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
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
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
  \[\Rightarrow\] efficiency problems during parsing
- many common sub-trees
  \[\Rightarrow\] development and maintenance problems
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
  $\leadsto$ **generation** of factorized trees from the descriptions
**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): \( T[G_+, t; G_] \equiv T[t].\sigma_+ \cup T[\epsilon].\sigma_- \)

**Tree factoring**

- does not change the expressive power or the complexity of TAGs
- but unfactoring \( \implies \) exponential increase of the grammar
- used directly by DyALog
Several possible realizations for the subject (NP, cln, S, ...)
In French, the subject may be missing, under conditions

\[ V.\text{top.mode} = \neg \text{inf|imp|part} \]

\[ V.\text{top.mode} = \text{inf|imp|part} \]
Free ordering

The verbal arguments are not ordered (rough approximation)

Unfactoring: \((1_{\text{no subj}} + 3_{\text{subj}}) \times (1_{\text{no arg}} + 2_{1 \text{ arg}} + 2_{\text{2 args}}) = 20 \text{ trees}\)
Natural description of coordination by repetition:
Definition (Meta-grammar)

Modular Description with **classes** grouping **constraints**, and with **inheritance**
Definition (Meta-grammar)

Modular Description with **classes** grouping **constraints**, and with **inheritance**

```plaintext
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(Inf).bot.inv = value(+),
    $arg.extracted = value(−),
    $arg.real = value(N2),
    desc.extraction = value(~−),
    node(V).top.mode= value(~inf|imp|...);
  ~psubj => node(Inf).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 −**
- **Guards** (=>)
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
  - $C_1[-R \cup K_1]xC_2[+R \cup K_2] = (C_1xC_2)[=R \cup K_1 \cup K_2]$
  - (Namespace) $\Rightarrow$ import producing classe with renaming
    $C_1[-N::R \cup K_1]xC_2[+R \cup K_2] = (C_1xN::C_2)[=N::R \cup K_1 \cup N::K_2]$

- Guard reduction (whenever possible)
- Constraint checking

**Step 3: TAG Trees**

Use of constraints of neutral classes to build trees

- underspecified precedence between sibling nodes $\Rightarrow$ interleaving
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, ...)  
  ➞ importance of semantic features provided by the lexicon (LEFFF)
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.
Clause complements are modifiers, hence represented by TAG auxiliary trees, with a foot node sharing a common category with the root node.

```plaintext
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;
}
```

```
$X$
  id=Root
  *$X$
    id=Foot,top=_1,bot=_1
```
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
}

\[ x \star x \]
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.

```java
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);
}
```

```
incise
  $x$
  incise
  $mod$

$mod$
```
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.

\textit{ce matin}, Paul part tôt.

Paul, \textit{ce matin}, part très tôt.

```cpp
class modifier_at_S_level {
  + s_modifier;
  - x_modifier;
  - s_adj_pos;

  node(Root).bot = node(Foot).top;
  node(Root).cat = value(S|VMod);
}

class modifier_on_S {
  + s_adj_pos;

  node(Root).cat = value(S);
}
```

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(\sim \text{très | intensive | equalizer});
  ...
}
Another frequent realization provided by the temporal nominal phrases *il a, cet après-midi, fini sa présentation.*

```java
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
}
```
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*

```plaintext
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);
}
```
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
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(+);
}
```
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
}

Class diagram:

```
* N2
   N2 bot.semtype: event
       incise
           adv
               top.adv_kind: très|intensive|equalizer
```
The various combinations (realizations × attach point × 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) \( \Rightarrow \)

<table>
<thead>
<tr>
<th>Tree Type</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>79 trees</td>
<td>( \sim 8% )</td>
</tr>
<tr>
<td>PP on Vmod</td>
<td>4.3%</td>
</tr>
<tr>
<td>PP on S (beginning of sentence)</td>
<td>0.8%</td>
</tr>
<tr>
<td>temporal NP</td>
<td>0.35%</td>
</tr>
<tr>
<td>conj. subordinate</td>
<td>0.13%</td>
</tr>
</tbody>
</table>

We observe a fast decrease of the frequencies.
### Classes
- Trees: 333+33
- Init.: 78
- Aux.: 288
- Wrap Aux.: 45
- Left Aux.: 83
- Right Aux.: 159

#### Distribution per tree kinds

<table>
<thead>
<tr>
<th>Anchor</th>
<th>v</th>
<th>coo</th>
<th>adv</th>
<th>adj</th>
<th>csu</th>
<th>prep</th>
<th>aux</th>
<th>np</th>
<th>nc</th>
<th>det</th>
<th>pro</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>229</td>
<td>42</td>
<td>44</td>
<td>21</td>
<td>9</td>
<td>10</td>
<td>2</td>
<td>3</td>
<td>25</td>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>

