Syntax-Semantic Interface and Tree Adjoining Grammar

Djamé Seddah

NCLT, DUBLIN

01/18/06

Objective: to build a predicate argument structure from a TAG analysis

From a Tree Adjunct Grammar (TAG) analysis, we want :

- to build a deep syntactic structure with all argumental relations represented
- to have all the analysis in the same structure

We claim it is possible if we work directly in a structure which combines **derived tree** and **derivation tree** : **A shared forest**

What is an interface between syntax and semantic ?

A way to construct a semantic meaning of a given sentence.
So, what do you mean by semantic meaning ?

- a logical formula ?
- a dependency graph ?
- a predicate argument structure ?

Even, if they provide different level of informations, they rely on the same principle :The Freege theorem.

The meaning of an expression is a function of the meanings of its parts

Implication of The Freege Theorem 1/2

- If we associate minimal sense to each part of a sentence
- If we provide an interpretation function f

$\Rightarrow$ *We can obtain a meaning*

let's do that for "(1) Tarzan loves Jane" and let f assigns the first argument of 'LOVE to the longest noun :

Tarzan	:	'TARZAN
loves	:	[1] 'LOVE [2]
Jane	:	'JANE

Semantic meaning of (1) : 'TARZAN 'LOVE 'JANE

But "(2) Jane loves Tarzan" has the same meaning ! We must rely on a better interpretation function and for that we may use the order induced by syntax to assign argument positions to words.

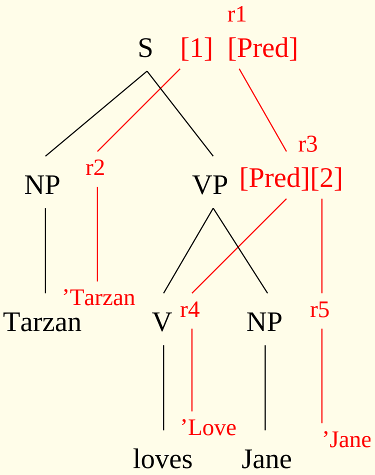
Implication of The Freege Theorem 2/2

- Providing this mini model :

r1	S $\longrightarrow$ NP VP	([1]=$f(NP)$) ([PRED]=$f(VP)$)
r2	NP $\longrightarrow$ Tarzan	'TARZAN
r3	VP $\longrightarrow$ V NP	' ([PRED]=$f(V)$) ([2]=$f(NP)$)
r4	V $\longrightarrow$ love	'LOVE
r5	NP $\longrightarrow$ Jane	'JANE

$\Rightarrow$ *We have to apply the rules and therefore to follow the derivations to get the proper result*

Applying this model



- *After the applications of the derivation rules, we obtain :*
- **'TARZAN 'LOVE 'JANE**

⇒ *Once again, we have the correct interpretation*

⇒ *But what if we want to analyze "Jane is loved by Tarzan" ?*

Applying this model

Syntactically speaking we have 2 options :

- adding the following rules :

r3	VP -> V' PP	([PRED]=$f(V)$) ([1]=$f(PP)$)
r6	V' -> be loved	'LOVE
r7	PP -> Prep NP	[PRED]=$f(NP)$
r8	..	..

$\Rightarrow$ *We have to modify deeply the corresponding semantic rules (r1 for the inversion of the arguments, etc..)*

- trying to use the fact that we are still trying to express relation between words even if this is hidden by the mechanism behind the rules

$\Rightarrow$ *So we should try to lexicalize this grammar a little bit...*

Lexicalization and semantic 1

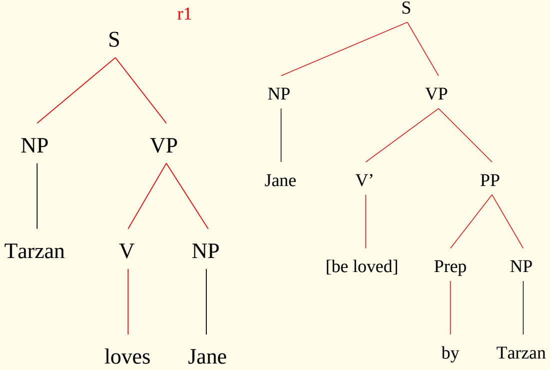
if we replace the main VP rules by :

r3	VP -> love NP	'Love ([1]=$f(NP)$)
r3'	VP -> be loved GP	'Love ([2]=$f(PP)$)

