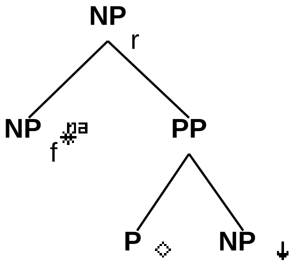
Family ”prepositions”

March 5, 2008

1 Tree ”betanxPnx”

1.1 graphe



1.2 comments

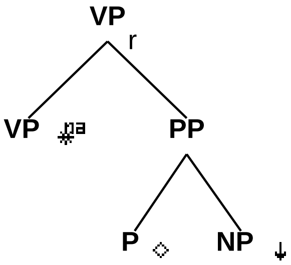
```
Prepositional phrase
```

1.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>

NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>
NP:<wh> = -
NP_f.b:<case> = acc/nom
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<gerund> = NP_r.b:<gerund>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

2 Tree "betavxPnx"

2.1 graphe



2.2 comments

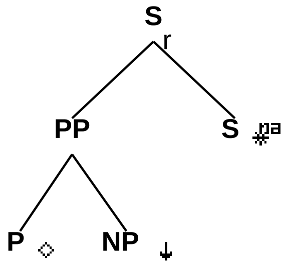
```
NIL
```

2.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP_r.b:<conj> = VP.t:<conj>
VP.t:<compar> = P.t:<compar>
VP.t:<equiv> = P.t:<equiv>
VP_r.b:<compar> = -
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

3 Tree "betaPnxs"

3.1 graphe



3.2 comments

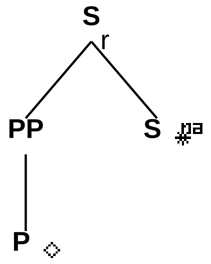
```
Sentential adverbial tree
Adverb on the left of the sentence
 'Obviously John loves Mary'
```

3.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<conj> = S_r.b:<conj>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

4 Tree "betaPs"

4.1 graphe



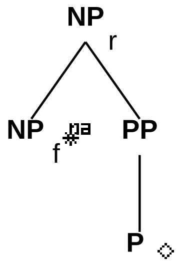
4.2 comments

4.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>
S.t:<inv> = S_r.b:<inv>
S.t:<inv> = S_r.b:<invlink>
S.t:<conj> = S_r.b:<conj>
S.t:<extracted> = S_r.b:<extracted>
S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = P.t:<wh>

S.b:<comp> = nil
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

5 Tree "betanxP"

5.1 graphe



5.2 comments

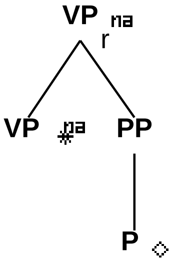
```
Prepositional phrase
```

5.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
NP_f.b:<case> = acc/nom

NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
P.t:<wh> = -
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

6 Tree ”betavxP”

6.1 graphe



6.2 comments

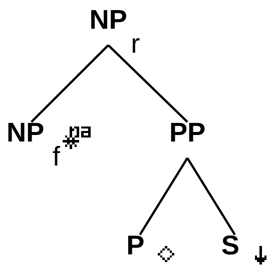
```
NIL
```

6.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
VP.t:<conj> = VP_r.b:<conj>
P.b:<wh> = PP.b:<wh>
VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

7 Tree ”betanxPs”

7.1 graphe



7.2 comments

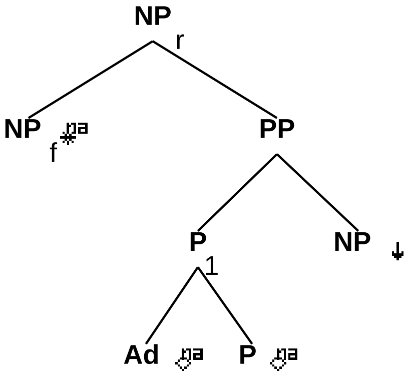
Prepositional phrase

7.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
NP_f.b:<case> = acc/nom

NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
S.t:<mode> = ind/prep/nom
S.t:<comp> = nil
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

8 Tree "betanxARBPnx"

8.1 graphe



8.2 comments

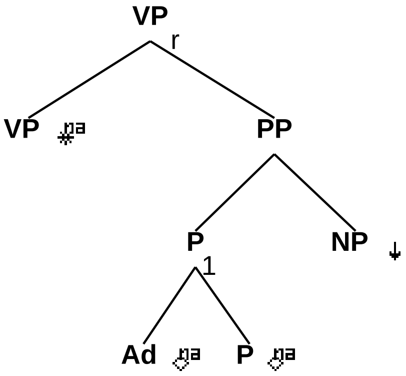
```
PP modifying NP
Two-word P, first word is an Ad
'The girl next to the palm just winked at you'
```

8.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>
NP:<wh> = -
NP_f.b:<case> = acc/nom

NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

9 Tree ”betavxARBPnx”

9.1 graphe



9.2 comments

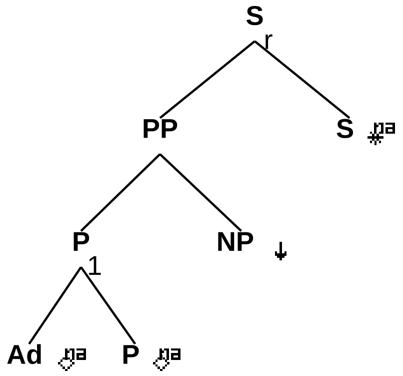
```
PP modifying VP
Two-word P, first word is Ad
'Yahoo ran ahead of the pack'
```

9.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

10 Tree "betaARBPnxs"

10.1 graphe



10.2 comments

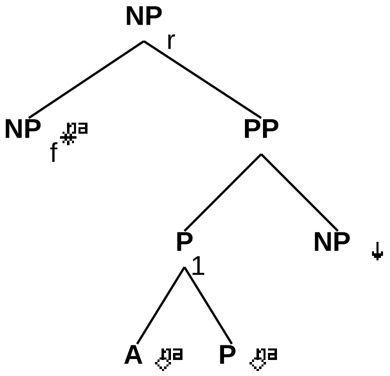
```
PP modifying S
Two-word P, first word is Ad
 'Instead of walking to the dumpster,
 he threw his banana peel down the
 sewer grate'
```

10.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<conj> = S_r.b:<conj>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

11 Tree ”betanxAPnx”

11.1 graphe



11.2 comments

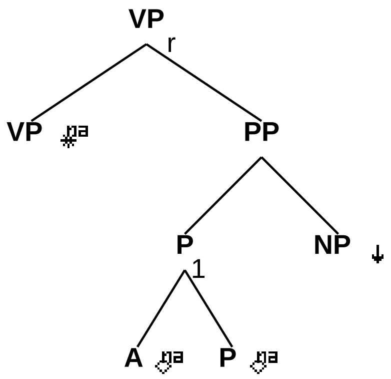
```
PP modifying NP
Two-word P, first word is A
'The woman devoid of all hope
 began to shout at the people
 passing by'
```