#### Distribution per anchor kind

<table>
<thead>
<tr>
<th>Canonical</th>
<th>Extr.</th>
<th>Active</th>
<th>Passive</th>
<th>Quest.</th>
<th>Rel.</th>
<th>Cleft</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>18</td>
<td>25</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>5</td>
</tr>
</tbody>
</table>

#### Distribution per syntactic phenomena

<table>
<thead>
<tr>
<th>Guards</th>
<th>Disjunctions</th>
<th>Interleaving</th>
<th>Kleene star</th>
</tr>
</thead>
<tbody>
<tr>
<td>3853</td>
<td>214</td>
<td>40</td>
<td>48</td>
</tr>
</tbody>
</table>

#### Use of factoring operators
Tree #198: canonical verb (simplified)

- 36 terminal classes
- 106 guards, 6 disj., 1 free order
- Hand-crafting impossible
### Grammaire FRMG

**hypertag #198**

| arg0 | extracted -  
|      | kind subj  
|      | pcas -     |
| real | real0 - CS |   N | PP | S | cln | prel | pri |

| 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 | extracted -  
|      | kind kind2 - | prepacomp | prepobj | prepscomp |
|      | 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
  - kind subj
  - pcas -
  - real real0 - CS | N2 | PP | S | cln | prel | pri
- arg0
- arg1
  - kind kind1 - | acomp | obj | prepacomp | prepobj
  - pcas pcas1 + | - | apres | à | avec | de | par | ...
  - real real1 - CS | N | N2 | PP | S | V | adj | cla | ...
- arg1
- arg2
  - kind kind2 - | prepacomp | prepobj | prepcomp |
  - pcas pcas2 - | + | apres | à | ...
  - real real2 - CS | N | N2 | PP | S | ...
- refl

**Lexique LEFFF**

**hypertag «promettre»**

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

The variables \(X\) propagate the hypertag values to nodes and guards =⇒ block non valid paths in the factored trees.
lexicon/grammar interaction: *hypertag*

The variables $X$ propagate the hypertag values to nodes and guards $\implies$ block non valid paths in the factored trees.
Degree & impact of factoring

- complete tree unfactoring $\Rightarrow \sim 2$ millions trees (FRMG 2007)

- partial unfactoring on #198 (keeping disjunctions)
  + intersection with LEFFF 195 subcat frames $\Rightarrow 5729$ trees (+ 206)

- too large for computing the left-corner relation

- test of the 2 versions on 4000 EasyDev sentences $\Rightarrow$ factoring: no overhead and better optimization