The model is a lot more readable and simplified but the problem of the inversion argument in rule r1 is still here

Lexicalization and semantic 2

Let's consider these 2 trees :



Active Parse Tree   Passive Parse Tree

- Assume that the red part is a single unit, called α_{active} (resp. $\alpha_{passive}$) and represents by itself a set of derivation rules

Lexicalization and semantic 3

We could then express the derivation trees differentlty

Observations :

- They are very similar
- *w.r.t to the order of the arcs*

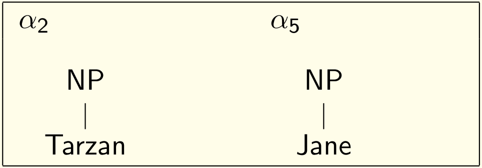
$\Rightarrow$ *one solution : numbered the nodes according to argument positions*

$\Rightarrow$ *Implicit : one argument position is linked to a derivation operation on a leaf node of* α_X

$\Rightarrow$ *Hypothesis : would it be simpler to deal with trees instead of rules -> it would simplify the semantic model*

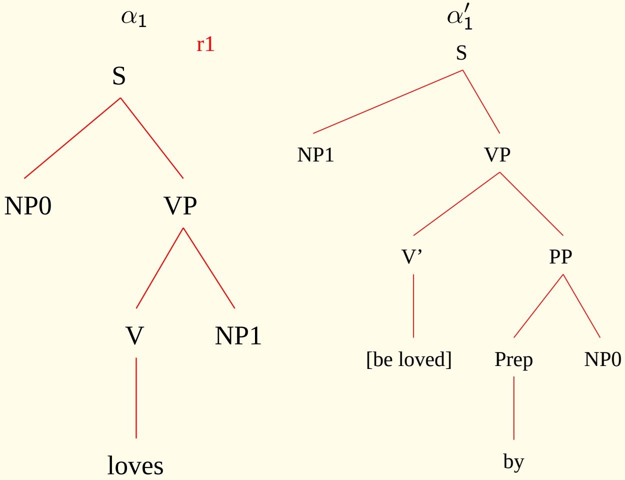
Dealing with trees

let's call α_2 and α_5 the tree corresponding to r2 (NP $\rightarrow$ Tarzan) and r5 (NP $\rightarrow$ Jane) :



Dealing with trees 2

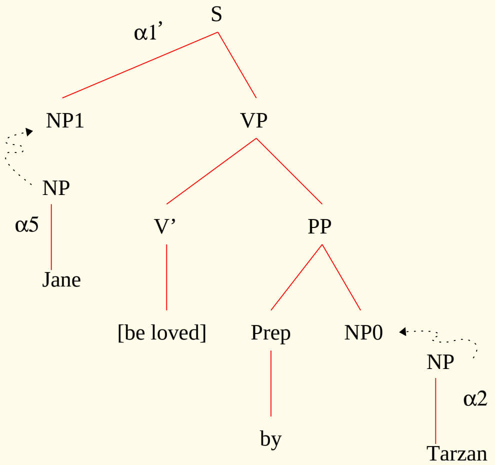
let's call α_1 the tree corresponding to α_{active} (resp. α'_1 and $\alpha_{passive}$) :



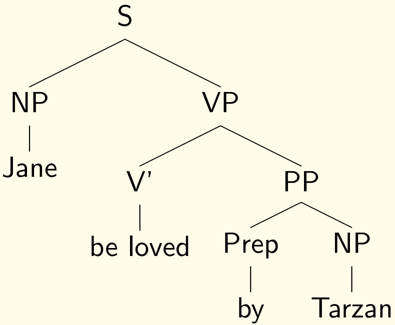
- Notice the number on the leaf nodes

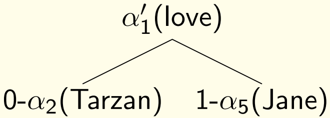
Dealing with trees 3

analysis for “Jane is loved by Tarzan”:



Result : Derived Tree (Parse Tree) and Derivation Tree (History of what have been derived).





