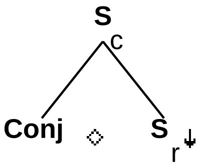
Family ”conjunctions”

March 5, 2008

1 Tree ”betaCONJs”

1.1 graphe



1.2 comments

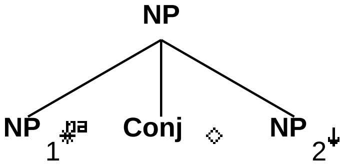
```
This tree is used for discourse conjunction,
i.e. single sentences which begin with ’and’ or ’but’.
Ex: And Truffula trees are what everyone needs.
```

1.3 features

```
S_c.b:<mode> = S_r.t:<mode>

S_c.b:<extracted> = S_r.t:<extracted>
S_c.b:<wh> = S_r.t:<wh>
S_c.b:<inv> = S_r.t:<inv>
S_c.b:<comp> = S_r.t:<comp>
S_r.t:<punct term> = nil
S_r.t:<punct struct> = comma/nil
S_r.t:<comp> = nil
S_r.t:<conj>=and/or/but/nil
S_c.b:<conj>=disc
```

2 Tree "betanx1CONJnx2"

2.1 graphe



2.2 comments

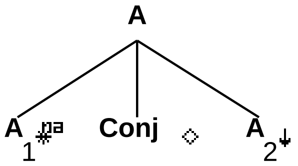
```
Tree for doing NP conjunction.  The new NP adjoins to the right of the
original NP.  The wh and case values must unify, and the agreement of
the conjoined is NP is specified , the agreement of
the conjoined is NP is specified by the conjunction, and the
determiner features are taken from the left conjunct. Conjunction may be:
'and', comma, or 'or'.
Ex: the boy and the girl
```

2.3 features

```
NP.b:<case> = NP_1.t:<case>
NP.b:<case> = NP_2.t:<case>
NP.b:<case> = nom/acc
NP.b:<wh> = NP_1.t:<wh>
NP_1.t:<wh> = NP_2.t:<wh>

Conj:<conj> = NP.b:<conj>
NP.b:<const> = NP_1.t:<const>
NP.b:<gen> = NP_1.t:<gen>
NP.b:<definite> = NP_1.t:<definite>
NP.b:<quan> = NP_1.t:<quan>
NP.b:<card> = NP_1.t:<card>
NP.b:<decreas> = NP_1.t:<decreas>
NP.b:<compar> = NP_1.t:<compar>
NP.b:<compar> = NP_2:<compar>
NP.b:<equiv> = NP_2:<equiv>
```

3 Tree "betaa1CONJa2"

3.1 graphe



3.2 comments

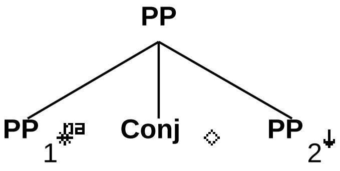
Tree for doing Adjective conjunction. The new Adjective adjoins to the right of the origi
forms. Conjunction may be:'and', comma, or 'or'.
Ex: the dark and dreary day

3.3 features

```
Conj:<conj> = A.b:<conj>
A_1.t:<neg> = -
A_2.t:<neg> = -
A.b:<neg> = -
A_1.t:<compar> = A.b:<compar>
A_2:<compar> = A.b:<compar>
A.b:<compar> = -
```

4 Tree "betapx1CONJpx2"

4.1 graphe



4.2 comments

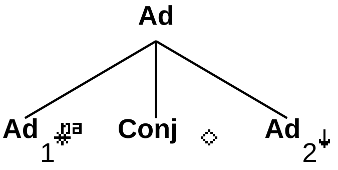
Tree for doing PP conjunction. The new PP adjoins to the
right of the original PP. There are no constraints as to
forms. Conjunction may be:'and', comma, or 'or'.
Ex: down the street and around the corner

4.3 features

```
Conj:<conj> = PP.b:<conj>
```

5 Tree "betaarb1CONJarb2"

5.1 graphe



5.2 comments

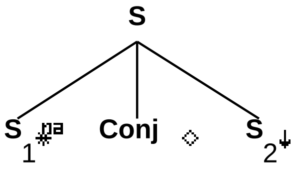
```
Tree for doing Adverb conjunction.  The new Adverb adjoins to the
right of the original Adverb.  There are no constraints as to forms. Conjunction may be:'a
Ex: slowly and carefully
```

5.3 features

```
Conj:<conj> = Ad.b:<conj>
```

6 Tree "betas1CONJs2"

6.1 graphe



6.2 comments

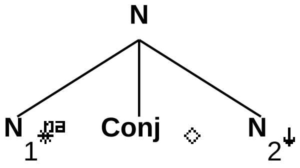
```
Tree for doing sentential conjunction.  The new sentence adjoins to the
right of the original sentence.  The modes of the two sentences are
constrained to be the same, and may be of type ind/inf/ger/nom/prep/imp.
Null complementizers are allowed for inf and ind sentences for both
sentences.  Note that this tree does not do VP conjunction. Conjunction may be:'and', comm
Ex: the day was dark and the phone never rang.
```