11.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>
NP:<wh> = -
NP_f.b:<case> = acc/nom

NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<gerund> = NP_r.b:<gerund>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

12 Tree "betavxAPnx"

12.1 graphe



12.2 comments

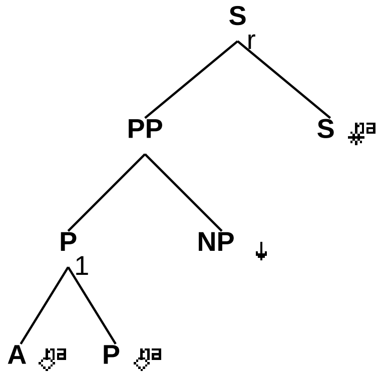
```
PP modifying VP
Two-word P, first word is A
'I changed the message prior to leaving
 the house'
```

12.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

13 Tree "betaAPnxs"

13.1 graphe



13.2 comments

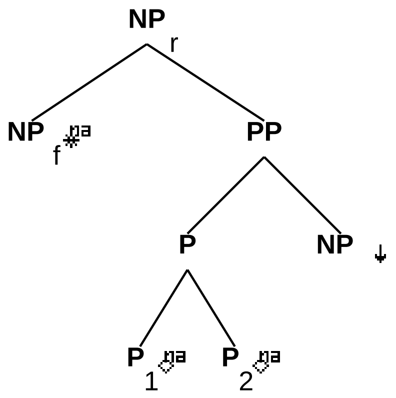
```
PP modifying S
Two-word P, first word is A
'Preparatory to the exam, the children
 learned an extensive vocabulary'
```

13.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<conj> = S_r.b:<conj>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

14 Tree "betanxPPnx"

14.1 graphe



14.2 comments

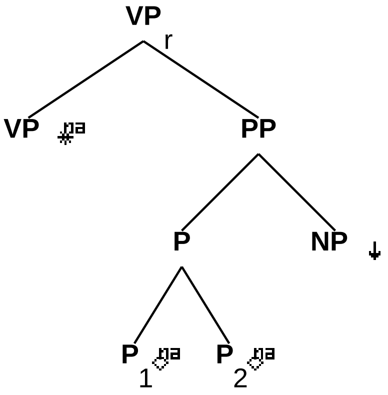
```
PP modifying NP
Two-word P, first word is a P
'The dog inside of that house
 is going crazy'
```

14.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>
NP:<wh> = -
NP_f.b:<case> = acc/nom

NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<gerund> = NP_r.b:<gerund>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

15 Tree "betavxPPnx"

15.1 graphe



15.2 comments

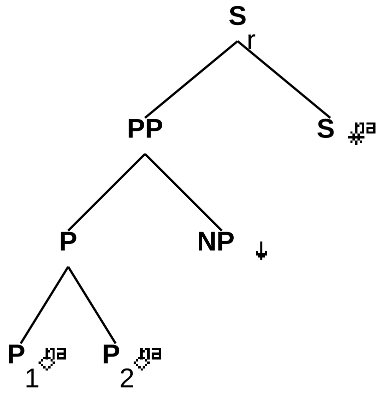
```
PP modifying VP
Two-word P, first word is P
'The jester walked outside of
 the building to meet with his
 death sentence'
```

15.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

16 Tree "betaPPnxs"

16.1 graphe



16.2 comments

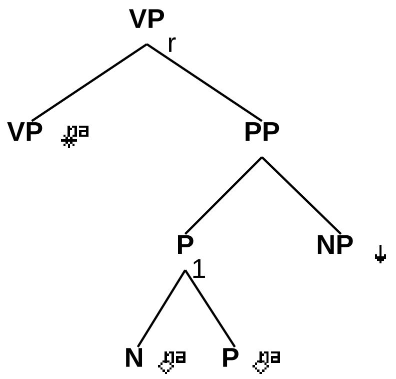
```
PP modifying S
Two-word P, first word is P
'Along with the rest of the
 pack, we are going to climb
 this mountain'
```

16.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<conj> = S_r.b:<conj>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

17 Tree ”betavxNPnx”

17.1 graphe



17.2 comments

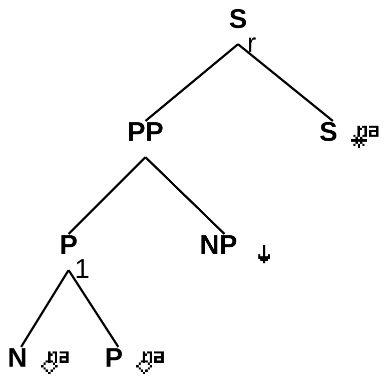
```
PP modifying VP
Two-word P, first word is N
'We arrived safely thanks to the sunny
 skies'
```

17.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

18 Tree "betaNPnxs"

18.1 graphe



18.2 comments

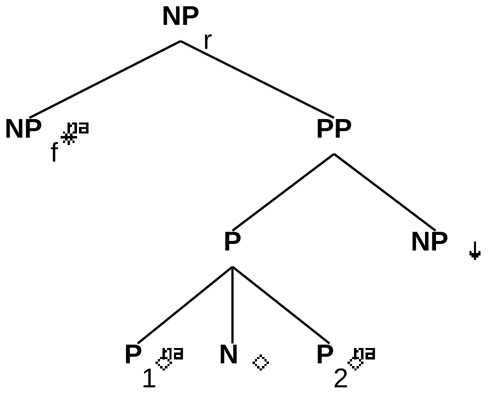
```
PP modifying S
Two-word P, first word is N
'Thanks to you, we won't be able to
 leave tonight'
```

18.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P_1.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

19 Tree ”betanxPNPnx”

19.1 graphe



19.2 comments

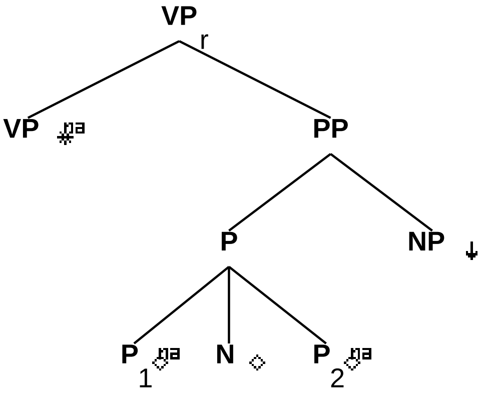
```
PP modifying NP
Three-word P
'The peasant in need of water just
 trapsed across the dais'
```