<table>
<thead>
<tr>
<th>parser</th>
<th>avg</th>
<th>median</th>
<th>90%</th>
<th>99%</th>
</tr>
</thead>
<tbody>
<tr>
<td>+fact. -lc (207 trees)</td>
<td>1.33</td>
<td>0.46</td>
<td>2.63</td>
<td>12.24</td>
</tr>
<tr>
<td>-fact. -lc (5935 trees)</td>
<td>1.43</td>
<td>0.44</td>
<td>2.89</td>
<td>14.94</td>
</tr>
<tr>
<td>+fact. +lc (207 trees)</td>
<td>0.64</td>
<td>0.16</td>
<td>1.14</td>
<td>6.22</td>
</tr>
</tbody>
</table>
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 ?
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.
```
1 Designing

2 Using

3 Improving
Try to improve along 3 axes:

- increasing the **coverage** in terms of full parses
  \[\Rightarrow\] unsupervised machine learning methods on large corpora: error mining

- improving parsing **accuracy**
  \[\Rightarrow\] 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)

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

- Also difference matrices between 2 runs
### Performance evolution on EsyDev

<table>
<thead>
<tr>
<th></th>
<th>% parses</th>
<th>Chunks</th>
<th>Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>full</td>
<td>prec.</td>
</tr>
<tr>
<td>R006 (05/2007)</td>
<td>42.16%</td>
<td>78.12%</td>
<td>71.27%</td>
</tr>
<tr>
<td>R027 (12/07)</td>
<td>56.06%</td>
<td>83.66%</td>
<td>82.90%</td>
</tr>
<tr>
<td>R076 (02/2009)</td>
<td>59.47%</td>
<td>84.23%</td>
<td>82.91%</td>
</tr>
<tr>
<td>R101 (06/09)</td>
<td>59.56%</td>
<td>83.24%</td>
<td>79.63%</td>
</tr>
<tr>
<td>R139 (09/2009)</td>
<td>64.73%</td>
<td>87.41%</td>
<td>86.00%</td>
</tr>
<tr>
<td>R157 (10/2009)</td>
<td>67.03%</td>
<td>87.71%</td>
<td>86.84%</td>
</tr>
<tr>
<td>R240 (11/2010)</td>
<td>69.01%</td>
<td>88.24%</td>
<td>89.20%</td>
</tr>
<tr>
<td>R374 (12/2011)</td>
<td>82.57%</td>
<td>89.46%</td>
<td>89.90%</td>
</tr>
<tr>
<td>05/2012</td>
<td>83.58%</td>
<td>89.15%</td>
<td>89.43%</td>
</tr>
</tbody>
</table>

### Campaign f-measure chunks f-mesure relations

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
<td>69%</td>
<td></td>
<td>41%</td>
</tr>
<tr>
<td>2007</td>
<td>89%</td>
<td></td>
<td>63%</td>
</tr>
</tbody>
</table>

**Note:** Our evaluation tools provide lower figures than the official tools.
<table>
<thead>
<tr>
<th>Corpus</th>
<th>#sentences</th>
<th>Cov.</th>
<th>t. avg. (s)</th>
<th>amb.</th>
<th>avg. 09/09</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUROTRA</td>
<td>334</td>
<td>100%</td>
<td>0.09</td>
<td>0.81</td>
<td>10/10</td>
</tr>
<tr>
<td>TSNLP</td>
<td>1661</td>
<td>95.18%</td>
<td>0.04</td>
<td>0.48</td>
<td>11/10</td>
</tr>
<tr>
<td>EasyDev</td>
<td>3879</td>
<td>69.01%</td>
<td>0.46</td>
<td>1.10</td>
<td>11/10</td>
</tr>
<tr>
<td>JRCacquis</td>
<td>1.1M</td>
<td>51.26%</td>
<td>1.41</td>
<td>1.1</td>
<td>59.46%</td>
</tr>
<tr>
<td>Europarl</td>
<td>0.8M</td>
<td>70.19%</td>
<td>1.69</td>
<td>1.36</td>
<td>78.33%</td>
</tr>
<tr>
<td>EstRep</td>
<td>1.6M</td>
<td>67.05%</td>
<td>0.69</td>
<td>0.92</td>
<td>75.06%</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>2.2M</td>
<td>69.11%</td>
<td>0.49</td>
<td>0.87</td>
<td>79.48%</td>
</tr>
<tr>
<td>Wikisource</td>
<td>1.5M</td>
<td>61.08%</td>
<td>0.71</td>
<td>0.89</td>
<td>66.79%</td>
</tr>
<tr>
<td>AFP</td>
<td>1.6M</td>
<td>52.15%</td>
<td>0.51</td>
<td>1.06</td>
<td></td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Corpus</th>
<th>#sent.</th>
<th>Cover. (%)</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftb train</td>
<td>9,881</td>
<td>91.22</td>
<td>0.70</td>
</tr>
<tr>
<td>ftb dev</td>
<td>1,235</td>
<td>90.68</td>
<td>0.59</td>
</tr>
<tr>
<td>ftb test</td>
<td>1,235</td>
<td>90.59</td>
<td>0.56</td>
</tr>
<tr>
<td>ftb all</td>
<td>12,351</td>
<td>91.11</td>
<td>0.72</td>
</tr>
</tbody>
</table>

(run inline.old43 2012-06-09)
<table>
<thead>
<tr>
<th>id</th>
<th>form</th>
<th>lemma</th>
<th>POS</th>
<th>head</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>par</td>
<td>par</td>
<td>P</td>
<td>15</td>
<td>p_obj</td>
</tr>
<tr>
<td>2</td>
<td>qui</td>
<td>qui</td>
<td>PRO</td>
