Surface Realisation using Tree Adjoining Grammar. Application to Computer Aided Language Learning

Claire Gardent
CNRS / LORIA, Nancy, France
(Joint work with Eric Kow and Laura Perez-Beltrachini)

March 24, 2014

TAG-Based Surface realisation (SR) ...

- maps data to text
- using a grammar which relates NL expressions, syntactic structures and meaning representations

John loves Mary

Parsing $\Downarrow$	Grammar Lexicon Algorithm	$\Uparrow$ Generation

l1:john(j), l2:mary(m), l3:love(e,j,m)

Outline

1. Grammars: TAG, FB-LTAG and Implementation
2. Algorithms for Surface Realisation
3. Application to Computer Aided Language Learning

Tree Adjoining Grammar

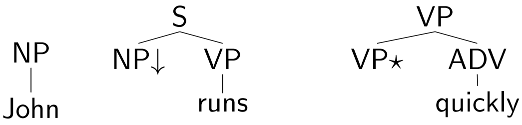
A Tree Adjoining Grammar (TAG) is a tuple $G = \langle N, T, I, A, S \rangle$ such that

- T and N are terminals and nonterminals categories,
- I is a finite set of initial trees, and
- A is a finite set of auxiliary trees,
- S is a distinguished non terminal (the axiom)

The trees in $I \cup A$ are called **elementary** trees.
The trees of G are combined using **adjunction** and **substitution**.
Each derivation yields two structures: a derived and a derivation tree.

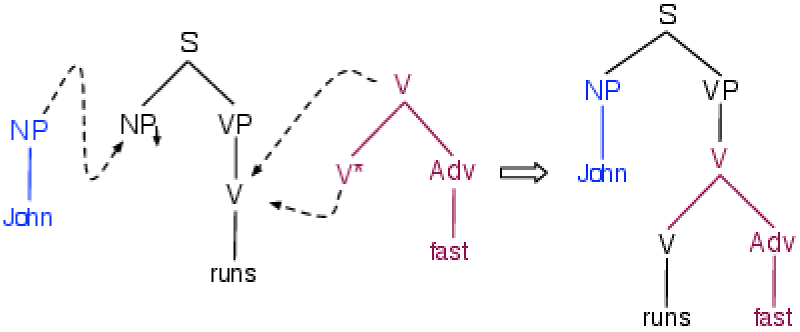
Example Elementary Trees

- **Initial trees** are elementary trees whose leaves are labelled with non terminal or terminal categories. Leaf nodes labelled with non terminal are substitution nodes marked with ↓
- **Auxiliary trees** are elementary trees with a designated *foot node*. The root and the foot nodes are labelled with the same category.



Example TAG Derivation

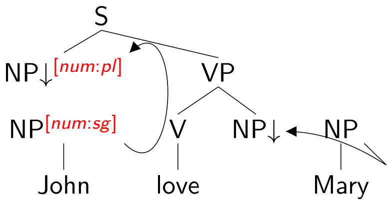
- **Substitution** inserts a derived or elementary tree at the substitution node of a TAG tree.
- **Adjunction** inserts an auxiliary tree into a tree (Adjunction is not allowed on substitution nodes)



Feature-Based TAG

- tree nodes are decored with two feature structures called **top** and **bottom**
- unifications on these feature structures are performed
  - during derivation, each time a substitution or an adjunction takes place
  - after derivation: at the end of the derivation, the top and bot FS of each node are unified

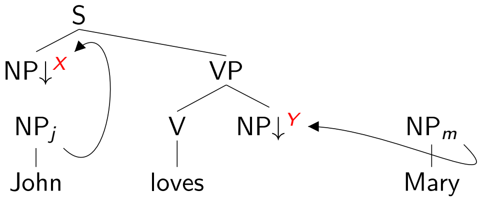
Using Features to capture Subject/Verb Agreement



Unification based Semantic construction in FTAG

- Semantic representation language : unification-based flat underspecified formulae (aka MRS)
- Each elementary tree is associated with a formula ϕ representing its meaning
  Missing semantic arguments are represented by unification variables
- Elementary tree nodes are decorated with semantic indices occurring in ϕ
- The meaning of a derived tree is the union of the meanings associated with the elementary trees modulo the unifications made during processing

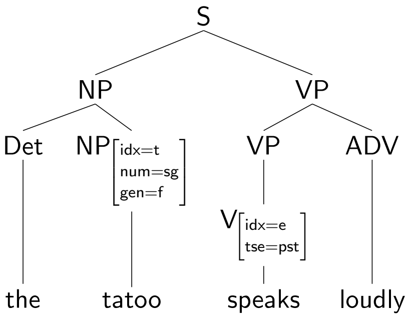
Capturing the Interplay between Syntax and Semantics



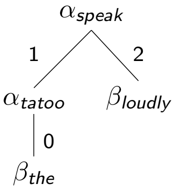
l_j:name(j,john) l_l:love(X,Y) l_m:name(m,mary)

l_l:love(j,m),l_j:name(j,john),l_m:name(m,mary)

Derived and Derivation Trees



(a) Derived tree



(b) Derivation tree

Implementing a TAG

A large coverage TAG consists of several thousands of trees

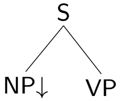
For each word type, there are as many trees as there are different possible syntactic contexts for that word

But these trees often share subtrees

To implement a FB-LTAG, we use the XMG specification language and compiler

As a result, each TAG tree is associated with the set of XMG classes used to produce it.