19.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>
NP:<wh> = -
NP_f.b:<case> = acc/nom

NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<gerund> = NP_r.b:<gerund>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

20 Tree "betavxPNPnx"

20.1 graphe



20.2 comments

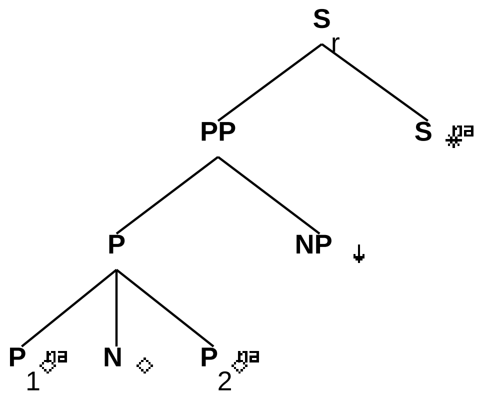
```
PP modifying VP
Three-word P
'She succeeded merely by means of her wit'
```

20.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

21 Tree "betaPNPnxs"

21.1 graphe



21.2 comments

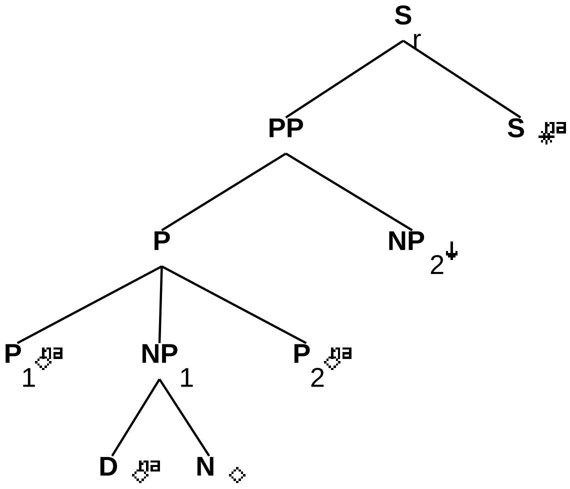
```
PP modifying S
Three-word P
'On behalf of all of us here tonight,
 we would like to thank you for you
 never-ending generosity'
```

21.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

22 Tree ”betaPDNPnxs”

22.1 graphe



22.2 comments

```
PP modifying S
Three-word P
The noun can take a determiner,
 and the determiner is an anchor of the P.
’On the grounds of insufficient proof,  the judge dismissed the case’
```

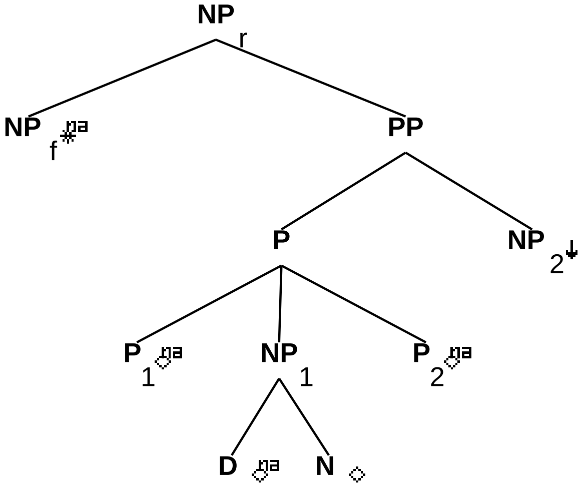
22.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP_2:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP_2:<case>
NP_1.b:<agr> = N.t:<agr>
NP_1.b:<case> = N.t:<case>
NP_1.b:<conj> = N.t:<conj>
```

```
NP_1.b:<const> = D.t:<const>
NP_1.b:<definite> = D.t:<definite>
NP_1.b:<quan> = D.t:<quan>
NP_1.b:<card> = D.t:<card>
NP_1.b:<gen> = D.t:<gen>
NP_1.b:<decreas> = D.t:<decreas>
NP_1.b:<wh> = D.t:<wh>
NP_1.t:<case> = nom/acc
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

23 Tree "betanxPDNPnx"

23.1 graphe



23.2 comments

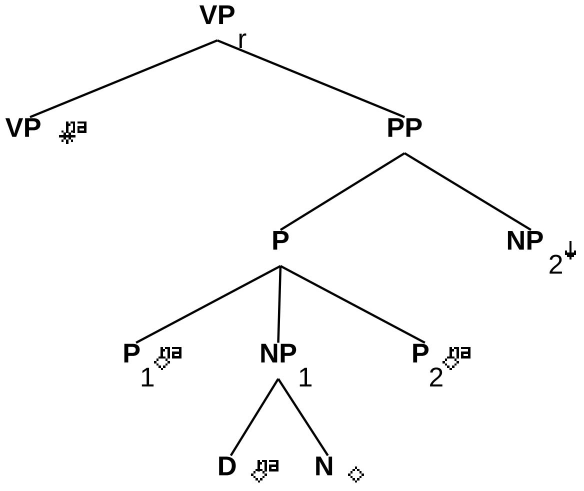
```
PP modifying NP
Three-word P
The noun can take a determiner, and the
 determiner is an anchor of the P.
'Walking for the sake of your health
 is always a good idea'
```

23.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP_2.t:<case>
NP_2:<wh> = -
NP_f.b:<case> = acc/nom

NP_1.b:<agr> = N.t:<agr>
NP_1.b:<case> = N.t:<case>
NP_1.b:<conj> = N.t:<conj>
NP_1.b:<const> = D.t:<const>
NP_1.b:<definite> = D.t:<definite>
NP_1.b:<quan> = D.t:<quan>
NP_1.b:<card> = D.t:<card>
NP_1.b:<gen> = D.t:<gen>
NP_1.b:<decreas> = D.t:<decreas>
NP_1.b:<wh> = D.t:<wh>
NP_1.t:<case> = nom/acc
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<gerund> = NP_r.b:<gerund>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

24 Tree ”betavxPDNPnx”

24.1 graphe



24.2 comments

PP modifying VP

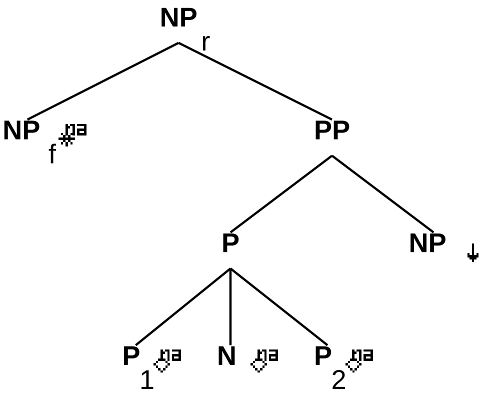
```
Three-word P
The noun can take a determiner, and the
 determiner is an anchor of the P.
'She laughed in the face of defeat'
```