6.3 features

```
S_1.t:<mode> = S_2.t:<mode>
S.b:<mode> = S_1.t:<mode>
S_1.t:<assign-comp> = inf_nil/ind_nil
S_2.t:<assign-comp> = inf_nil/ind_nil
S_1.t:<mode> = ind/inf/ger/nom/prep/imp

Conj:<conj> = S.b:<conj>
S_1.t:<wh> = S_2.t:<wh>
S.b:<wh> = S_1.t:<wh>
S_1.t:<extracted> = S_2.t:<extracted>
S.b:<extracted> = S_1.t:<extracted>
S_1.t:<comp> = S_2.t:<comp>
S.b:<comp> = S_1.t:<comp>
```

7 Tree "betan1CONJn2"

7.1 graphe



7.2 comments

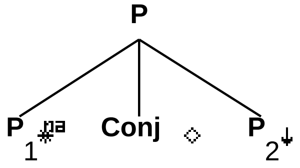
```
Tree for doing N conjunction.  The new N adjoins to the right of the
original N.  The wh and case values must unify, and the agreement of
the conjoined is N is specified to be plural. Conjunction may be:'and', comma, or 'or'.
Ex: the (boy and girl)
```

7.3 features

```
N.b:<case> = N_1.t:<case>
N.b:<case> = N_2.t:<case>
N.b:<case> = nom/acc
N.b:<wh> = N_1.t:<wh>
N_1.t:<wh> = N_2.t:<wh>

Conj:<conj> = N.b:<conj>
N.b:<const> = N_1.t:<const>
N.b:<gen> = N_1.t:<gen>
N.b:<definite> = N_1.t:<definite>
N.b:<quan> = N_1.t:<quan>
N.b:<card> = N_1.t:<card>
N.b:<decreas> = N_1.t:<decreas>
N.b:<compar> = N_1.t:<compar>
N.b:<compar> = N_2:<compar>
N.b:<compar> = -
```

8 Tree "betap1CONJp2"

8.1 graphe



8.2 comments

```
Tree for doing P conjunction.  The new P adjoins to the
```

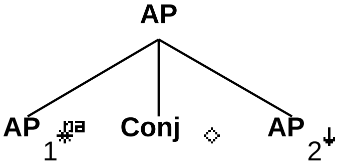
right of the original P. There are no constraints as to forms. Conjunction may be:'and',
Ex: to and from the office

8.3 features

```
P.b:<assign-case> = P_1.t:<assign-case>

Conj:<conj> = P.b:<conj>
```

9 Tree "betaax1CONJax2"

9.1 graphe



9.2 comments

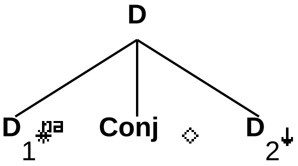
Tree for doing Adjective Phrase conjunction. The new AP adjoins to the left of the origin
forms. Conjunction may be:'and', comma, or 'or'.
Ex: Bill is certain that Bill is still alive and confident that he will reappear

9.3 features

```
Conj:<conj> = AP.b:<conj>
AP_1.t:<compar> = AP.b:<compar>
AP_2:<compar> = AP.b:<compar>
AP.b:<equiv> = AP_2:<equiv>
```

10 Tree "betad1CONJd2"

10.1 graphe



10.2 comments

Tree for doing determiner conjunction. The new determiner adjoins to the right of the original determiner. This is because the determiner phrases generally build to the left. The wh value of the two determiners are constained to be the same, and all other feature

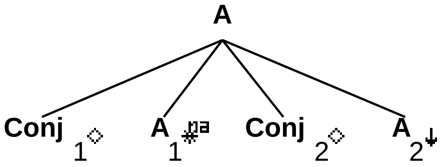
values are taken from the left D (again, since determiners build to the left). Conjunction may be:'and', comma, or 'or'.
Ex: All but one

10.3 features

```
D.b:<definite> = D_1.t:<definite>
D.b:<agr> = D_1.t:<agr>
D.b:<quan> = D_1.t:<quan>
D.b:<card> = D_1.t:<card>
D.b:<gen> = D_1.t:<gen>
D.b:<decreas> = D_1.t:<decreas>
D.b:<const> = D_1.t:<const>
D.b:<wh> = D_1.t:<wh>
D.b:<wh> = D_2.t:<wh>
Conj:<conj> = D.b:<conj>
```

11 Tree "betaCONJa1CONJa2"

11.1 graphe



11.2 comments

Tree for doing Adjective conjunction. The new Adjective adjoins to the right of the origi
forms. The multi-word pairs either-or, neither-nor and both-and select this tree.
Ex: both yellow and green dogs

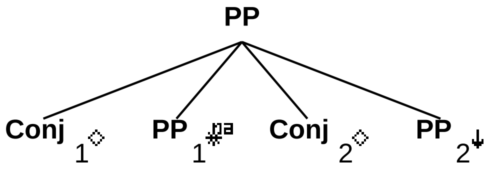
Prob: how to block
*the both yellow and green dogs