Structure Sharing in TAG

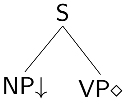


(Canonical Subject)

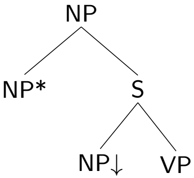
$\wedge$

(Active Verb)

$\Rightarrow$



Canonical Subject,
Active Verb
(e.g. the boy sleeps)

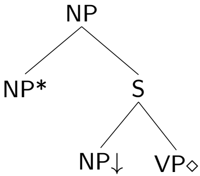


(Relative Subject)

$\wedge$

(Active Verb)

$\Rightarrow$



Relative Subject,
Active Verb
(e.g. the boy who sleeps)

SemFRAG, a grammar for French

- An FB-LTAG for French with unification-based compositional semantics
- Roughly 6 000 elementary trees
- Coverage :
  - 35 basic verbal subcategorisation frames
  - Argument redistributions: active, passive, middle, neuter, reflexivisation, impersonal, passive impersonal
  - Argument realisations: cliticisation, extraction, omission, permutations, etc.)
  - Predicative (adjectival, nominal and prepositional) and light verb constructions
  - Basic descriptions for adverbs, determiners and prepositions.

XXTAG, a grammar for English

- An FB-LTAG for English
- Roughly 1200 elementary trees
- Reimplementation of the XTAG grammar developed at UPenn
- Coverage : most of the Penn Treebank

Complexity

Surface realisation from flat semantics is exponential in the length of the input (Brew92,Kay96)

- **Unordered Input**: At least 2^n possible combinations with n the number of literals in the input
- **intersective modifiers**: 2^{m+1} possible intermediate structures with m the number of modifiers for a given structure
- **lexical ambiguity**: $\prod_{i=1}^{i=n} Lex_i$ possible combinations with Lex_i the number of lexical entries associated with literal l_i and n the number of literals in the input

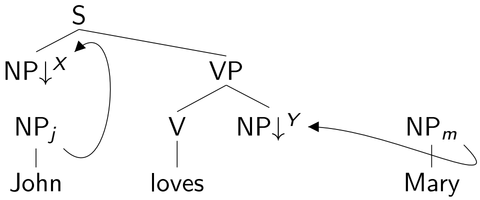
SR Algorithm 1: GenI

3 steps:

1. **Lexical selection**: select trees whose semantics subsumes the input semantics.
2. **Combination**: perform substitutions or adjunctions between selected trees.
3. **Extraction**: Return the trees which are syntactically complete and whose semantics matches the input semantics.

Claire Gardent and Eric Kow.
A Symbolic Approach to Near-Deterministic Surface Realisation useing Tree Adjoining Grammar.
ACL 2007, Prague

Example



l_j*:name(j,john)* l_l*:love(X,Y)* l_m*:name(m,mary)*

l_l*:love(j,m),*l_j*:name(j,john),*l_m*:name(m,mary)*

Optimisations

- Semantic indices used to limit the impact of unordered input
- Substitutions before Adjunctions to deal with intersective modifiers
- Polarity based filtering to reduce the initial search space and limit the impact of lexical ambiguity

Intersective modifiers

`fierce(x),little(x),cat(x),black(x)`

15 intermediate structures:

F,L,B,FL,FB,BL,BF,LB,LF,FLB,FBL,BLF,BFL,LBF,LFB

multiplied by the context :

x 2: the *F,L,B,FL,FB,BL,BF,LB,LF,FLB,FBL,BLF,BFL,LBF,LFB*

x 2: the *F,L,B,FL,FB,BL,BF,LB,LF,FLB,FBL,BLF,BFL,LBF,LFB* runs

45 structures built

Substitutions before Adjunctions

Adjunction restricted to syntactically complete trees
The 2^{m+1} intermediate structures are not multiplied out by the context :

the cat runs

the fierce cat runs, the black cat runs, the little cat runs, the fierce little cat runs, the fierce black cat runs, the black fierce cat runs, ...

