **quoi que tu en penses**, toutes ses chances.
Paul a, **plus que son frère**, le sens de la famille.
Il mange une pomme et, **parfois**, une poire.
Il part, avec, **toutefois**, une pointe de regret.
L'annonce, **ce matin**, d'un remaniement a surpris tous les commentateurs.

TAG auxiliary trees

Clause complements are modifiers, hence represented by TAG auxiliary trees, with a foot node sharing a common category with the root node.

```
class auxiliary {  % Class for TAG auxiliary trees
   Root >>+ Foot;                         % root dominates foot
   node(Root).cat = node(Foot).cat; % same cat. on foot and root
   node Root : [id:Root];
   node Foot : [id:Foot];
   node(Foot).type = value(foot);
   node(Root).type = value(std);
   node(Foot).top = node(Foot).bot;
}
```



shallow auxiliary

Actually, shallow aux. trees are sufficient: the root node is a parent of the foot node

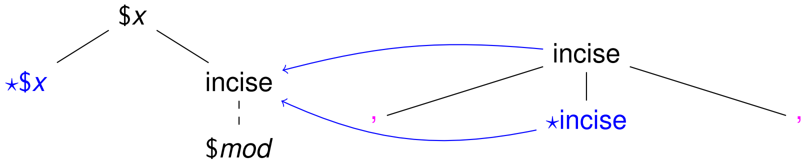
```
class shallow_auxiliary {
    <: auxiliary;
    + shallow_auxiliary;    %% provide functionality
    Root >> Foot;           %% root is parent of foot
}
```

Modifier on category X

Clause complements are modifiers (on a catégory X to be precised). Modifiers may be preceded and followed by incise marks.

> ***Avec son ami****, il a décidé de partir* ***sans tarder***
> *il a (****souvent****) faim.*

```
class modifier_on_x {
  + x_modifier;
  - shallow_auxiliary;      % require functionality
  Root >> Incise;           % root parent of node incise
  node Incise : [cat:incise, id:incise, type: std];
  Incise >>+ Modifier;      % incise dominates the modifier
  % incise_kind controls the marks
  node(Incise).bot.incise_kind = value(coma|comastrict|par|dash);
}
```

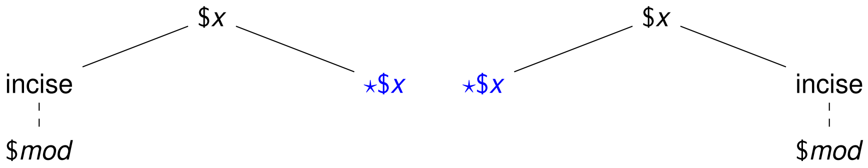


Position of modifiers wrt the modified

Modifier may precede or follow the modified, with possibly distinct properties.

> ***Désormais***, *Paul part plus tôt.*
> *Paul est parti tard* ***ce matin***.

```
class modifier_before_x {
   <: modifier_on_x;
   Incise < Foot;        % ante modifier
   node(Incise).adj = node(Incise).ante.adj;
   desc.position = value(ante);}
class modifier_after_x{
   <: modifier_on_x;
   Foot < Incise;      % post modifier
   node(Incise).adj = node(Incise).post.adj;
   desc.position = value(post);}
```

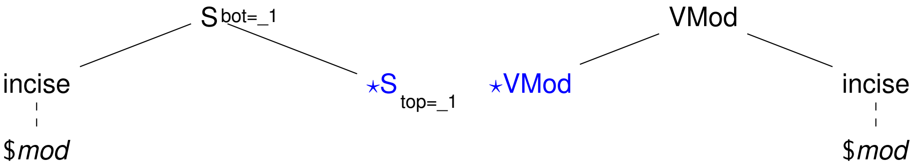


Attach points

Clause complements may attach on the clause root (beginning or end of a clause), but also on various attach points provided by **VMod** nodes.