11.3 features

```
Conj_1:<conj> = A.b:<conj>
A_1.t:<compar> = A.b:<compar>
A_2:<compar> = A.b:<compar>
A_2:<equiv> = A.b:<equiv>
```

12 Tree "betaCONJpx1CONJpx2"

12.1 graphe



12.2 comments

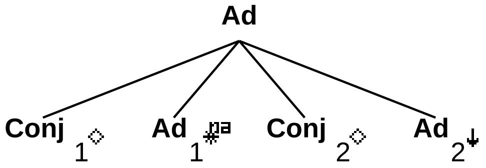
```
Tree for doing multi-word PP conjunction.  The new PP adjoins to the
right of the original PP.  There are no constraints as to
forms. The multi-word pairs either-or, neither-nor and both-and select
this tree.
Ex: either down the street or around the corner
   both in your homes and in your schools
```

12.3 features

```
Conj_1:<conj> = PP.b:<conj>
```

13 Tree "betaCONJarb1CONJarb2"

13.1 graphe



13.2 comments

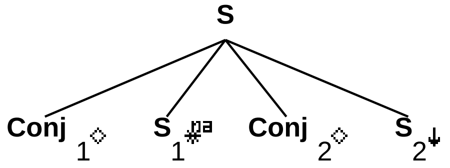
```
Tree for doing Adverb conjunction.  The new Adverb adjoins to the
right of the original Adverb.  There are no constraints as to forms. The multi-word pairs
this tree.
Ex: either very quickly or very slowly
    neither brilliantly nor well
```

13.3 features

```
Conj_1:<conj> = Ad.b:<conj>
```

14 Tree "betaCONJs1CONJs2"

14.1 graphe



14.2 comments

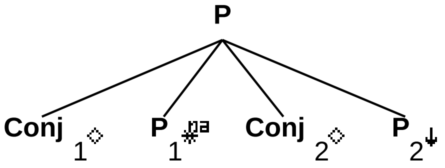
```
Tree for doing sentential conjunction.  The new sentence adjoins to the
right of the original sentence. The multi-word pairs either-or, neither-nor and both-and s
this tree. The modes of the two sentences are
constrained to be the same, and may be of type ind/inf/ger/nom/prep/imp.
Null complementizers are allowed for inf and ind sentences for both
sentences.  Note that this tree does not do VP conjunction.
Ex: Either Mary will attend the meeting or she will listen via
conference call.
Prob case:
   I'm either going to the store or to the gym.

       In protege2, a method EITHER is undecomposable, in which case
we call it a mechanism, OR it can bedecomposed into a set of subtasks.


These cases should be handled with multi-component sets, of XconjX plus
'either' anchoring an adverb-like tree.
```

14.3 features

```
S_1.t:<mode> = S_2.t:<mode>
S.b:<mode> = S_1.t:<mode>
S_1.t:<assign-comp> = inf_nil/ind_nil
S_2.t:<assign-comp> = inf_nil/ind_nil
S_1.t:<mode> = ind/inf/ger/nom/prep/imp

Conj_1:<conj> = S.b:<conj>
S_1.t:<wh> = S_2.t:<wh>
S.b:<wh> = S_1.t:<wh>
S_1.t:<extracted> = S_2.t:<extracted>
S.b:<extracted> = S_1.t:<extracted>
S_1.t:<comp> = S_2.t:<comp>
S_1:<comp> = S.b:<comp>
```

15 Tree "betaCONJp1CONJp2"

15.1 graphe



15.2 comments

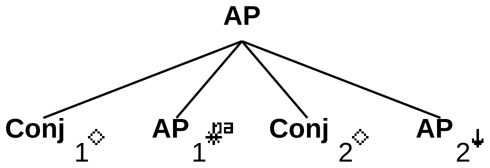
```
Tree for doing P conjunction.  The new P adjoins to the
right of the original P.  There are no constraints as to forms. The multi-word pairs eithe
this tree.

Ex: either for or against Clinton
    both in and out of school
```

15.3 features

```
P.b:<assign-case> = P_1.t:<assign-case>

Conj_1:<conj> = P.b:<conj>
```

16 Tree "betaCONJax1CONJax2"

16.1 graphe



16.2 comments

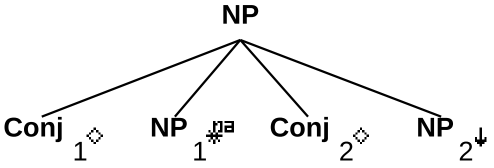
```
Tree for doing Adjective Phrase conjunction.  The new AP adjoins to the left of the origin
forms. The multi-word pairs either-or, neither-nor and both-and select
this tree.
Ex: Bill is either certain that Bob is still alive or confident that he will reappear
```

16.3 features

```
Conj_1:<conj> = AP.b:<conj>
AP_1.t:<compar> = AP.b:<compar>
AP_2:<compar> = AP.b:<compar>
AP_2:<equiv> = AP.b:<equiv>
```