24.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP_2.t:<case>

NP_1.b:<agr> = N.t:<agr>
NP_1.b:<case> = N.t:<case>
NP_1.b:<conj> = N.t:<conj>
NP_1.b:<const> = D.t:<const>
NP_1.b:<definite> = D.t:<definite>
NP_1.b:<quan> = D.t:<quan>
NP_1.b:<card> = D.t:<card>
NP_1.b:<gen> = D.t:<gen>
NP_1.b:<decreas> = D.t:<decreas>
NP_1.b:<wh> = D.t:<wh>
NP_1.t:<case> = nom/acc
VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

25 Tree "betanxPNaPnx"

25.1 graphe



25.2 comments

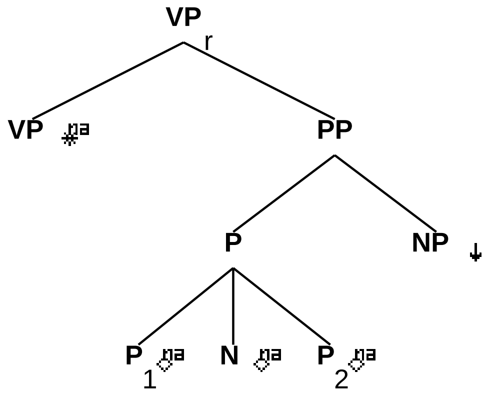
```
PP modifying NP
Three-word P
'The peasant in need of water just
 trapsed across the dais'
```

25.3 features

```
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_r.b:<assign-comp> = NP_f.t:<assign-comp>
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<conj> = NP_f.t:<conj>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>
NP:<wh> = -
NP_f.b:<case> = acc/nom

NP_r.b:<card> = NP_f.t:<card>
NP_r.b:<const> = NP_f.t:<const>
NP_r.b:<quan> = NP_f.t:<quan>
NP_r.b:<decreas> = NP_f.t:<decreas>
NP_r.b:<definite> = NP_f.t:<definite>
NP_r.b:<gen> = NP_f.t:<gen>
NP_f.t:<rel-clause> = NP_r.b:<rel-clause>
NP_f.t:<gerund> = NP_r.b:<gerund>
NP_f.t:<compar> = NP_r.b:<compar>
NP_f.t:<equiv> = NP_r.b:<equiv>
```

26 Tree "betavxPNaPnx"

26.1 graphe



26.2 comments

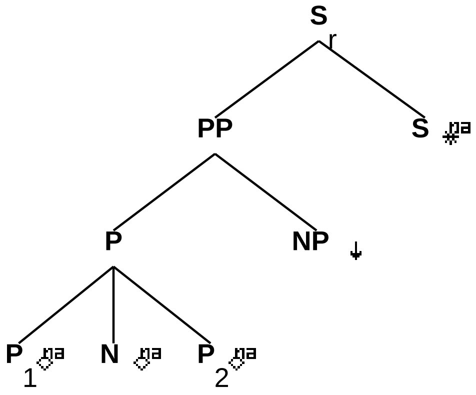
```
PP modifying VP
Three-word P
'She succeeded merely by means of her wit'
```

26.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP.t:<mainv>
VP.t:<mainv> = +
```

27 Tree "betaPNaPnxs"

27.1 graphe



27.2 comments

```
PP modifying S
Three-word P
'On behalf of all of us here tonight,
 we would like to thank you for you
 never-ending generosity'
```

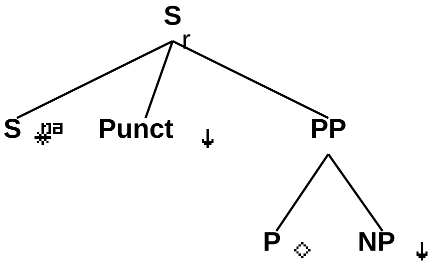
27.3 features

```
S.t:<comp> = nil
```

```
S.t:<comp> = S_r.b:<comp>
S.t:<conj> = S_r.b:<conj>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

28 Tree "betaspuPnx"

28.1 graphe



28.2 comments

```
Post-clausal, punctuation separated PP. These can often been
interpreted as post-posed from somewhere inside the clause. However,
unlike extraposed arguments, it is very difficult to decide the base
position, so a multi-component analysis such as one might use for NP
extraposition seems undesirable for this construction.

John has been more and more exhausted, in recent months.

It increases employee commitment to the company, with
all that means for efficiency and quality control.
```

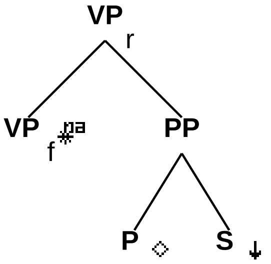
28.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
```

```
S.t:<conj> = S_r.b:<conj>
S.t:<extracted> = S_r.b:<extracted>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<wh> = S_r.b:<wh>
S.t:<inv> = S_r.b:<inv>
S.t:<invlink> = S_r.b:<invlink>
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

Punct.t:<punct struct> = comma/dash
P.t:<assign-case> = PP.b:<assign-case>
PP.b:<assign-case> = NP.t:<case>
S.b:<punct struct> = nil
S.b:<punct term> = nil
```

29 Tree "betavxPs"

29.1 graphe



29.2 comments

```
Tree for sentential adjuncts following main clause:
      The emu left the zoo because she needed more space.

This tree adjoins to the main clause, and the subordinate/adjunct
clause is a substitution site. In general, adjuncts may be indicative,
subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction; the subordinating conjunction may be overt
or null. Sentential adjuncts may not be inverted, extracted or wh+ in
the current analysis, although we may extend the analysis to include
inverted conditional clauses.
```

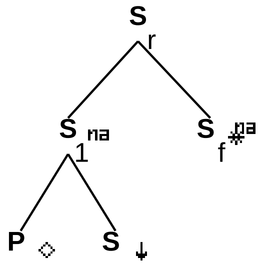
29.3 features

```
S.t:<inv> = -
```

```
S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil

VP_r.b:<conj> = VP_f:<conj>
VP_f:<mode> = ind/imp
VP_r.b:<mode> = VP_f:<mode>
VP_f:<mode> = ind/imp
VP_r.b:<assign-comp> = VP_f:<assign-comp>
VP_r.b:<agr> = VP_f:<agr>
VP_r.b:<tense> = VP_f:<tense>
VP_r.b:<assign-case> = VP_f:<assign-case>

VP_r.b:<passive> = VP_f:<passive>
P.t:<compar> = VP_f.t:<compar>
P.t:<equiv> = VP_f.t:<equiv>
VP_r.b:<compar> = -
VP_r.b:<mainv> = VP_f.t:<mainv>
VP.t:<mainv> = +
```

30 Tree "betaPss"

30.1 graphe



30.2 comments

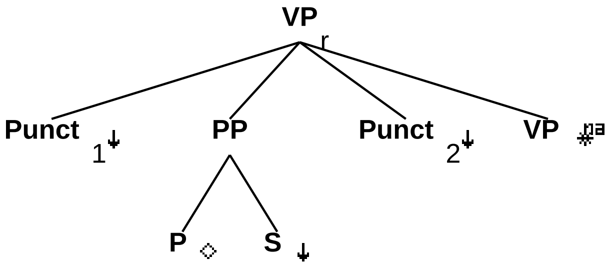
```
Tree for sentential adjuncts preceding the main clause:
      Since she was exhausted Mary went home.