> ***ce matin***, *Paul part tôt.*
> *Paul,* ***ce matin***, *part très tôt.*

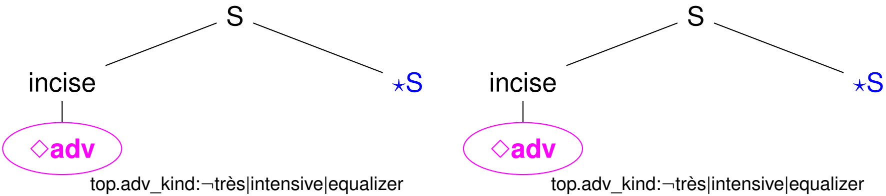
```
class modifier_at_S_level {
  + s_modifier;
  - x_modifier;
  - s_adj_pos;
  node(Root).bot = node(Foot).top;  %% same info on Root and Foot
  node(Root).cat = value(S|VMod);}   %% may adjoin on S or VMod
class modifier_on_S {                %% specialization for S
  + s_adj_pos;
  node(Root).cat = value(S);}
```



Realization: adverbs

We can now implement the many realizations for clause complements
the most frequent is provided by (some) adverbs

```
class adv_s {   %%  Adverbs on sentences
  <: adv;        %% inherit properties for adverbs
  - s_modifier; Adv = Modifier;
  node(Foot).dummy.cat = value(adv);
  %% restrictions on allowed adverbs
  node(Adv).bot.adv_kind = value(~très|intensive|equalizer);
  ...
}
```

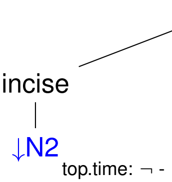


Réalisation: adverbiaux temporels

Another frequent realization provided by the temporal nominal phrases

*il a, **cet après-midi**, fini sa présentation.*

```
class cnoun_as_adv {
  node N2: [cat: N2, id:time_mod, type: subst];
  %% N2 should carry a time property (other than -)
  node(N2).top.time = value(~ -);
  - s_modifier; N2 = Modifier;
  node(Foot).dummy.cat = value(adv); %% N2 acts as an adv
}
```



Realization: body part

We have more exotic realizations, with semantic restrictions and idiomatic constructions:

> ***cheveux au vent**, il fait face dans la tempête.*
> *il me fit promettre, **les mains sur la Bible**, que je viendrais*
> *Mme Bovary, **le dos tourné**, avait la figure posée contre un carreau*

```
class bodypart_cnoun_as_modifier {
 <: cnoun;          %% inherit from nominal phrase
 desc.@kind0 = value(-);
 - s_modifier; Modifier = N2Root;
 node(Foot).dummy.cat = value(adv); %% s_modifier acting as adv
 node(Root).cat = value(~ N2);      %% not attached on event nouns
 node(Root).bot = node(Foot).top;
 node(Anchor).bot.semtype = value(bodypart); %% semantic property
 node(N2).adj = value(strict);      %% mandatory right adjoining
 node(N2).adjleft = value(no);
 node(N2).adjwrap = value(no);
}
```

Realization: temporal with "il y a"

We have idiomatic constructions, frozen or semi-frozen, like *il y a* or *X oblige*, that should inherit from verbal structures.

> *elle a été adoptée **il n'y a pas trois mois**.*
> *je ne conclurai pas, **exception culturelle oblige**, par une citation.*
> *Il est parti, **voici deux semaines**, avec des amis.*

```
class verb_ilya_as_time_mod {
  <: _verb_canonical;        %% inherits canonical verb construction
  - s_modifier; S = Modifier;
  node v:[cat:v, lex: avoir];
  desc.ht.imp = value(+);  %% impersonal subject expected
  desc.ht.extraction = value(-); %% no argument extraction
  ... }
```

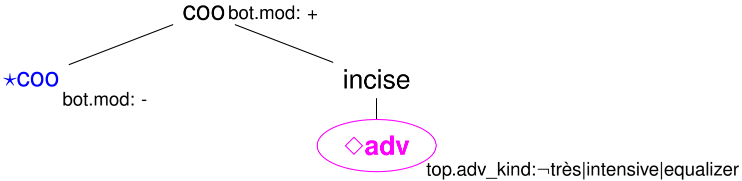
But how to capture a large number of such idiomatic constructions

Attach points

New attach points are regularly found and easily added, as illustrated for coordinations (similar after prepositions):