17 Tree "betaCONJnx1CONJnx2"

17.1 graphe



17.2 comments

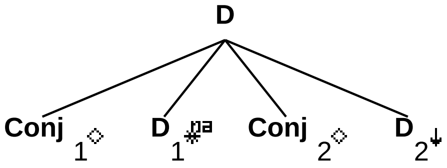
```
Tree for doing NP conjunction.  The new NP adjoins to the right of the
original NP.  The wh and case values must unify, the agreement of
the conjoined is NP is specified by the conjunction, and the
determiner features are taken from the left conjunct. The multi-word pairs either-or, neit
this tree.
Ex: neither the boy nor the girl
```

17.3 features

```
NP.b:<case> = NP_1.t:<case>
NP.b:<case> = NP_2.t:<case>
NP.b:<case> = nom/acc
NP.b:<wh> = NP_1.t:<wh>
NP_1.t:<wh> = NP_2.t:<wh>

Conj_1:<conj> = NP.b:<conj>
NP.b:<const> = NP_1.t:<const>
NP.b:<gen> = NP_1.t:<gen>
NP.b:<definite> = NP_1.t:<definite>
NP.b:<quan> = NP_1.t:<quan>
NP.b:<card> = NP_1.t:<card>
NP.b:<decreas> = NP_1.t:<decreas>
```

18 Tree "betaCONJd1CONJd2"

18.1 graphe



18.2 comments

```
Tree for doing multi-word determiner conjunction. The multi-word pairs either-or, neither-
this tree. The new determiner adjoins to
```

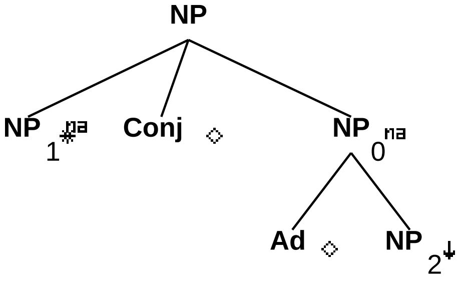
the right of the original determiner. This is because the determiner phrases generally build to the left. The wh value of the two determiners are constained to be the same, and all other feature values are taken from the left D (again, since determiners build to the left).
Ex: either five or ten
    both this and that
Prob: There are no constraints on the two Dets having similar features; this needs to be added to block, eg. *'the and ten men'

18.3 features

```
D.b:<definite> = D_1.t:<definite>
D.b:<agr> = D_1.t:<agr>
D.b:<quan> = D_1.t:<quan>
D.b:<card> = D_1.t:<card>
D.b:<gen> = D_1.t:<gen>
D.b:<decreas> = D_1.t:<decreas>
D.b:<const> = D_1.t:<const>
D.b:<wh> = D_1.t:<wh>
D.b:<wh> = D_2.t:<wh>
Conj_1:<conj> = D.b:<conj>
```

19 Tree "betanx1CONJARBnx2"

19.1 graphe



19.2 comments

Tree for doing NP conjunction with 'but not',
'e not' and 'and not'. The new NP adjoins to the right of the original NP. The wh and case values must unify, and the determiner features are taken from the left conjunct..
Ex: the boy not the girl, Joe and not John

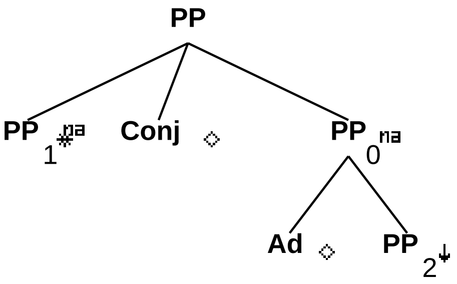
19.3 features

```
NP.b:<neg>=+
Conj:<conj> = NP.b:<conj>
```

```
NP.b:<case> = NP_2.t:<case>
NP.b:<case> = nom/acc
NP.b:<case> = NP_1.t:<case>

NP.b:<wh> = NP_1.t:<wh>
NP_1.t:<wh> = NP_2.t:<wh>
NP.b:<const> = NP_1.t:<const>
NP.b:<gen> = NP_1.t:<gen>
NP.b:<definite> = NP_1.t:<definite>
NP.b:<quan> = NP_1.t:<quan>
NP.b:<card> = NP_1.t:<card>
NP.b:<decreas> = NP_1.t:<decreas>
```

20 Tree ”betapx1CONJARBpx2”

20.1 graphe



20.2 comments

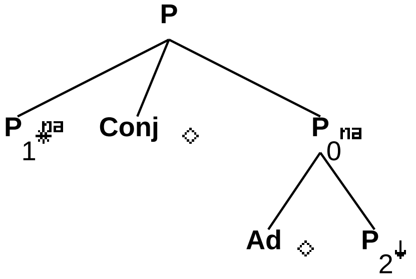
Tree for doing PP conjunction with ’but not’,
’e not’ and ’and not’. The new PP
adjoins to the right of the original PP. There are no constraints as to
forms.
Ex: down the street and not to the end