This tree adjoins to the main clause, and the subordinate/adjunct
clause is a substitution site. Separating punctuation is optional,
adjoining using the PUs tree. In general, adjuncts may be indicative,
subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction; the subordinating conjunction may be overt
or null. Sentential adjuncts may not be inverted, extracted or wh+ in
the current analysis, although we may extend the analysis to include
inverted conditional clauses.
```

30.3 features

```
S.t:<wh> = -
S.t:<extracted> = -
S.t:<inv> = -

S.t:<punct struct> = nil

S_r.b:<comp> = S_f:<comp>
S_r.b:<conj> = S_f:<conj>
S_r.b:<assign-comp> = S_f:<assign-comp>
S_r.b:<tense> = S_f:<tense>
S_r.b:<mode> = S_f:<mode>
S_f:<mode> = ind/imp
S_r.b:<assign-case> = S_f:<assign-case>
S_r.b:<agr> = S_f:<agr>

S_r.b:<wh> = S_f:<wh>
S_r.b:<comp> = nil
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

31 Tree "betapuPpuvx"

31.1 graphe



31.2 comments

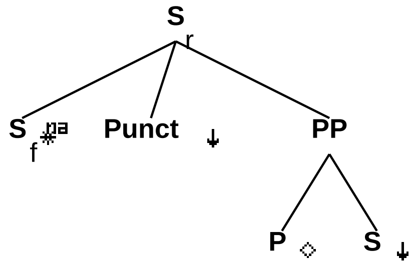
```
Tree for sentential adjuncts occurring between the subject and VP of
the matrix clause:
   Mary, as she was exhasted, took a nap.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. The adjunct clause must be separated from the main
clause by a pair of commas or dashes. In general, adjuncts may be
indicative, subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction; the subordinating conjunction may be overt
or null. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.
```

31.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
Punct_1.t:<punct struct> = Punct_2.t:<punct struct>
Punct_1.t:<punct struct> = comma/dash
VP_r.b:<punct struct> = Punct_1.t:<punct struct>
VP.t:<punct struct> = nil
VP.t:<punct bal> = nil
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<passive> = VP.t:<passive>
VP_r.b:<assign-case> = VP.t:<assign-case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
```

32 Tree ”betaspuPs”

32.1 graphe



32.2 comments

```
Tree for sentential adjuncts following main clause, separated by a
punctuation mark:
   The emu left the zoo, because she needed more space.

We typically attach post-clausal modifiers at the VP node, as you
typically get scope ambiguity effects with negation (’John didn’t
go because he was tired’ - did he go or not?). However, with
post-sentential, comma-separated adjuct clauses there is no ambiguity
- in ’John didn’t go, because he was tired’ he definitely did not
go. Thus, we have a separate tree for these adjunct, which requires
the punctuation mark.
```

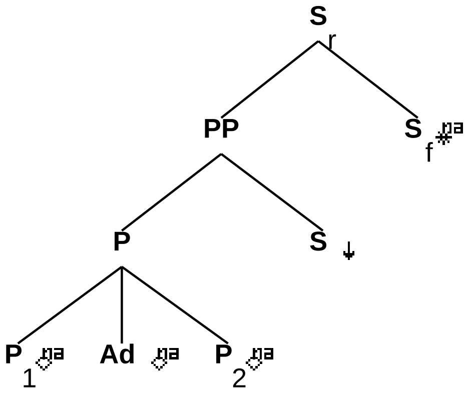
This tree adjoins to the main clause, and the subordinate/adjunct clause is a substitution site. The adjunct clause must be separated from the main clause by a comma or dash. In general, adjuncts may be indicative, subjunctive or participial - this and whether they allow a complementizer are specified in the lexical entry for the subordinating conjunction; the subordinating conjunction may be overt or null. Sentential adjuncts may not be inverted, extracted or wh+ in the current analysis, although we may extend the analysis to include inverted conditional clauses.

32.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil

S_f.t:<punct struct> = nil
S_f.t:<comp> = nil
S_f.t:<comp> = S_r.b:<comp>
S_f.t:<conj> = S_r.b:<conj>
S_f.t:<extracted> = S_r.b:<extracted>
S_f.t:<assign-comp> = S_r.b:<assign-comp>
S_f.t:<tense> = S_r.b:<tense>
S_f.t:<wh> = S_r.b:<wh>
S_f.t:<inv> = S_r.b:<inv>
S_f.t:<invlink> = S_r.b:<invlink>
S_f.t:<mode> = ind/imp
S_f.t:<mode> = S_r.b:<mode>
S_f.t:<assign-case> = S_r.b:<assign-case>
S_f.t:<agr> = S_r.b:<agr>

Punct.t:<punct struct> = comma/dash
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

33 Tree "betaPARBPss"

33.1 graphe



33.2 comments

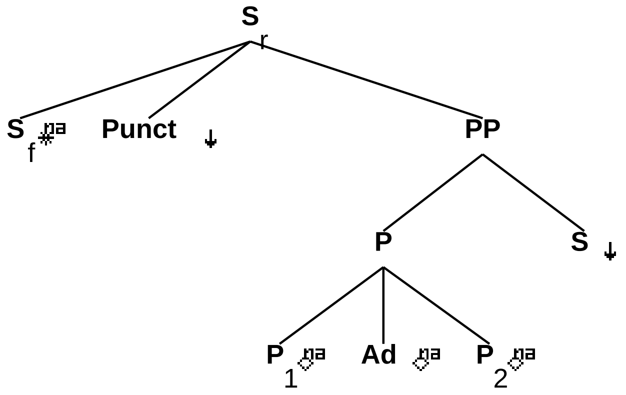
```
Tree for untensed sentential adjuncts introduced by 'as soon as'
preceding the main clause:
      As soon as elected to office, Bill began to make changes.

A tensed adjunct with 'as soon as' would use the CONJarbCONJ set of
subordinating conjunction trees.