16 structures built

Polarity based filtering (Perrier 2003)

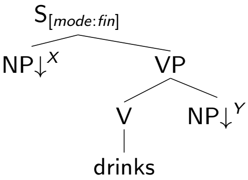
Polarity based filtering filters out all combinations of lexical items which cannot result in a grammatical structure

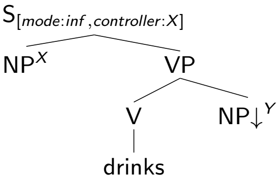
- The grammar trees are associated with polarities reflecting their syntactic resources and requirements
- A combination of trees covering the input semantics but whose polarity is not zero is necessarily syntactically invalid and is therefore filtered out.
- A finite state automata is built which represent the possible choices (transitions) and the cumulative polarity (states)
- The paths leading to a state with polarity other than zero are deleted (automata minimisation)

Polarity Filtering

Many combinations are syntactically incompatible. Polarity filtering aims to detect these combinations and to filter them out.

`john(j)`	`drink(e,j,w)`	`water(w)`	Polarity Count
$+1$np	S_{FIN}-2np	$+1$np	$+0$np
	S_{INF}-1np		$+1$np



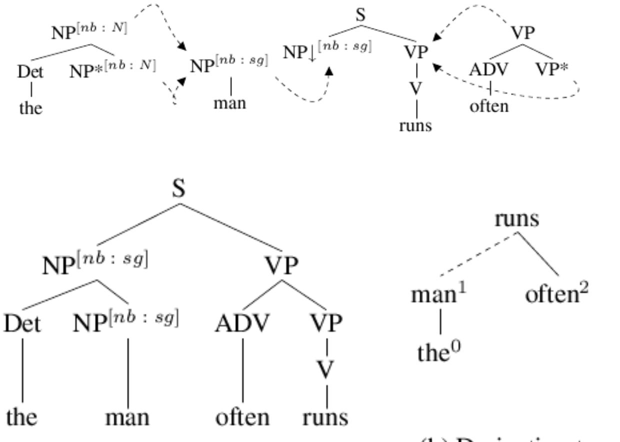


SR Algorithm 2: RTGen

- Builds derivation rather than derived trees ...
- using a conversion from FB-LTAG to Feature Based Regular Tree Grammar (FB-RTG, Schmitz and Leroux 2009)
- Earley algorithm with packing and sharing

Claire Gardent, Benjamin Gottesman, and Laura Perez-Beltrachini. Using Regular Tree Grammars to enhance Sentence Realisation. *Natural Language Engineeering*, 2011.

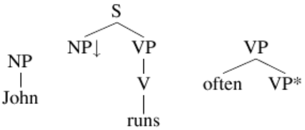
Derived and Derivation Tree



(a) Derived tree

(b) Derivation tree

Converting a TAG to an RTG



r1.	NP_S	$\rightarrow$	$john(NP_A)$
r2.	S_S	$\rightarrow$	$runs(S_A\, NP_S\, VP_A\, V_A)$
r3.	VP_A	$\rightarrow$	$often(VP_A)$
r4.	NP_A	$\rightarrow$	ϵ
r5.	S_A	$\rightarrow$	ϵ
r6.	V_A	$\rightarrow$	ϵ
r7.	VP_A	$\rightarrow$	ϵ

Earley Algorithm

Axiom $\overline{[S' \rightarrow \bullet S_S, \emptyset]}$

Goal $[S' \rightarrow S_S \bullet, \phi]$ where ϕ is the input semantics.

Prediction $\dfrac{[A \rightarrow a(\alpha \bullet B_x\ \beta), \varphi]}{[\sigma(B^0 \rightarrow b(\bullet B^1, ..., B^n), \psi)]}$

with $\langle B \rightarrow b(B^1, ..., B^n), \psi \rangle$ a grammar rule

$\sigma = mgu(B, B^0)$, $P[x] \in \psi$ and $\varphi \cap \psi = \emptyset$

Completion $\dfrac{[A \rightarrow a(\alpha \bullet B\ \delta), \varphi][B \rightarrow b(\beta) \bullet, \psi]}{[\sigma(A \rightarrow a(\alpha(B, f) \bullet \delta), \varphi \cup \psi)]}$

with $\sigma = mgu(B, B^0)$, $\varphi \cap \psi = \emptyset$

RTGen vs GenI

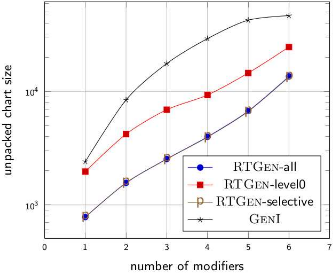
- All trees are taken into account while filtering (GenI's polarity filtering ignores auxiliary trees)
- All features can be used (GenI's polarity filtering can only use ground features i.e., categories)
- All syntactic constraints are applied (not just counting)
- Intersective Modifiers are handled using packing and sharing

Comparison on two Automatically Generated Benchmarks

- Modifiers benchmark: modification differently distributed over the predicate argument structures + lexical ambiguity in modifiers. 1 789 input formulae.
- All benchmark: modification, varying number and type of verb arguments. 890 input formulae.