20.3 features

```
PP.b:<neg>=+

Conj:<conj> = PP.b:<conj>
```

21 Tree "betap1CONJARBp2"

21.1 graphe



21.2 comments

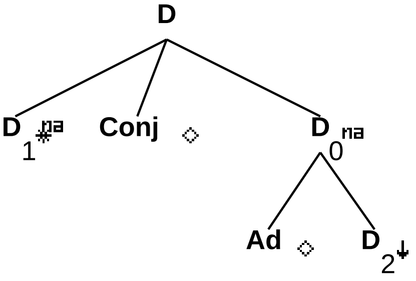
```
Tree for doing P conjunction with 'but not',
'e not' and 'and not'.  The new P adjoins to the  right of the
original P.  There are no constraints as to forms.
Ex: near not in the office, before and not after the race
```

21.3 features

```
P.b:<neg>=+
P.b:<assign-case> = P_1.t:<assign-case>

Conj:<conj> = P.b:<conj>
```

22 Tree "betad1CONJARBd2"

22.1 graphe



22.2 comments

```
Tree for doing determiner conjunction with 'but not',
'e not' and 'and not'.  The new determiner adjoins to the right of
the original determiner.  This is because the determiner phrases
generally build to the left.  The wh value of the two determiners are
constained to be the same, and all other feature values are taken from
the left D (again, since determiners build to the left).
```

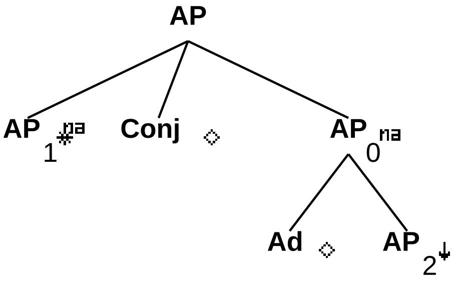
```
Ex: All not some
```

22.3 features

```
D.b:<neg>=+
D.b:<wh> = D_2.t:<wh>
Conj:<conj> = D.b:<conj>

D.b:<definite> = D_1.t:<definite>
D.b:<agr> = D_1.t:<agr>
D.b:<quan> = D_1.t:<quan>
D.b:<card> = D_1.t:<card>
D.b:<gen> = D_1.t:<gen>
D.b:<decreas> = D_1.t:<decreas>
D.b:<const> = D_1.t:<const>
D.b:<wh> = D_1.t:<wh>
```

23 Tree "betaax1CONJARBax2"

23.1 graphe



23.2 comments

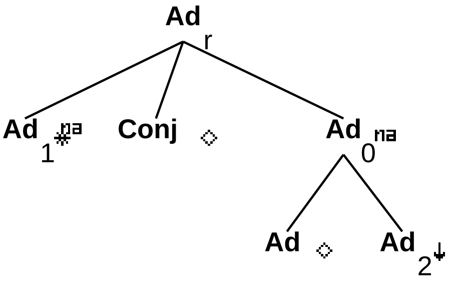
```
Tree for doing Adjective Phrase conjunction with 'but not',
'e not'  and 'and not'.  The new AP adjoins to the left of the
original AP.  There are no constraints as to forms.
Ex: Bill is certain that Max is still alive but not confident that he will reappear
```

23.3 features

```
AP.b:<neg>=+

Conj:<conj> = AP.b:<conj>
AP_1.t:<compar> = AP.b:<compar>
AP_2:<compar> = AP.b:<compar>
AP_2:<equiv> = AP.b:<equiv>
```

24 Tree "betaarb1CONJARBarb2"

24.1 graphe



24.2 comments

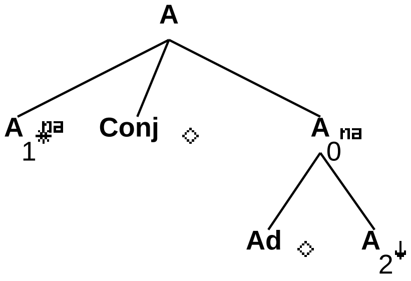
```
Tree for doing Adverb conjunction with 'but not',
'e not'  and 'and not'.  The new Adverb
adjoins to the right of the original Adverb.  There are no constraints
as to forms.  Ex: slowly but not carefully, slowly and painstakingly
```

24.3 features

```
Conj:<conj> = Ad_r.b:<conj>
```

25 Tree "betaa1CONJARBa2"

25.1 graphe



25.2 comments

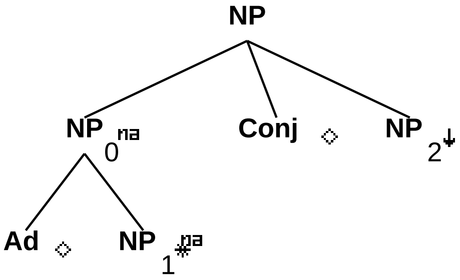
```
Tree for doing Adjective conjunction with 'but not',
'e not'  and 'and not'.
The new Adjective adjoins to the right of the original Adjective.
There are no constraints as to forms.

Ex: dark but not dreary days, yellow not green bananas
```