This tree adjoins to the main clause, and the subordinate/adjunct
clause is a substitution site. Separating punctuation is optional,
adjoining using the PUs tree. In general, adjuncts may be indicative,
subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.
```

33.3 features

```
S.t:<wh> = -
S.t:<extracted> = -
S.t:<inv> = -

S.t:<punct struct> = nil

S_r.b:<comp> = S_f:<comp>
S_r.b:<conj> = S_f:<conj>
S_r.b:<assign-comp> = S_f:<assign-comp>
S_r.b:<tense> = S_f:<tense>
S_r.b:<mode> = S_f:<mode>
S_f:<mode> = ind/imp
S_r.b:<assign-case> = S_f:<assign-case>
S_r.b:<agr> = S_f:<agr>

S_r.b:<wh> = S_f:<wh>
S_r.b:<comp> = nil
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

34 Tree "betaspuPARBPs"

34.1 graphe



34.2 comments

Tree for untensed sentential adjuncts introduced by 'as soon as'
following main clause and separated by a punctuation mark:
      Bill began to make changes, as soon as elected to office.

A tensed adjunct with 'as soon as' would use the CONJarbCONJ set of
subordinating conjunction trees.

We typically attach post-clausal modifiers at the VP node, as you
typically get scope ambiguity effects with negation ('John didn't
go because he was tired' - did he go or not?). However, with
post-sentential, comma-separated adjuct clauses there is no ambiguity
- in 'John didn't go, because he was tired' he definitely did not
go. Thus, we have a separate tree for these adjunct, which requires
the punctuation mark.

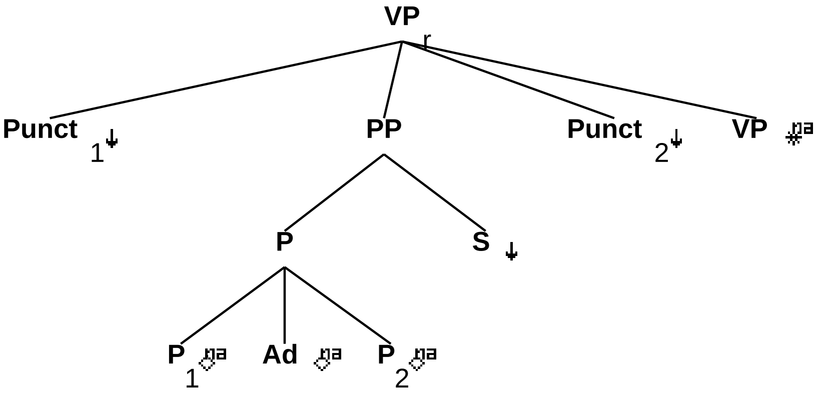
This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. The adjunct clause must be separated from the main
clause by a comma or dash. In general, adjuncts may be indicative,
subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.

34.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil

S_f.t:<punct struct> = nil
S_f.t:<comp> = nil
S_f.t:<comp> = S_r.b:<comp>
S_f.t:<conj> = S_r.b:<conj>
S_f.t:<extracted> = S_r.b:<extracted>
S_f.t:<assign-comp> = S_r.b:<assign-comp>
S_f.t:<tense> = S_r.b:<tense>
S_f.t:<wh> = S_r.b:<wh>
S_f.t:<inv> = S_r.b:<inv>
S_f.t:<invlink> = S_r.b:<invlink>
S_f.t:<mode> = ind/imp
S_f.t:<mode> = S_r.b:<mode>
S_f.t:<assign-case> = S_r.b:<assign-case>
S_f.t:<agr> = S_r.b:<agr>

Punct.t:<punct struct> = comma/dash
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

35 Tree "betapuPARBPpuvx"

35.1 graphe



35.2 comments

```
Tree for untensed sentential adjuncts introduced by 'as soon as'
occurring between the subject and VP of the matrix clause:
      Bill, as soon as elected to office,  began to make changes.
```

A tensed adjunct with 'as soon as' would use the CONJarbCONJ set of subordinating conjunction trees.

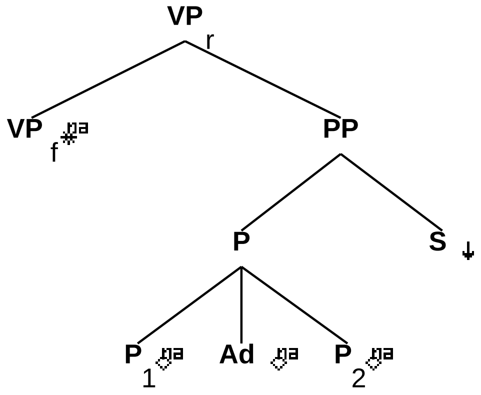
This tree adjoins to the main clause, and the subordinate/adjunct clause is a substitution site. The adjunct clause must be separated from the main clause by a pair of commas or dashes. In general, adjuncts may be indicative, subjunctive or participial - this and whether they allow a complementizer are specified in the lexical entry for the subordinating conjunction. Sentential adjuncts may not be inverted, extracted or wh+ in the current analysis, although we may extend the analysis to include inverted conditional clauses.

35.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
Punct_1.t:<punct struct> = Punct_2.t:<punct struct>
Punct_1.t:<punct struct> = comma/dash
VP_r.b:<punct struct> = Punct_1.t:<punct struct>
VP.t:<punct struct> = nil
VP.t:<punct bal> = nil
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<passive> = VP.t:<passive>
VP_r.b:<assign-case> = VP.t:<assign-case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
```

36 Tree "betavxPARBPs"

36.1 graphe



36.2 comments

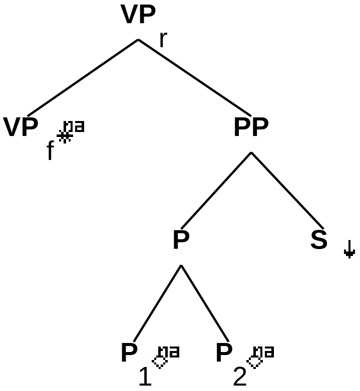
```
Tree for untensed sentential adjuncts introduced by 'as soon as'
following main clause:
      Bill began to make changes as soon as elected to office.