Claire Gardent, Benjamin Gottesman, and Laura Perez-Beltrachini.
Comparing the performance of two TAG-based surface realisers using controlled grammar traversal.
COLING 2010: Posters, Beijing, China.

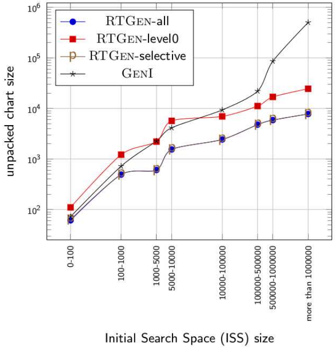
Modification



Space performance results on the MODIFIERS-benchmark.

- when using no features (RTGEN-level0) over-generation increases the number of intermediate structures.
- when using all the features (RTGEN-all) or only a selected set of them (RTGEN-selective) (almost) the same number of intermediate structures are produced.

Overall efficiency



Space

performance results on the

- For more complex cases, RTGen's sharing and packing mechanisms perform better.

Generating Grammar Exercises

Generate sentences

Use the detailed linguistic information output by the generator to select and build exercises

Three types of exercises: FIB, Shuffle and Reformulation

C. Gardent and L. Perez-Beltrachini.
Using FB-LTAG Derivation Trees to Generate Transformation-Based Grammar Exercices.
TAG+11: The 11th International Workshop on Tree Adjoining Grammars and Related Formalisms, Paris, France, September 2012.

L. Perez-Beltrachini, C. Gardent and G. Kruszewski
Generating Grammar Exercices.
The 7th Workshop on Innovative Use of NLP for Building Educational Applications, NAACL-HLT Worskhop 2012, Montreal, Canada, June.

Grammar Exercises

Built from a single sentence.

[FIB] Complete with an appropriate personal pronoun.

(S) *Elle adore les petits tatous*
(She loves the small armadillos)
(Q) ______ adore les petits tatous (gender=fem)
(K) elle

[Shuffle] Use the words below to make up a sentence.

(S) *Tammy adore les petits tatous*
(Tammy loves the small armadillos)
(Q) tatous / les / Tammy / petits / adore
(K) Tammy adore les petits tatous.

Grammar Exercises

Built from a pair of syntactically related sentences

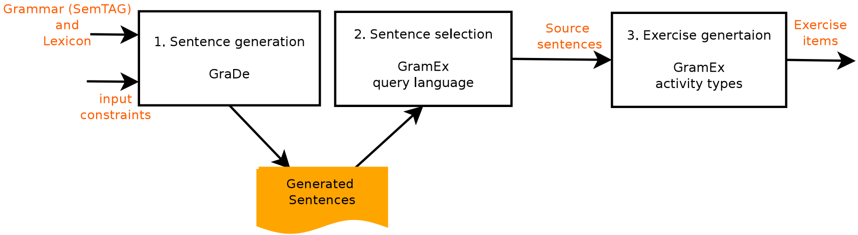
[Reformulation] Rewrite the sentence using passive voice

(Q) C'est Tex qui a fait la tarte.
(It is Tex who has baked the pie.)

(K) C'est par Tex que la tarte a été faite.
(It is Tex by whom the pie has been baked.)

Active/Passive, NP/Pronoun, Assertion/Wh-Question, Assertion/YN-Question

The *GramEx* framework: generating and selecting sentences to build exercises



Creating a grammar exercise

aime(e,be,bi),bijou(bi),les(bi),betty(be)

$\Downarrow$

Creating a grammar exercise

aime(e,be,bi),bijou(bi),les(bi),betty(be)

$\Downarrow$

Bette aime le bijou.

C'est Bette qui aime les bijoux.

Bette aime les bijoux.

Creating a grammar exercise

Goal: Plural form of irregular nouns.
Exercise type: Fill-in-the-blank.

aime(e,be,bi),bijou(bi),les(bi),betty(be)

$\Downarrow$

Bette aime le bijou.

C'est Bette qui aime les bijoux.

Bette aime les bijoux.

Creating a grammar exercise

aime(e,be,bi),bijou(bi),les(bi),betty(be)

Bette aime le bijou.

C'est Bette qui aime les bijoux.

Bette aime les bijoux.

Goal: Plural form of irregular nouns.
Exercise type: Fill-in-the-blank.

$\Downarrow$

1. Select sentences
   $\mathrm{NP}[\mathit{num} = \mathit{pl} \wedge \mathit{plural} = \mathit{irreg}]$
   $\wedge$ *CanonicalOrder*

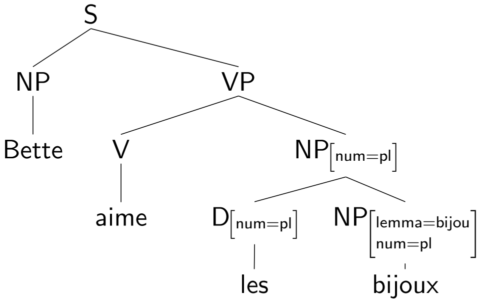
Creating a grammar exercise

aime(e,be,bi),bijou(bi),les(bi),betty(be)

Bette aime le bijou.