25.3 features

```
A.b:<neg>=+

Conj:<conj> = A.b:<conj>
A_1.t:<compar> = A.b:<compar>
A_2:<compar> = A.b:<compar>
A_2:<equiv> = A.b:<equiv>
```

26 Tree "betaARBnx1CONJnx2"

26.1 graphe



26.2 comments

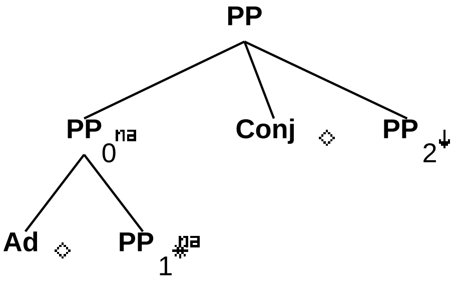
Tree for doing NP conjunction with ‘not but’. The new NP
adjoins to the right of the original NP. The wh and case values must unify.
Ex: not the boy but the girl

26.3 features

```
NP.b:<neg>=+
NP.b:<wh> = NP_2.t:<wh>
NP.b:<case> = NP_2.t:<case>
NP.b:<case> = nom/acc
NP.b:<wh> = NP_0.t:<wh>
NP.b:<case> = NP_0.t:<case>

NP_0.b:<case> = NP_1.t:<case>
NP_0.b:<wh> = NP_1.t:<wh>

Conj:<conj> = NP.b:<conj>
```

27 Tree ”betaARBpx1CONJpx2”

27.1 graphe



27.2 comments

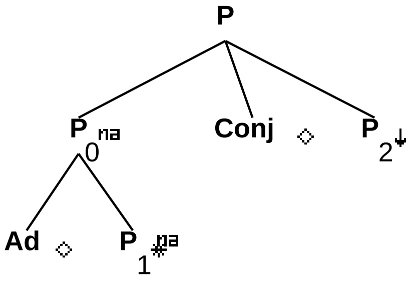
```
Tree for doing PP conjunction with ‘not but’.  The new PP
adjoins to the  right of the original PP.  There are no constraints as to
forms.
Ex: not down the street but around the corner
```

27.3 features

```
PP.b:<neg>=+


Conj:<conj> = PP.b:<conj>
```

28 Tree ”betaARBp1CONJp2”

28.1 graphe



28.2 comments

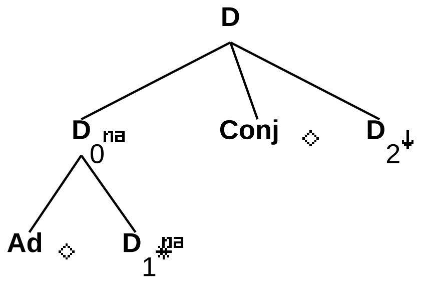
```
Tree for doing P conjunction with ‘not but’.  The new P
adjoins to the  right of the original P.  There are no constraints as
to forms.  Ex: not near but in the office
```

28.3 features

```
P.b:<neg>=+
P.b:<assign-case> = P_0.t:<assign-case>

P_0.b:<assign-case> = P_1.t:<assign-case>

Conj:<conj> = P.b:<conj>
```

29 Tree "betaARBd1CONJd2"

29.1 graphe



29.2 comments

```
Tree for doing determiner conjunction with ‘not but’.  The new determiner adjoins to
the right of the original determiner.  This is because the determiner
phrases generally build to the left.  The wh value of the two
determiners are constained to be the same, and all other feature
values are taken from the left D (again, since determiners build to
the left).
Ex: Not all but some
```

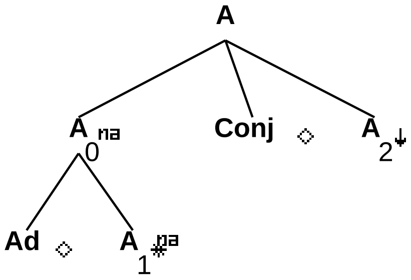
29.3 features

```
D.b:<neg>=+
D.b:<wh> = D_2.t:<wh>
Conj:<conj> = D.b:<conj>

D.b:<definite> = D_0.t:<definite>
D.b:<agr> = D_0.t:<agr>
D.b:<quan> = D_0.t:<quan>
D.b:<card> = D_0.t:<card>
D.b:<gen> = D_0.t:<gen>
D.b:<decreas> = D_0.t:<decreas>
D.b:<const> = D_0.t:<const>
D.b:<wh> = D_0.t:<wh>

D_0.b:<definite> = D_1.t:<definite>
D_0.b:<agr> = D_1.t:<agr>
```

```
D_0.b:<quan> = D_1.t:<quan>
D_0.b:<card> = D_1.t:<card>
D_0.b:<gen> = D_1.t:<gen>
D_0.b:<decreas> = D_1.t:<decreas>
D_0.b:<const> = D_1.t:<const>
D_0.b:<wh> = D_1.t:<wh>
```