*Paul mange une pomme et, **parfois**, une poire. le texte est signé par le Premier ministre et, **le cas échéant**, par les ministres responsables.*

```
class modifier_on_coo {
  + s_adj_pos;
  node(Root).cat = value(coo);
  desc.position = value(post); %% modifier on the right side of coo
  %% restriction: no more than one modifieur per coo node
  node(Foot).top.modifier = value(-);
  node(Root).bot.modifier = value(+);
}
```

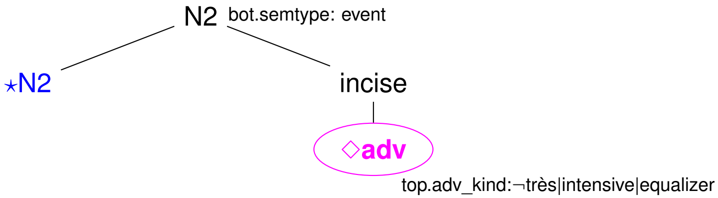


Attaching on event nouns

And also after event nouns (semantic constraints on the governor)

> *L'annonce, **ce matin**, d'un remaniment a surpris tous les commentateurs*

```
class mod_on_N2 {
  + s_modifier;
  - x_modifier;
  - s_adj_pos;
  node(Root).cat = value(N2);
  node(Root).bot.semtype = value(event); %% semantic property event
}
```



Results: a few figures

The various combinations (realizations $\times$ attach point $\times$ positions) generate a relatively large number of trees: **79**

To be contrasted with only 42 trees anchored by verbs (canonical constructions + passive + extractions [rel, wh, clefted, topic]).

However, the trees for the clause complements are generally quite simple

A count of the trees used for parsing of 10,096 sentences of the French TreeBank (journalistic style) $\Longrightarrow$

79 trees	$\sim$ 8%
PP on Vmod	4.3%
PP on S (beginning of sentence)	0.8%
temporal NP	0.35%
conj. subordinate	0.13%

We observe a fast decrease of the frequencies

Quantitative FRMG (2012/10/07)

Classes	Trees	Init.	Aux.	Wrap Aux.	Left Aux.	Right Aux.
279	333+33	78	288	45	83	159

Distribution per tree kinds

anchor	v	coo	adv	adj	csu	prep	aux	np	nc	det	pro	¬anchor
229	42	42	44	21	9	10	2	3	25	1	6	104

Distribution per anchor kind

Canonical	Extr.	Active	Passive	Quest.	Rel.	Cleft	Topic
8	18	25	9	3	3	7	5

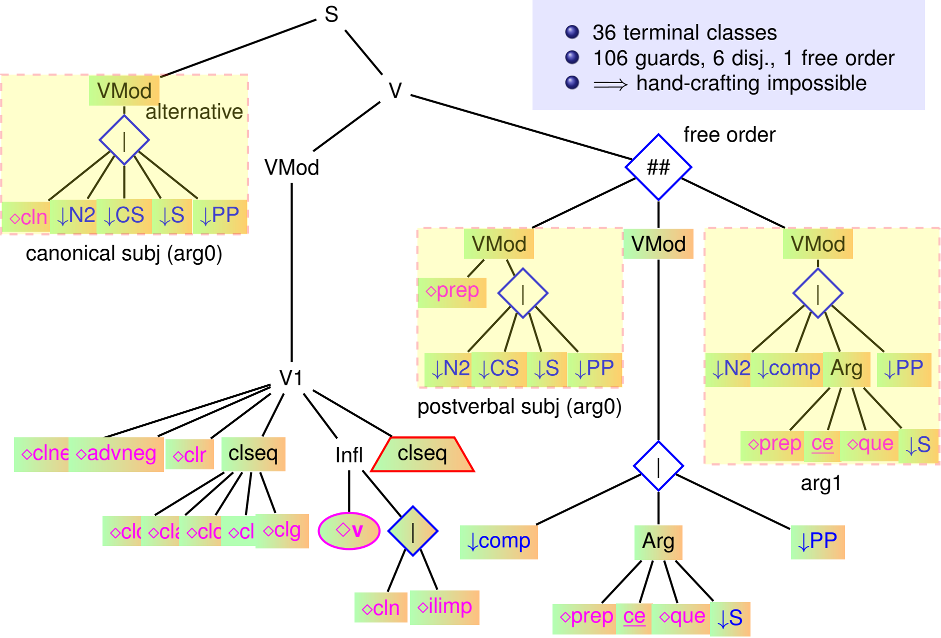
Distribution per syntactic phenomena

Guards	Disjunctions	Interleaving	Kleene star
3853	214	40	48

Use of factoring operators

Tree #198: canonical verb (simplified)