C'est Bette qui aime les bijoux.

Bette aime les bijoux.



$\{$*CanonicalObject*, *CanonicalSubject*, *ActiveVerb*$\}$

Goal: Plural form of irregular nouns.
Exercise type: Fill-in-the-blank.

$\Downarrow$

1. Select sentences
   $\mathrm{NP}[num = pl \wedge plural = irreg]$
   $\wedge$ *CanonicalOrder*

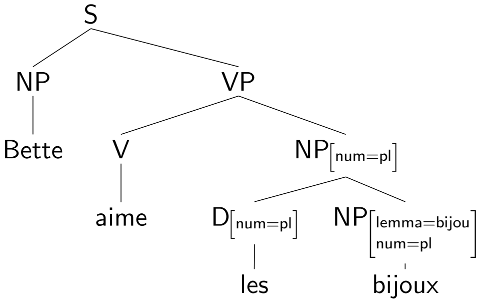
Creating a grammar exercise

aime(e,be,bi),bijou(bi),les(bi),betty(be)

Bette aime le bijou.

C'est Bette qui aime les bijoux.

Bette aime les bijoux.



$\{$*CanonicalObject*, *CanonicalSubject*, *ActiveVerb*$\}$

Goal: Plural form of irregular nouns.
Exercise type: Fill-in-the-blank.

$\Downarrow$

1. Select sentences
   $NP[num = pl \wedge plural = irreg]$
   $\wedge$ *CanonicalOrder*

2. Process the selected sentence
   $NP[num = pl] \Rightarrow$ blank
   $NP[lemma = bijou] \Rightarrow$ hint

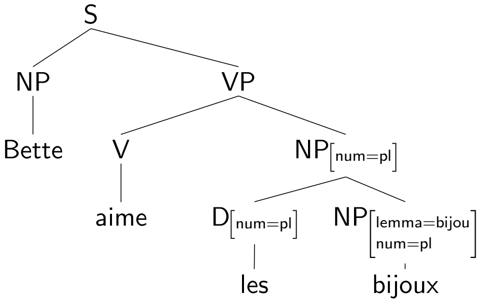
Creating a grammar exercise

aime(e,be,bi),bijou(bi),les(bi),betty(be)

Bette aime le bijou.

C'est Bette qui aime les bijoux.

Bette aime les bijoux.



$\{$*CanonicalObject*, *CanonicalSubject*, *ActiveVerb*$\}$

Goal: Plural form of irregular nouns.

Exercise type: Fill-in-the-blank.

$\Downarrow$

1. Select sentences
   $\text{NP}[num = pl \wedge plural = irreg]$
   $\wedge$ *CanonicalOrder*

2. Process the selected sentence
   $\text{NP}[num = pl] \Rightarrow$ blank
   $\text{NP}[lemma = bijou] \Rightarrow$ hint

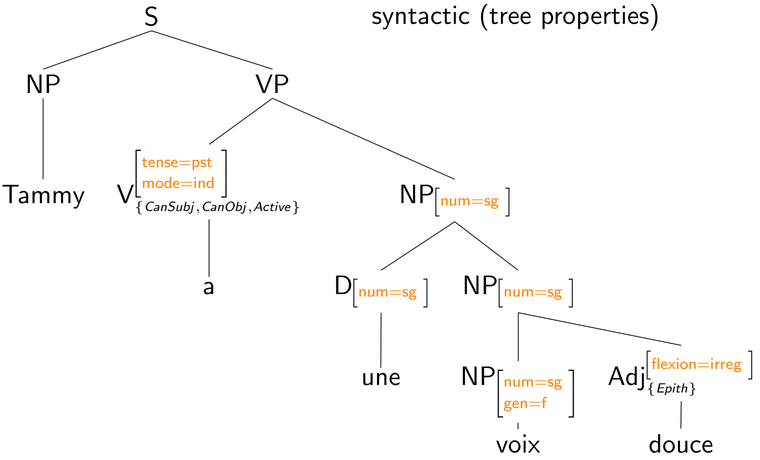
   (Q) Bette aime les ______. (bijou)

   (K) bijoux

Selecting appropriate sentences

GramEx's boolean constraint language over linguistic constants

morpho-syntactic (feature structures)
syntactic (tree properties)



Selecting appropriate sentences

GramEx's boolean constraint language: syntax and use

Boolean constraint language

- conjunction, disjunction and negation of morpho-syntactic and syntactic properties