A tensed adjunct with 'as soon as' would use the CONJarbCONJ set of
subordinating conjunction trees.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. In general, adjuncts may be indicative, subjunctive
or participial - this and whether they allow a complementizer are
specified in the lexical entry for the subordinating
conjunction. Sentential adjuncts may not be inverted, extracted or wh+
in the current analysis, although we may extend the analysis to
include inverted conditional clauses.
```

36.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
VP_r.b:<conj> = VP_f:<conj>
VP_f:<mode> = ind/imp
VP_r.b:<mode> = VP_f:<mode>
VP_f:<mode> = ind/imp
VP_r.b:<assign-comp> = VP_f:<assign-comp>
VP_r.b:<agr> = VP_f:<agr>
VP_r.b:<tense> = VP_f:<tense>
VP_r.b:<assign-case> = VP_f:<assign-case>
VP_r.b:<passive> = VP_f:<passive>
VP_f.t:<compar> = -
VP_r.b:<compar> = VP_f.t:<compar>
VP_r.b:<mainv> = VP_f.t:<mainv>
VP.t:<mainv> = +
```

37 Tree "betavxPPs"

37.1 graphe



37.2 comments

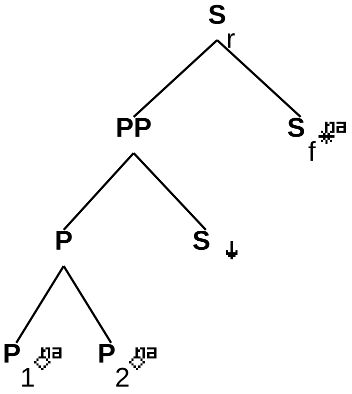
```
Tree for a multi-word subordinating conjunction introducing a
sentential adjunct which follows the main clause:
      Mary will move to LA even if she cannot find a job.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. In general, adjuncts may be indicative, subjunctive
or participial - this and whether they allow a complementizer are
specified in the lexical entry for the subordinating
conjunction. Sentential adjuncts may not be inverted, extracted or wh+
in the current analysis, although we may extend the analysis to
include inverted conditional clauses.
```

37.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
VP_r.b:<conj> = VP_f:<conj>
VP_f:<mode> = ind/imp
VP_r.b:<mode> = VP_f:<mode>
VP_f:<mode> = ind/imp
VP_r.b:<assign-comp> = VP_f:<assign-comp>
VP_r.b:<agr> = VP_f:<agr>
VP_r.b:<tense> = VP_f:<tense>
VP_r.b:<assign-case> = VP_f:<assign-case>

VP_r.b:<passive> = VP_f:<passive>
VP_f.t:<compar> = -
VP_r.b:<compar> = VP_f.t:<compar>
VP_r.b:<mainv> = VP_f.t:<mainv>
VP.t:<mainv> = +
```

38 Tree "betaPPss"

38.1 graphe



38.2 comments

Tree for a multi-word subordinating conjunction introducing a sentential adjunct which precedes the main clause:

Even if she cannot find a job Mary will move to LA.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a substitution site. Separating punctuation is optional, adjoining using the PUs tree. In general, adjuncts may be indicative, subjunctive or participial - this and whether they allow a complementizer are specified in the lexical entry for the subordinating conjunction. Sentential adjuncts may not be inverted, extracted or wh+ in the current analysis, although we may extend the analysis to include inverted conditional clauses.

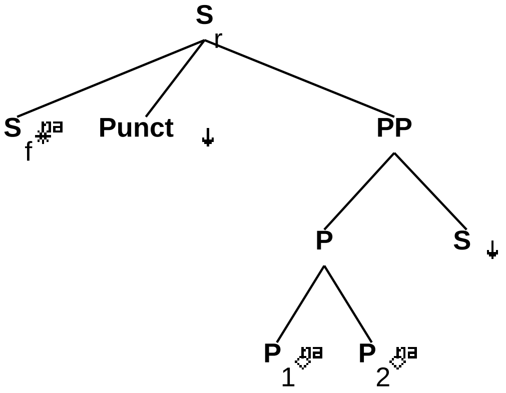
38.3 features

```
S.t:<wh> = -
S.t:<extracted> = -
S.t:<inv> = -

S.t:<punct struct> = nil
```

```
S_r.b:<comp> = S_f:<comp>
S_r.b:<conj> = S_f:<conj>
S_r.b:<assign-comp> = S_f:<assign-comp>
S_r.b:<tense> = S_f:<tense>
S_r.b:<mode> = S_f:<mode>
S_f:<mode> = ind/imp
S_r.b:<assign-case> = S_f:<assign-case>
S_r.b:<agr> = S_f:<agr>

S_r.b:<wh> = S_f:<wh>
S_r.b:<comp> = nil
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

39 Tree "betaspuPPs"

39.1 graphe



39.2 comments

```
Tree for multi-word subordinating conjunctions, introducing a
sentential adjunct which follows the main clause and is separated by a
punctuation mark:
   Mary will move to LA, even if she does not find a job.

We typically attach post-clausal modifiers at the VP node, as you
typically get scope ambiguity effects with negation ('John didn't
go because he was tired' - did he go or not?). However, with
post-sentential, comma-separated adjuct clauses there is no ambiguity
- in 'it John didn't go, because he was tired' he definitely did not
go. Thus, we have a separate tree for these adjunct, which requires
the punctuation mark.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. The adjunct clause must be separated from the main
clause by a comma or dash. In general, adjuncts may be indicative,
```

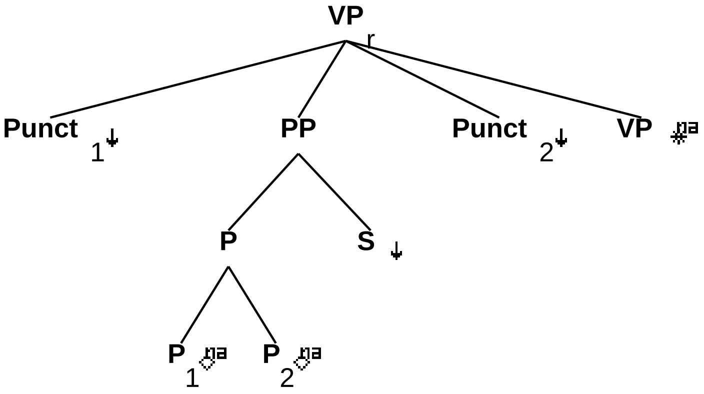
subjunctive or participial - this and whether they allow a complementizer are specified in the lexical entry for the subordinating conjunction. Sentential adjuncts may not be inverted, extracted or wh+ in the current analysis, although we may extend the analysis to include inverted conditional clauses.