Pause : Where is the semantic model ?

- the derivation tree here is the semantic model
- take the head as a predicate
- take its leaves as its arguments

⇒ *a predicate-argument structure, or a first order term*

⇒ *We do not need anymore the manually crafted semantic rules*

How is it possible ?

- Lexicalization :

⇒ *Each unit of the grammar is anchored by a lexical unit*

- Minimal Semantic Principle

⇒ *Each tree must correspond to a minimal semantic unit (msu)*

- predicate-argument cooccurence principle

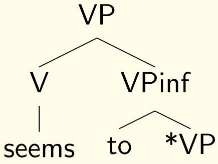
⇒ *Each argumental leaves nodes of a tree has to be fully realized*

this the so famous Well formedness principles

Where is the adjunction ?

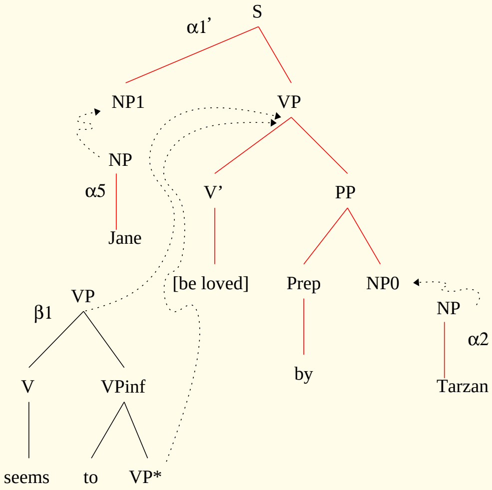
So far, we described only Lexicalized Substitution Tree Grammars. In order to fully lexicalized CFG, we need an optional operation of tree insertion : The adjunction.

- only a certain type of tree can be adjoined :
  The auxiliary tree (always prefixed by β)
- they must have a leaf node, the foot node with the same label than the root of the tree, the path from the root to the foot is called the spine
- let's β_1 the auxiliary tree for the raising verb "to seem" :



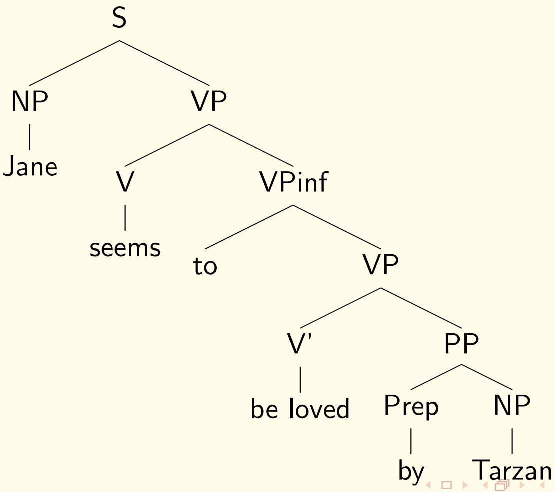
Example of adjunction 1/3

Analysis for “Jane seems to be loved by Tarzan” :



Example of adjunction 2 /3

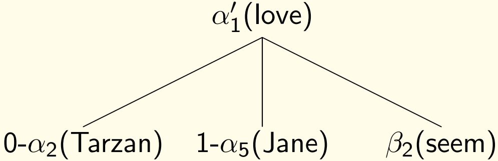
Result : Derived Tree



]

Example of adjunction 3/3

Results (suite) : Derivation tree for “Tarzan seems to love Jane” :



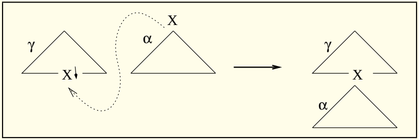
- the link between β_2(seem) and α'_1(love) do not reflect a dependency relation but a modifier one (it’s of course disputable)
- depending of the type of anchors (predicative or modifier), the adjunction link can be in the other direction.... Here are come the problems we will discuss the next time.