Describes the linguistic requirements imposed by pedagogical goals

- Permits retrieving appropriate sentences from the DB

Selecting appropriate sentences

Some examples

Pedagogical goal: *Pre/post nominal irregular adjectives*

[Epith $\wedge$ flexion: irreg]

✓ *Tammy a une voix douce* (Tammy has a soft voice)

X *Tammy a une jolie voix* (Tammy has a nice voice)

Pedagogical goal: *Prepositions with infinitives*

POBJinf $\wedge$ CLAUSE

POBJinf $\Leftrightarrow$ (DE-OBJinf $\vee$ A-OBJinf)

CLAUSE $\Leftrightarrow$ Vfin $\wedge$ $\neg$Mod $\wedge$ $\neg$CCoord $\wedge$ $\neg$Sub

✓ *Tammy refuse de chanter* (Tammy refuses to sing)

X *Jean dit que Tammy refuse de chanter* (John says that Tammy refuses to sing)

Transformation-based grammar exercices

Finding syntactically related sentences (e.g. active/passive)

(Q) C'est Tex qui a <u>fait</u> la tarte.

(It is Tex who baked the pie.)

✗ *(K) Tex a fait la tarte.*

(Tex baked the pie.)

✗ *(K) La tarte a été faite par Tex.*

(The pie was baked by Tex.)

✗ *(K) C'est par Tex que la tarte sera faite.*

(It is Tex who will bake the pie.)

✗ *(K) Est-ce que la tarte a été faite par Tex ?*

(Has the pie been baked by Tex ?)

✓ *(K) C'est par Tex que la tarte a été faite.*

(It is Tex by whom the pie was baked.)

Creating transformation-based grammar exercises

To identify pairs of sentences that are identical up to a single syntactic transformation, we ...

- Use the information contained in derivation trees
- Define tree filters on pairs of derivation trees
- Retrieve sentences pairs that match those tree filters

Why Derivation Trees?

α-**faire:**{Active,CleftSubj,CanObj}
(num:sg,tse:pst,mode:ind,pers:3)

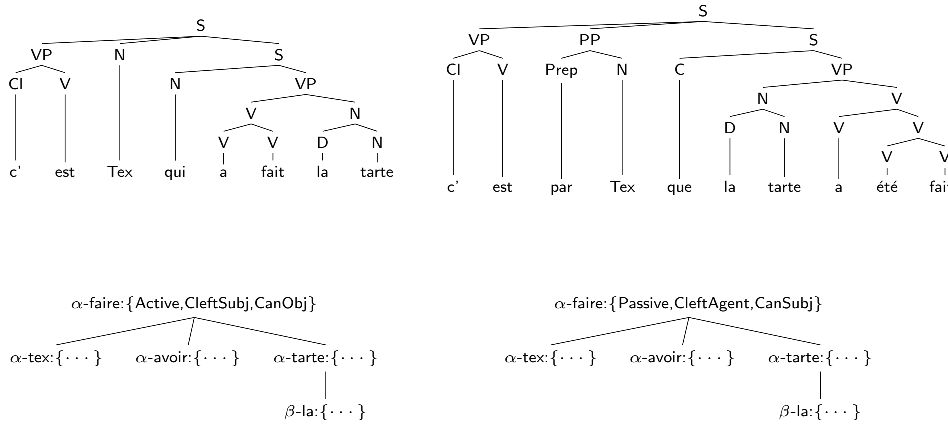
α-**tex:**{ProperNoun}
(fun:subj,gen:fem,num:sg,pers:3)

α-**tarte:**{Noun}
(fun:obj,gen:fem,num:sg)

β-**la:**{DefDet}
(gen:fem,num:sg)

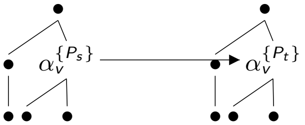
- Detailed syntactic information
- More compact than derived trees. Allow fewer and simpler filters.

Derivation vs Derived trees



Derivation Tree Filters

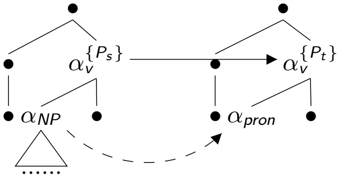
Tree filter types



e.g. `active/passive`

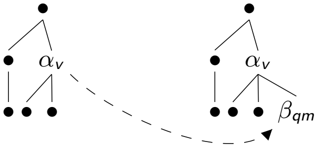
$\bullet_s$\{Active,CleftSubj,CanObj\}

$\leftrightarrow \bullet_t$\{Passive,CleftAgent,CanSubj\}



e.g. `NP/Pronoun`

$\bullet_s$\{CanSubj\} $\leftrightarrow \bullet_t$\{CliticSubj\}



e.g. `Assertion/YN-Question`

$\varnothing \leftrightarrow \bullet_q$\{questionMark\}

Meaning Preserving Transformations

Same core meaning (e.g. active/passive)