- 36 terminal classes
- 106 guards, 6 disj., 1 free order
- ⟹ hand-crafting impossible



lexicon/grammar interaction: *hypertag*

Grammaire FRMG

hypertag #198

- 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**

lexicon/grammar interaction: *hypertag*

Grammaire FRMG
hypertag #198

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 LEFFF
hypertag «promettre»

arg0 [kind **subj | -**, pcas **-**]

arg1 [kind **obj | scomp | -**, pcas **-**]

arg2 [kind **prepobj | -**, pcas **à | -**]

refl **-**

lexicon/grammar interaction: *hypertag*

Grammaire FRMG — hypertag #198

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** | prep-scomp | prepvcomp | scomp | vcomp | whcomp
		pcas	**pcas2**	**-** | + | apres | **à** | ...
		real	**real2**	**- | CS | N | N2 | PP | S | ...**
cat	**v**			
diathesis	**active**			
refl	**refl**	**-**		

Lexique LEFFF — hypertag «prometttre»

arg0	kind	**subj | -**
	pcas	**-**
arg1	kind	**obj | scomp | -**
	pcas	**-**
arg2	kind	**prepobj | -**
	pcas	**à | -**
refl	**-**	

The variables x propagate the hypertag values to nodes and guards

⟹ block non valid paths in the factored trees

Degree & impact of factoring

- complete tree unfactoring $\Longrightarrow \sim$ 2 millions trees (FRMG 2007)
- partial unfactoring on #198 (keeping disjunctions)
  + intersection with LEFFF 195 subcat frames $\Longrightarrow$ 5729 trees (+ 206)
- too large for computing the left-corner relation
- test of the 2 versions on 4000 **EasyDev** sentences
  $\Longrightarrow$ **factoring**: no overhead and better optimization

parser	avg	median	90%	99%
+fact. -lc (207 trees)	**1.33**	0.46	2.63	12.24
-fact. -lc (5935 trees)	1.43	0.44	2.89	14.94
+fact. +lc (207 trees)	**0.64**	0.16	1.14	6.22