<td>1</td>
<td>obj</td>
</tr>
<tr>
<td>3</td>
<td>a</td>
<td>avoir</td>
<td>V</td>
<td>5</td>
<td>aux_tps</td>
</tr>
<tr>
<td>4</td>
<td>-t-elle</td>
<td>-t-elle</td>
<td>CL</td>
<td>5</td>
<td>suj</td>
</tr>
<tr>
<td>5</td>
<td>voulu</td>
<td>vouloir</td>
<td>V</td>
<td>0</td>
<td>root</td>
</tr>
<tr>
<td>6</td>
<td>que</td>
<td>que</td>
<td>C</td>
<td>5</td>
<td>obj</td>
</tr>
<tr>
<td>7</td>
<td>ces</td>
<td>ce</td>
<td>D</td>
<td>9</td>
<td>det</td>
</tr>
<tr>
<td>8</td>
<td>deux</td>
<td>deux</td>
<td>A</td>
<td>9</td>
<td>mod</td>
</tr>
<tr>
<td>9</td>
<td>livres</td>
<td>livre</td>
<td>N</td>
<td>15</td>
<td>suj</td>
</tr>
<tr>
<td>10</td>
<td>et</td>
<td>et</td>
<td>C</td>
<td>9</td>
<td>coord</td>
</tr>
<tr>
<td>11</td>
<td>ce</td>
<td>ce</td>
<td>D</td>
<td>12</td>
<td>det</td>
</tr>
<tr>
<td>12</td>
<td>DVD</td>
<td>DVD</td>
<td>N</td>
<td>10</td>
<td>dep_coord</td>
</tr>
<tr>
<td>13</td>
<td>lui</td>
<td>lui</td>
<td>CL</td>
<td>15</td>
<td>a_obj</td>
</tr>
<tr>
<td>14</td>
<td>soient</td>
<td>être</td>
<td>V</td>
<td>15</td>
<td>aux_pass</td>
</tr>
<tr>
<td>15</td>
<td>rendus</td>
<td>rendre</td>
<td>V</td>
<td>6</td>
<td>obj</td>
</tr>
<tr>
<td>16</td>
<td>?</td>
<td>?</td>
<td>PONCT</td>
<td>5</td>
<td>ponct</td>
</tr>
</tbody>
</table>
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 $d' \in D_{i,j}$ are bad
  $\implies$ should reinforce when correctly kept (or unkept), and penalize when incorrectly kept (or unkept)

Actually 4 cases:

<table>
<thead>
<tr>
<th></th>
<th>kept</th>
<th>unkept</th>
</tr>
</thead>
<tbody>
<tr>
<td>ok</td>
<td>reinforce</td>
<td>penalize</td>
</tr>
<tr>
<td>bad</td>
<td>penalize</td>
<td>???</td>
</tr>
</tbody>
</table>
For the 4th case, we need an oracle indicating which dependency \( d' \in D_{i,j} \) 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 \( \tau \) and set of features \( S \), we determine how often it was OK to keep or discard the dependencies \( d \) satisfying \( \tau \) and \( S \).

\[
\begin{array}{cccccccccc}
\text{rule} & \# \delta & \text{delta}(s) & \mathbf{w} & \mathbf{k'} & \mathbf{k} & \mathbf{k-}\text{good?} & \mathbf{u} & \mathbf{u-}\text{u-good?} & \text{features} \\
\hline
\text{XCOMP_TOPIC} & 88 & -6.0/-27.0 & +1994.8 & 6.8 & 6.1 & (+2.8) & 62.5 & 24.6 & (+21.8) & \text{label=xcomp type=subst +slemma=dire +scat=v} \\
\text{XCOMP_TOPIC} & 85 & +46.0/+28.0 & +2200.2 & 23.5 & 2.7 & (+39.8) & 67.1 & 6.7 & (+40.9) & \text{label=xcomp type=subst +slemma=expliquer +scat=v} \\
\end{array}
\]

\[\Rightarrow\] iterative algorithm, where

- we compute \( \delta^i_{\tau,S} \), depending on \( k-\text{good?} \) and \( u-\text{good?} \) (and \( k- \) and \( u- \)) and temperature \( \theta \) (decreases at each step)
- new edge weight for \( d \) is \( \sum_{\tau}(w_{\tau} + \sum_{S}i\delta^i_{\tau,S}) \)
- forget \( \delta^i_{\tau,S} \) if last iteration \( i \) not beneficial
Evaluation scores on FTB

<table>
<thead>
<tr>
<th>system</th>
<th>train</th>
<th>FTB</th>
<th>test</th>
<th>EasyDev</th>
<th>rels</th>
</tr>
</thead>
<tbody>
<tr>
<td>BKY</td>
<td>-</td>
<td>86.5</td>
<td>86.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MALT</td>
<td>-</td>
<td>86.9</td>
<td>87.3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MST</td>
<td>-</td>
<td>87.5</td>
<td>88.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>FRMG before tuning</td>
<td>80.2</td>
<td>81.1</td>
<td>82.4</td>
<td>66.76</td>
<td></td>
</tr>
<tr>
<td>FRMG after tuning</td>
<td>86.1</td>
<td>84.8</td>
<td>85.9</td>
<td>68.52</td>
<td></td>
</tr>
</tbody>
</table>

**Graph:**
- **LAS:** 80, 82, 84, 86
- **Iteration:** 0, 1, 2, 3, 4, 5, 6
- **FRMG:** 80.2, 83.1, 85.5, 86.2, 86.3, 86.2, 86.2
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 (+ LEFF, 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 (~ 700 million words)
  French Wikipedia, AFP news, literacy wikisource, EuroParl, . . .
- efficiency could be improved using
  - probabilistic 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