A lexicalized Formalism

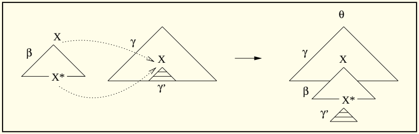
- Grammar contains elementary trees (initial and auxiliary trees)
- each tree is anchored by a lexical unit
- Two operations : substitution and adjonction
- As opposed to CFG, derivation tree and derived tree are not isomorphic anymore
- As opposed to LFG and HPSG, parsable in polynomial time

Operations on trees

- **The substitution** *is a context free derivation of an initial tree to a leaf node of any elementary tree*

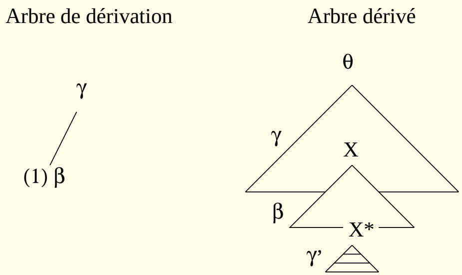


- **The adjunction** *is a contextual insertion operation of an auxiliary tree within an elementary tree*



Derivation and derived trees

- Derivation tree : describes the derived tree construction (*i.e* the strict record of the operations used to parse a sentence)
- Derived tree : syntactic structure of a sentence
- Ex: Given the trees γ and β, with β adjoined on the node 1 of γ



Unification and Features Structure

Differences with other formalism :

- Features are atomic values only and then non reentrant

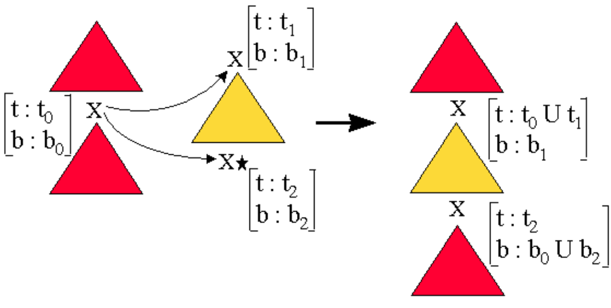
⇒ *Features are used only to control the subcategorization frame and to restrict the number of possible derivations according to a feature value*

⇒ *because of the adjunction, features are splitted into 2 fields by node : the top field and the bottom field*

⇒ *No Slash feature in TAG*

Unification and Features Structure -2

Adjunction : Update of the features



Substitution : Update of the features

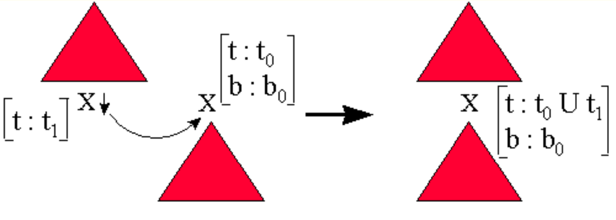
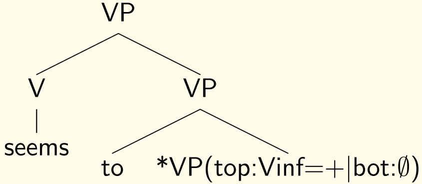


Illustration on a feature Vinf=+ :

If we want to be sure that β_2 adjoin on tree with an infinitive, we have to add some informations to this tree :