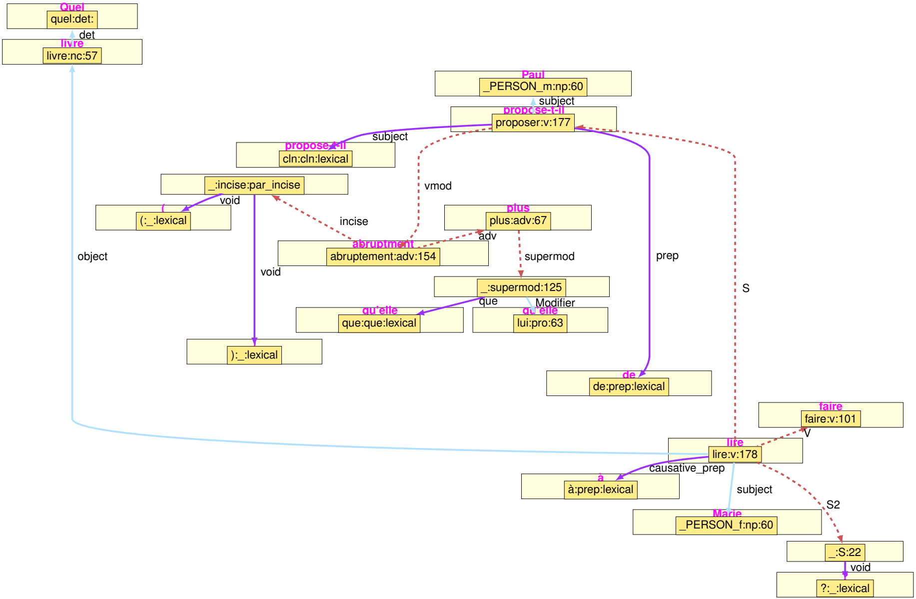
Outline

1. Designing
2. Using
3. Improving

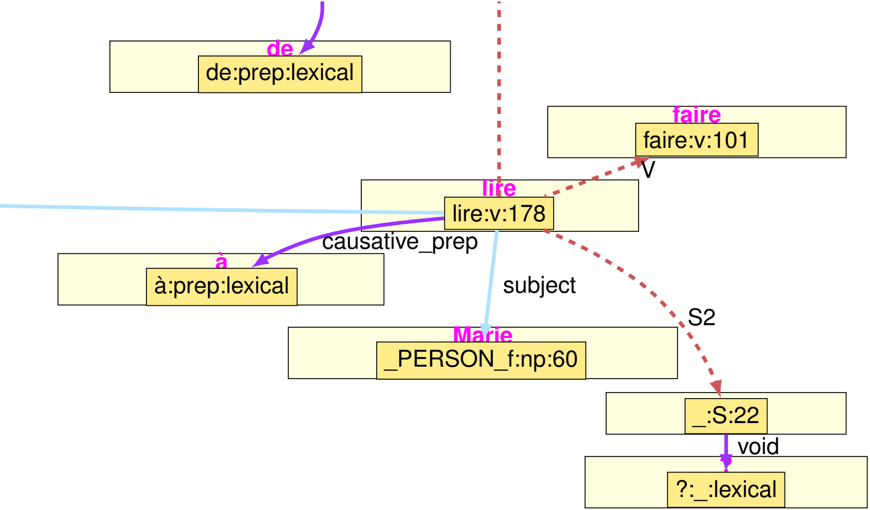
Parser functionalities

- detection of TIG (left and right) aux. trees et wrapping TAG aux. trees
- compilation of a tabular-based parser, using 2SA meta-transitions to describe tree traversals
- DAG of words in input (SxPipe+ lex. info Lefff)
- return a shared forest of all possible analysis
  may try some corrections (relaxation of agreement)
  possibility to return partial parses when no full parses
- 'just-in-time' mode
- lexical tree filtering (only a part of the grammar is loaded)
- use of the left-corner relation
- identification of features untouched by adjoining
- ...

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



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



Disambiguation

The shared dependency forest may be disambiguated :

- 1-best dynamic programming rule-based algorithm (in DyALog)
- summation of weights on each dependency
- the weights provided by heuristic rules on the dependency and neighbour dependencies
- hand-crafted weights on the rules
- time costly (similar to parsing or worse !)

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

Outline

1. Designing
2. Using
3. Improving

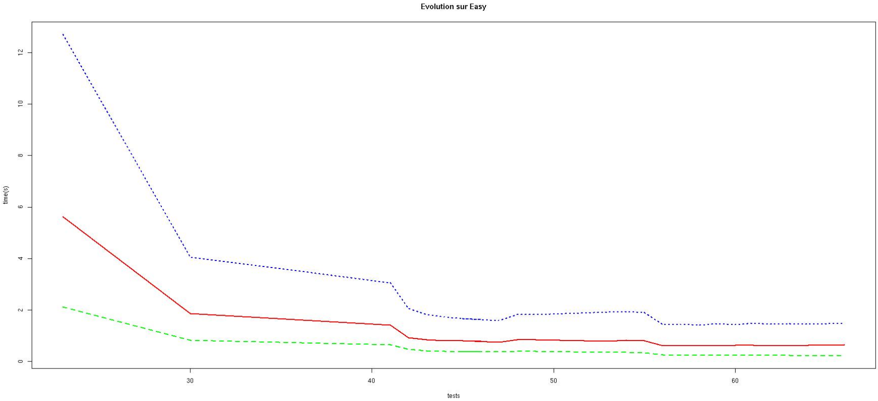
Improving FRMG

Try to improve along 3 axes:

- increasing the **coverage** in terms of full parses
  $\Longrightarrow$ unsupervised machine learning methods on large corpora: error mining
- improving parsing **accuracy**
  $\Longrightarrow$ supervised methods, using treebanks
- reducing **parsing times** and disambiguation time.
  important but not essential: use of machine grids (GRID5000)

These axes are partially contradictory.