30 Tree "betaARBa1CONJa2"

30.1 graphe



30.2 comments

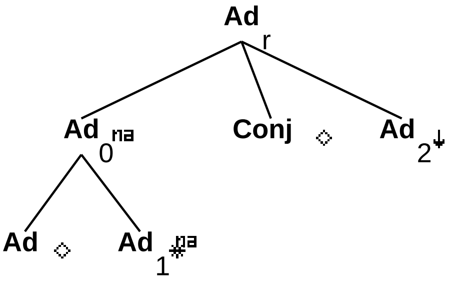
```
Tree for doing Adjective conjunction with 'not but'.  The new
Adjective adjoins to the right of the original Adjective.  There are
no constraints as to forms.
Ex: not dark but dreary days
```

30.3 features

```
A.b:<neg>=+


Conj:<conj> = A.b:<conj>
A_1.t:<compar> = A.b:<compar>
A_2:<compar> = A.b:<compar>
A_2:<equiv> = A.b:<equiv>
```

31 Tree "betaARBarb1CONJarb2"

31.1 graphe



31.2 comments

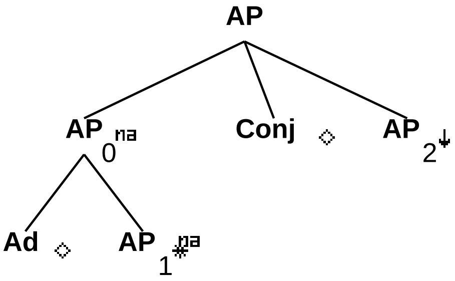
```
Tree for doing Adverb conjunction with 'not but'.  The new Adverb
adjoins to the right of the original Adverb.  There are no constraints
as to forms.  Ex: not slowly but carefully
```

31.3 features

```
Conj:<conj> = Ad_r.b:<conj>
```

32 Tree "betaARBax1CONJax2"

32.1 graphe



32.2 comments

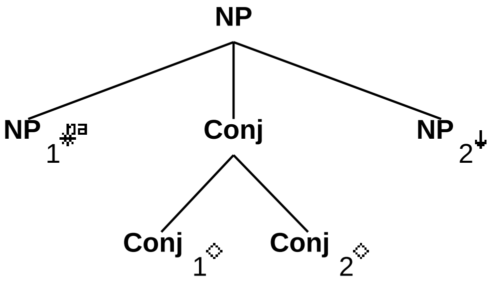
```
Tree for doing Adjective Phrase conjunction with 'not
but'.  The new AP adjoins to the left of the original AP.  There are
no constraints as to forms.
Ex: Bill is not certain to win the election but confident that he will
perform well
```

32.3 features

```
AP.b:<neg>=+
Conj:<conj> = AP.b:<conj>
AP_1.t:<compar> = AP.b:<compar>
AP_2:<compar> = AP.b:<compar>
AP_2:<equiv> = AP.b:<equiv>
```

33 Tree ”betanx1CONJCONJnx2”

33.1 graphe



33.2 comments

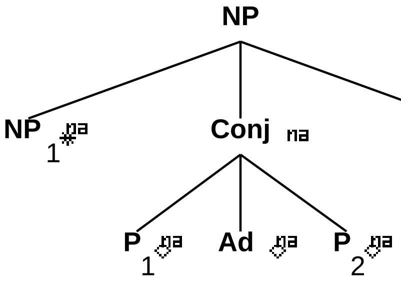
```
Tree for ', and'

Max, Bill, and Sue

Question is whether to have another level built in, to force this to
have a comma above it or to do it with features.

Changed conj2 to anchor from subst. node. Removed Conj entry from
conj.text:
INDEX: ,/9
ENTRY: ,
POS: Conj
TREES: alphaConj
FS: #Comma_conj
```

33.3 features

```
NP.b:<case> = NP_1.t:<case>
NP.b:<wh> = NP_1.t:<wh>

NP.b:<case> = nom/acc
NP_1.t:<wh> = NP_2.t:<wh>
NP_1.t:<case> = NP_2.t:<case>
NP.b:<conj> = Conj.t:<conj>
Conj.b:<conj> = Conj_2.t:<conj>
```

34 Tree "betanx1PARBPnx2"

34.1 graphe



34.2 comments

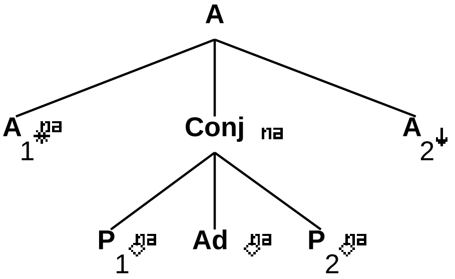
```
Tree for doing NP conjunction.  The new NP adjoins to the right of the
original NP.  The wh and case values must unify, and the agreement of
the conjoined is NP is specified , the agreement of
the conjoined is NP is specified by the conjunction, and the
determiner features are taken from the left conjunct.  In this case, the
conjunction is multi-word, namely, 'as well as'.

EX: That would benefit Chinese enterprises as well as Sino-foreign joint
ventures.
```