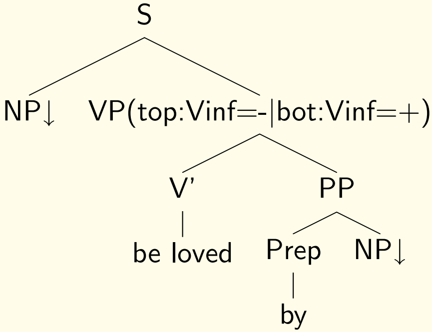
Tree β_2



Unification and Features Structure

Illustration on a feature Vinf=+ : (suite)

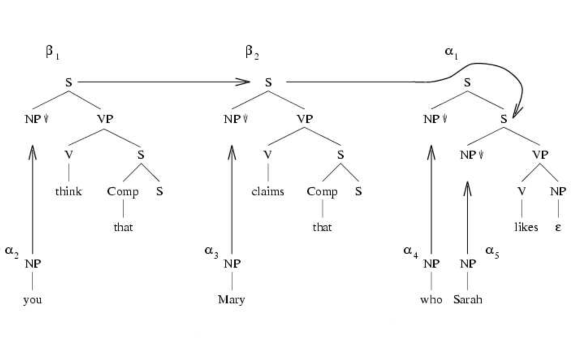
Tree α'_1



By having two different top and bottom values on the node VP, we force the adjunction of an auxiliary tree of root VP and whose foot node has the value top:Vinf=+, therefore no more unification clash.

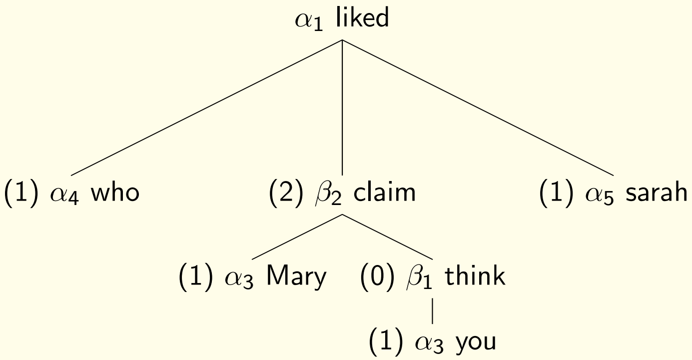
Derivation Process

Who do you think that Mary claim that Sarah liked ?

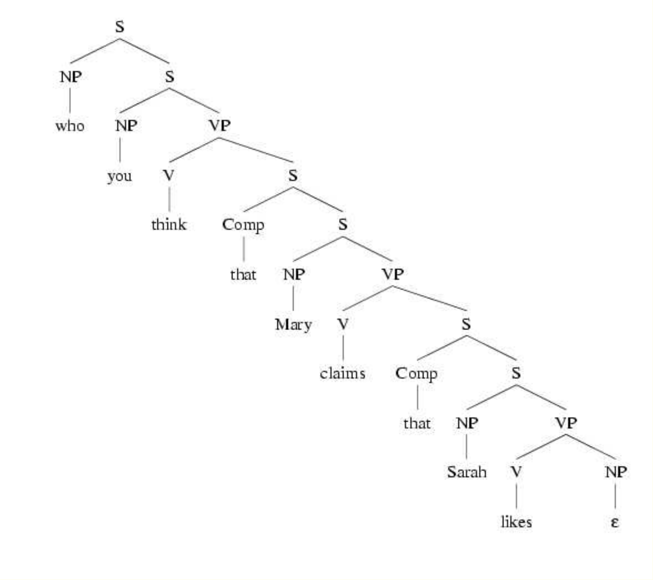


Long Distance Dependencies 2

Derivation Tree



Derived Tree



More to Come, July the 5th

Outline

- Is the Derivation Tree a good structure for Semantic ?
- Is it Possible to Use both Derived Tree and Derivation Tree for that ?
- What are Shared Forests, Derivation Forests or Dependancy Forests ?
- What more can we do than Regular LTAG ? (control, ellipsis..)
- What are Multi-Component TAG, Synchronous TAG and Metagrammars ?