Evolution of efficiency (during 2008)



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

- Multiples runs on test suites and corpora, with statistics
- many optimizations tested, most of them useless
- but unstable gains: important variations wrt the grammar

Improving the accuracy (old way)

Through the exploitation of reference annotations (treebanks)

- EasyDev (Easy format: 6 kinds of chunks and 14 kinds of relations)
- (in 2012) French TreeBank (FTB) in CONLL dependency format

Methods:

- Evaluations and feedback
- Exploration of the annotated sentences
- Confusion Matrices (EASy chunks and relations; CONLL dependencies)

réf ⟹ 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%)**

- also difference matrices between 2 runs

Performance evolution on EsyDev

	% parses	Chunks			Relations		
	full	prec.	recall	f	prec.	recall	f
R006 (05/2007)	42.16%	78.12%	71.27%	74.54%	62.29%	46.63%	53.34%
R027 (12/07)	56.06%	83.66%	82.90%	83.28%	64.27%	55.66%	59.65%
R076 (02/2009)	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 (09/2009)	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%
R240 (11/2010)	69.01%	88.24%	89.20%	88.72%	66.36%	63.32%	64.81%
R374 (12/2011)	82.57%	89.46%	89.90%	89.68%	67.81%	66.17%	66.98&
05/2012	83.58%	89.15%	89.43%	89.29%	69.56%	67.50%	68.52%

Campaign	f-measure chunks	f-mesure relations
2004	69%	41%
2007	89%	63%

Note: Our evaluation tools provide lower figures than the official tools

Coverage on several corpora (2010)

Corpus	#sentences	Cov.	t. avg. (s)	amb.	avg. 09/09
EUROTRA	334	100%	0.09	0.81	10/10
TSNLP	1661	95.18%	0.04	0.48	11/10
EasyDev	3879	69.01%	0.46	1.10	11/10
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	

Improving accuracy with FTB (new approach)

French TreeBank: Journalistic style (LeMonde), with Penn TreeBank like annotations [Abeillé]
converted to CONLL dependencies [Candito & al]

Preliminary results with FRMG

Corpus	#sent.	Cover. (%)	Time (s)
ftb train	9,881	91.22	0.70
ftb dev	1,235	90.68	0.59
ftb test	1,235	90.59	0.56
ftb all	**12,351**	**91.11**	**0.72**

(run `inline.old43` 2012-06-09)

CONLL format (converted from FRMG dependencies)

id	form	lemma	POS	head	label
1	par	par	P	15	p_obj
2	qui	qui	PRO	1	obj
3	a	avoir	V	5	aux_tps
4	-t-elle	-t-elle	CL	5	suj
5	voulu	vouloir	V	0	root
6	que	que	C	5	obj
7	ces	ce	D	9	det
8	deux	deux	A	9	mod
9	livres	livre	N	15	suj
10	et	et	C	9	coord
11	ce	ce	D	12	det
12	DVD	DVD	N	10	dep_coord
13	lui	lui	CL	15	a_obj
14	soient	être	V	15	aux_pass
15	rendus	rendre	V	6	obj
16	?	?	PONCT	5	ponct

Disambiguation tuning

Conversion to CONLL format originally done for evaluation purpose on FTB.

But can we use it to tune the weights of the disambiguation rule ?

- No simple one-to-one mapping from FRMG dependencies to FTB dependencies (no inverse conversion)
- Minimal hypothesis: if the FTB dependency on occurrence $w_{i,j}$ is OK then the FRMG dependency $\hat{d}_{i,j}$ on $w_{i,j}$ (kept when disambiguating) should also be OK, while the other $d' \in D^{\neg}_{i,j}$ are bad
  $\Longrightarrow$ should reinforce when correctly kept (or unkept), and penalize when incorrectly kept (or unkept)

Actually 4 cases:

	kept	unkept
ok	reinforce	penalize
bad	penalize	???