34.3 features

```
NP.b:<case> = NP_1.t:<case>
NP.b:<case> = NP_2.t:<case>
NP.b:<case> = nom/acc
NP.b:<wh> = NP_1.t:<wh>
NP_1.t:<wh> = NP_2.t:<wh>

Conj:<conj> = NP.b:<conj>
NP.b:<const> = NP_1.t:<const>
NP.b:<gen> = NP_1.t:<gen>
NP.b:<definite> = NP_1.t:<definite>
NP.b:<quan> = NP_1.t:<quan>
NP.b:<card> = NP_1.t:<card>
NP.b:<decreas> = NP_1.t:<decreas>
```

35 Tree "betaa1PARBPa2"

35.1 graphe



35.2 comments

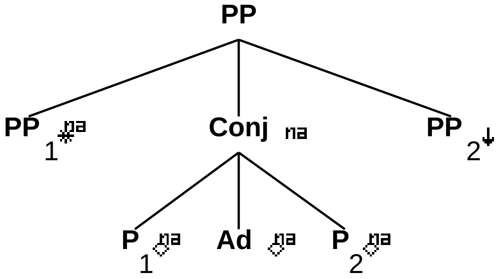
```
Tree for doing Adjective conjunction.  The new Adjective adjoins to the right
of the original Adjective.  There are no constraints as to forms. This tree
handles conjunction of adjectives by multi-word conjunctions, namely,
'as well as'.

EX: We do not discriminate in terms of size; tall as well as short people are
included in our groups.
```

35.3 features

```
Conj:<conj> = A.b:<conj>
A_1.t:<neg> = -
A_2.t:<neg> = -
A.b:<neg> = -
A_1.t:<compar> = A_2:<compar>
A_1.t:<compar> = A.b:<compar>
A_2:<equiv> = A.b:<equiv>
```

36 Tree "betapx1PARBPpx2"

36.1 graphe



36.2 comments

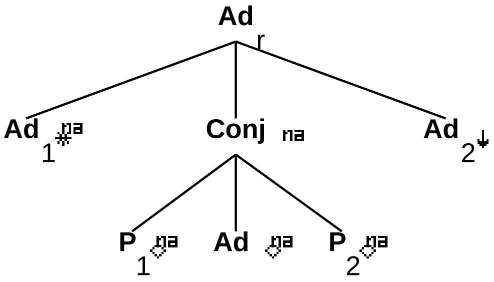
```
Tree for doing PP conjunction.  The new PP adjoins to the
right of the original PP.  There are no constraints as to
forms. This tree handles the multi-word conjunction 'as well as'.
Ex: down the street as well as around the corner
```

36.3 features

```
Conj:<conj> = PP.b:<conj>
```

37 Tree "betaarb1PARBParb2"

37.1 graphe



37.2 comments

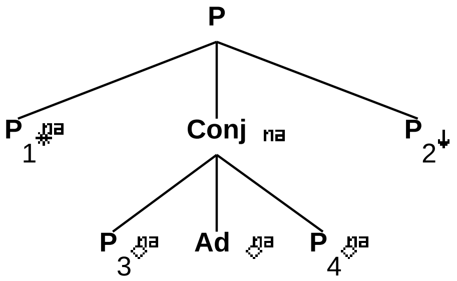
```
Tree for doing Adverb conjunction.  The new Adverb adjoins to the
right of the original Adverb.  There are no constraints as to forms. This tree
handles the multi-word conjunction 'as well as'.
Ex: He walked gingerly as well as swiftly through the woods.
```

37.3 features

```
Conj:<conj> = Ad_r.b:<conj>
```

38 Tree "betap1PARBPp2"

38.1 graphe



38.2 comments

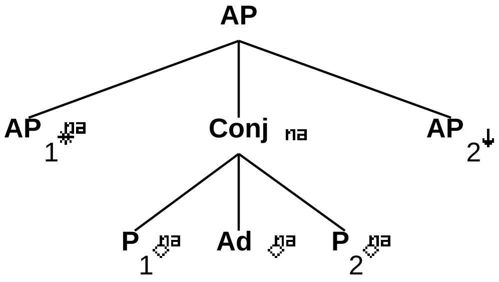
Tree for doing P conjunction. The new P adjoins to the
right of the original P. There are no constraints as to forms. This tree
handles the multi-word conjunction 'as well as'.
EX: He went to as well as from the office by bicycle.

38.3 features

```
P.b:<assign-case> = P_1.t:<assign-case>

Conj:<conj> = P.b:<conj>
```

39 Tree "betaax1PARBPax2"

39.1 graphe



39.2 comments

Tree for doing Adjective Phrase conjunction. The new AP adjoins to the left of the origin
forms. This handles the multi-word conjunction 'as well as'.
Ex: Bill is certain that Bill is still alive as well as confident that he will reappear

39.3 features

```
Conj:<conj> = AP.b:<conj>
AP_1.t:<compar> = AP.b:<compar>
AP_2:<compar> = AP.b:<compar>
AP.b:<equiv> = AP_2:<equiv>
```