(Q) *C'est Tex qui a fait la tarte.* $\leftrightarrow$ (K) *C'est par Tex que la tarte a été faite.*
(It is Tex who has baked the pie) (It is by Tex that the pie has been baked)

$\not\leftrightarrow$ (K) *La tarte a été faite par Tex.*
(The pie has been baked by Tex)

- α-faire:{Active,CleftSubj,CanObj}
  - α-tex:$\{\cdots\}$
  - β-avoir:$\{\cdots\}$
  - α-tarte:$\{\cdots\}$
    - β-la:$\{\cdots\}$

$\leftrightarrow$

- α-faire:{Passive,CleftAgent,CanSubj}
  - α-tex:$\{\cdots\}$
  - β-avoir:$\{\cdots\}$
  - α-tarte:$\{\cdots\}$
    - β-la:$\{\cdots\}$

$\not\leftrightarrow$

- α-faire:{Passive,Agent,CanSubj}
  - α-tex:$\{\cdots\}$
  - β-avoir:$\{\cdots\}$
  - α-tarte:$\{\cdots\}$
    - β-la:$\{\cdots\}$

$\bullet_s$ {Active,CleftSubj,CanObj}
$\leftrightarrow \bullet_t$ {Passive,CleftAgent,CanSubj}

Meaning Altering Transformations

Related core meaning: content deleted, added or replaced
(e.g. Assertion/Wh-Question)

α-dort:{ CanSubj }
|
α-tatou:{ ... }
|
β-chante:{ ... }
|
β-petit:{ ... }
|
β-le:{defDet}

Le petit tatou qui chantera dort.
The small armadillo that will sing sleeps

α-dort:{ whSubj }
|
α-tatou:{ ... }
|
β-petit:{ ... }
|
β-quel:{ WhDet }

Quel petit tatou dort?
Which small armadillo sleeps?

α-dort:{ whSubj }
|
α-tatou:{ ... }
|
β-quel:{ WhDet }

Quel tatou dort?
Which armadillo sleeps?

α-dort:{ whSubj }
|
β-qui:{ WhPron }

Qui dort?
Who sleeps?

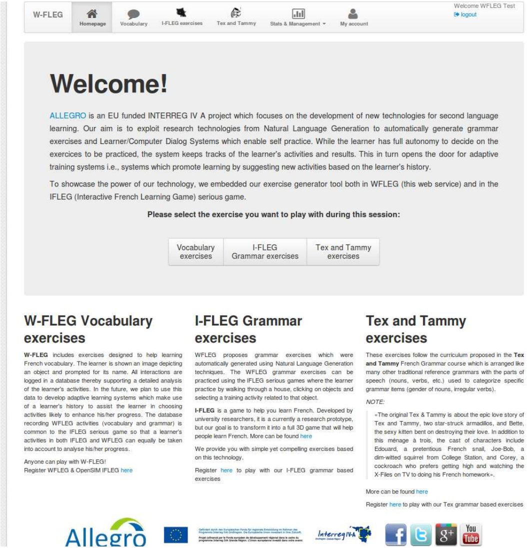
Correctness, Productivity, Integration

Manual annotation of a sample of generated exercises

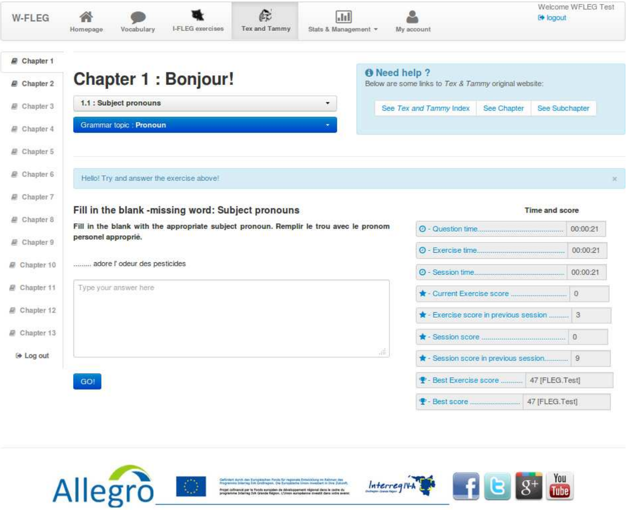
- using SemFraG and lexicon tailored to *Tex's French Grammar* vocabulary
- around 80% of the automatically generated exercises are correct
- 52 input formulae $\Rightarrow$ around 5000 exercises

Exercises generated by *GramEx* are integrated in I-FLEG (serious game) and WFLEG (web interface)