Semi-supervised Machine Learning

For the 4th case, we need an oracle indicating which dependency $d' \in D^{\neg}_{i,j}$ should have been kept:

- no ambiguities when only one remaining dependency d'
- use statistics to determine d'
- (recent) information from the reference (samehead, sametype, ...)

For a disambiguation rule τ and set of features $\mathcal{S}$, we determine how often it was OK to keep or discard the dependencies d satisfying τ and $\mathcal{S}$.

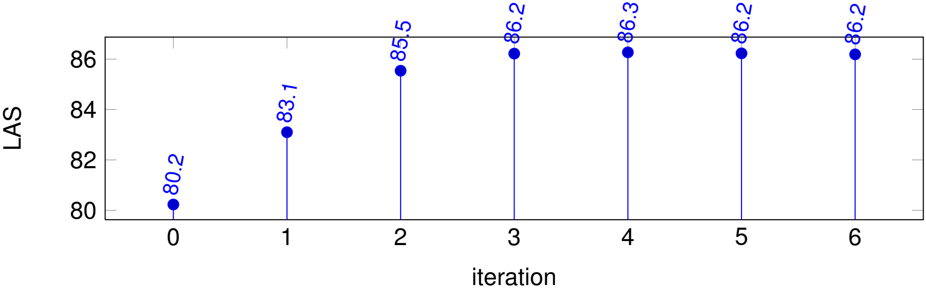
```
rule          #   delta(s)      w      k+    k-   k-good?   u+    u-  u-good?    features
+XCOMP_TOPIC 88 -6.0/-27.0  +1994.8 6.8   6.1 {+2.8}   62.5 24.6 {+21.8} label=xcomp type=subst +slemma=dire +scat=v
+XCOMP_TOPIC 85 +46.0/+28.0 +2200.2 23.5 2.7 {+39.8}  67.1 6.7  {+40.9} label=xcomp type=subst +slemma=expliquer +scat=v
```

$\Longrightarrow$ iterative algorithm, where

- we compute $\delta^i_{\tau,\mathcal{S}}$, depending on $\mathrm{k-good?}$ and $\mathrm{u-good?}$ (and $k-$ and $u-$) and *temperature* θ (decreases at each step)
- new edge weight for d is $\Sigma_\tau(w_\tau + \Sigma_{\mathcal{S}}\Sigma_i\delta^i_{\tau,\mathcal{S}})$
- forget $\delta^i_{\tau,\mathcal{S}}$ if last iteration i not beneficial

Evaluation scores on FTB

system	train	FTB dev	test	EasyDev rels
BKY	–	86.5	86.8	–
MALT	–	86.9	87.3	–
MST	–	87.5	88.2	–
FRMG before tuning	80.2	81.1	82.4	66.76
FRMG after tuning	86.1	84.8	85.9	68.52



Complements on tuning

Features:

- form, lemma, POS, capitalization, suffix and cluster on source and target nodes; edge label and type; distance between source and target; direction; position in sentence; subcat info on source and target; tree on source and target; . . .
- similar information on grand-parent node, when the source node is empty (unlexicalized trees)

The algorithm looks like a kind of perceptron with an oracle

Evolution:

1. improve the oracle
2. re-inject it inside FRMG's disambiguation to get an FRMG reference of the FTB
3. post-correct the reference
4. use a perceptron on the reference

Conclusion

Good to excellent coverage using meta-grammars and TAGs (+ LEFFF, SXPIPE)

- relatively easy to add new phenomena
- reasonable grammar size thanks to tree factoring
- possible future extensions with MC-TAGs (deep extractions)

Reasonable efficiency, specially using machine grids

- FRMG usable to parse very large corpora ($\sim$ 700 million words)
  French Wikipedia, AFP news, literacy wikisource, EuroParl, . . .
- efficiency could be improved using
  - probalistic DYALOG (beam-search, n-best)
  - (non-deterministic) tagging, supertagging, and hypertagging

Good accuracy, thanks to new semi-supervised ML techniques

- still a few points below best French stochastic parsers (why ?)
  (but tested and trained on the same French TreeBank)
- better stability on other corpora (to be checked)
- better extensibility, to deal with very rare phenomena

- FRMG used on large corpora for (linguistic) knowledge acquisition