WFLEG


W-FLEG
Homepage
Vocabulary
I-FLEG exercises
Tex and Tammy
Stats & Management
My account
Welcome WFLEG Test
logout
Welcome!
ALLEGRO is an EU funded INTERREG IV A project which focuses on the development of new technologies for second language learning. Our aim is to exploit research technologies from Natural Language Generation to automatically generate grammar exercises and Learner/Computer Dialog Systems which enable self practice. While the learner has full autonomy to decide on the exercises to be practiced, the system keeps tracks of the learner's activities and results. This in turn opens the door for adaptive training systems i.e., systems which promote learning by suggesting new activities based on the learner's history.
To showcase the power of our technology, we embedded our exercise generator tool both in WFLEG (this web service) and in the IFLEG (Interactive French Learning Game) serious game.
Please select the exercise you want to play with during this session:
Vocabulary exercises
I-FLEG Grammar exercises
Tex and Tammy exercises
W-FLEG Vocabulary exercises
W-FLEG includes exercises designed to help learning French vocabulary. The learner is shown an image depicting an object and prompted for its name. All interactions are logged in a database thereby supporting a detailed analysis of the learner's activities. In the future, we plan to use this data to develop adaptive learning systems which make use of a learner's history to assist the learner in choosing activities likely to enhance his/her progress. The database recording WFLEG activities (vocabulary and grammar) is common to the IFLEG serious game so that a learner's activities in both IFLEG and WFLEG can equally be taken into account to analyse his/her progress.
Anyone can play with W-FLEG!
Register WFLEG & OpenSIM IFLEG here
I-FLEG Grammar exercises
WFLEG proposes grammar exercises which were automatically generated using Natural Language Generation techniques. The WFLEG grammar exercises can be practiced using the IFLEG serious games where the learner practice by walking through a house, clicking on objects and selecting a training activity related to that object.
I-FLEG is a game to help you learn French. Developed by university researchers, it is a currently a research prototype, but our goal is to transform it into a full 3D game that will help people learn French. More can be found here
We provide you with simple yet compelling exercises based on this technology.
Register here to play with our I-FLEG grammar based exercises
Tex and Tammy exercises
These exercises follow the curriculum proposed in the Tex and Tammy French Grammar course which is arranged like many other traditional reference grammars with the parts of speech (nouns, verbs, etc.) used to categorize specific grammar items (gender of nouns, irregular verbs).
NOTE:
«The original Tex & Tammy is about the epic love story of Tex and Tammy, two star-struck armadillos, and Bette, the sexy kitten bent on destroying their love. In addition to this ménage à trois, the cast of characters include Edouard, a pretentious French snail, Joe-Bob, a dim-witted squirrel from College Station, and Corey, a cockroach who prefers getting high and watching the X-Files on TV to doing his French homework».
More can be found here
Register here to play with our Tex grammar based exercises
Allegro
You Tube

WFLEG


W-FLEG
Homepage
Vocabulary
I-FLEG exercises
Tex and Tammy
Stats & Management
My account
Welcome WFLEG Test
logout
Chapter 1
Chapter 2
Chapter 3
Chapter 4
Chapter 5
Chapter 6
Chapter 7
Chapter 8
Chapter 9
Chapter 10
Chapter 11
Chapter 12
Chapter 13
Log out
Chapter 1 : Bonjour!
1.1 : Subject pronouns
Grammar topic : Pronoun
Need help ?
Below are some links to Tex & Tammy original website:
See Tex and Tammy Index
See Chapter
See Subchapter
Hello! Try and answer the exercise above!
Fill in the blank -missing word: Subject pronouns
Fill in the blank with the appropriate subject pronoun. Remplir le trou avec le pronom personel approprié.
.... adore l' odeur des pesticides
Type your answer here
GO!
Time and score
- Question time 00:00:21
- Exercise time 00:00:21
- Session time 00:00:21
- Current Exercise score 0
- Exercise score in previous session 3
- Session score 0
- Session score in previous session 9
- Best Exercise score 47 [FLEG.Test]
- Best score 47 [FLEG.Test]
Allegro
Interreg IV-A

WFLEG


W-FLEG
Homepage
Vocabulary
I-FLEG exercises
Tex and Tammy
Stats & Management
My account
Welcome WFLEG Test
logout
Consult Scores
Update Profile
Tex and Tammy grammar exercises
Amount of tests done 616
Average score 43.67 %
Average time 00:01:39
Tests results
Progress ratio
Progress ratio in %
Adjective
Pronoun
Verb
Noun
Determiner
Tests results
Total score with Amount of tests done
Wrong answers
Correct answers
Adjective morphology
Subject pronouns
verb être conjugation: present
Singular noun gender : masculine and feminine
Determiner: definite articles
Determiner: indefinite articles
verb avoir conjugation: present
-er verbs in present
Possessive determiner
Type of exercises done
Last Tests results
Amount of tests done
Score of the last tests done
Response time of the last tests done
Date of the tests
Score
Time
I-FLEG Grammar exercises

Thanks!