39.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil

S_f.t:<punct struct> = nil
S_f.t:<comp> = nil
S_f.t:<comp> = S_r.b:<comp>
S_f.t:<conj> = S_r.b:<conj>
S_f.t:<extracted> = S_r.b:<extracted>
S_f.t:<assign-comp> = S_r.b:<assign-comp>
S_f.t:<tense> = S_r.b:<tense>
S_f.t:<wh> = S_r.b:<wh>
S_f.t:<inv> = S_r.b:<inv>
S_f.t:<invlink> = S_r.b:<invlink>
S_f.t:<mode> = ind/imp
S_f.t:<mode> = S_r.b:<mode>
S_f.t:<assign-case> = S_r.b:<assign-case>
S_f.t:<agr> = S_r.b:<agr>

Punct.t:<punct struct> = comma/dash
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

40 Tree "betapuPPpuvx"

40.1 graphe



40.2 comments

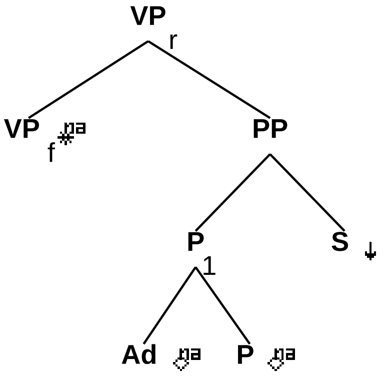
```
Tree for multi-word subordinating conjunctions, introducing a
sentential adjunct which occurs between the subject and VP of
the matrix clause:
   Mary, even if she does not find a job, will move to LA.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. The adjunct clause must be separated from the main
clause by a pair of commas or dashes. In general, adjuncts may be
indicative, subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.
```

40.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
Punct_1.t:<punct struct> = Punct_2.t:<punct struct>
Punct_1.t:<punct struct> = comma/dash
VP_r.b:<punct struct> = Punct_1.t:<punct struct>
VP.t:<punct struct> = nil
VP.t:<punct bal> = nil
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<passive> = VP.t:<passive>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
```

41 Tree "betavxARBPs"

41.1 graphe



41.2 comments

Tree for a multi-word subordinating conjunction introducing a sentential adjunct which follows the main clause:

    Mary will move to LA even if she cannot find a job.

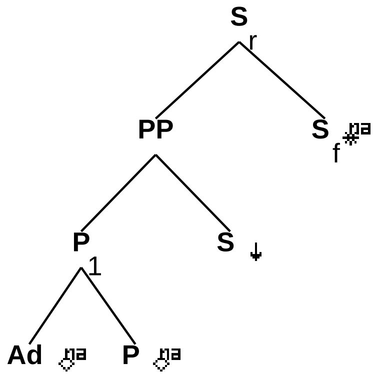
This tree adjoins to the main clause, and the subordinate/adjunct clause is a substitution site. In general, adjuncts may be indicative, subjunctive or participial - this and whether they allow a complementizer are specified in the lexical entry for the subordinating conjunction. Sentential adjuncts may not be inverted, extracted or wh+ in the current analysis, although we may extend the analysis to include inverted conditional clauses.

41.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
VP_r.b:<conj> = VP_f:<conj>
VP_f:<mode> = ind/imp
VP_r.b:<mode> = VP_f:<mode>
VP_f:<mode> = ind/imp
VP_r.b:<assign-comp> = VP_f:<assign-comp>
VP_r.b:<agr> = VP_f:<agr>
VP_r.b:<tense> = VP_f:<tense>
VP_r.b:<assign-case> = VP_f:<assign-case>

VP_r.b:<passive> = VP_f:<passive>
VP_f.t:<compar> = -
VP_r.b:<compar> = VP_f.t:<compar>
VP_r.b:<mainv> = VP_f.t:<mainv>
VP.t:<mainv> = +
```

42 Tree "betaARBPss"

42.1 graphe



42.2 comments

```
Tree for a multi-word subordinating conjunction introducing a
sentential adjunct which precedes the main clause:
      Even if she cannot find a job Mary will move to LA.

This tree adjoins to the main clause, and the subordinate/adjunct
clause is a substitution site. Separating punctuation is optional,
adjoining using the PUs tree. In general, adjuncts may be indicative,
subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.
```

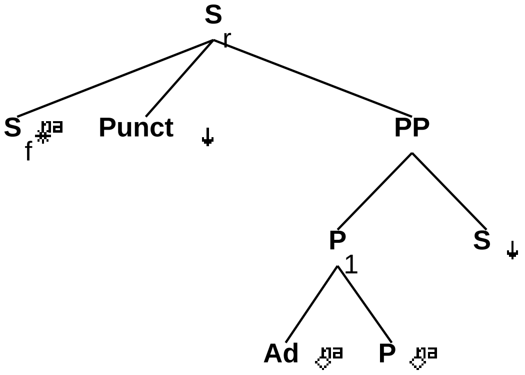
42.3 features

```
S.t:<wh> = -
S.t:<extracted> = -
S.t:<inv> = -

S.t:<punct struct> = nil

S_r.b:<comp> = S_f:<comp>
S_r.b:<conj> = S_f:<conj>
S_r.b:<assign-comp> = S_f:<assign-comp>
S_r.b:<tense> = S_f:<tense>
S_r.b:<mode> = S_f:<mode>
S_f:<mode> = ind/imp
```

```
S_r.b:<assign-case> = S_f:<assign-case>
S_r.b:<agr> = S_f:<agr>

S_r.b:<wh> = S_f:<wh>
S_r.b:<comp> = nil
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

43 Tree ”betaspuARBPs”

43.1 graphe



43.2 comments

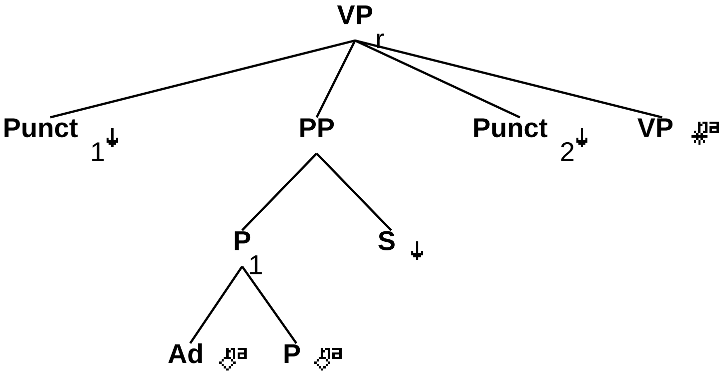
```
Tree for multi-word subordinating conjunctions, introducing a
sentential adjunct which follows the main clause and is separated by a
punctuation mark:
   Mary will move to LA, even if she does not find a job.

We typically attach post-clausal modifiers at the VP node, as you
typically get scope ambiguity effects with negation ('John didn't
go because he was tired' - did he go or not?). However, with
post-sentential, comma-separated adjuct clauses there is no ambiguity
- in 'it John didn't go, because he was tired' he definitely did not
go. Thus, we have a separate tree for these adjunct, which requires
the punctuation mark.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. The adjunct clause must be separated from the main
clause by a comma or dash. In general, adjuncts may be indicative,
subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.
```

43.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil

S_f.t:<punct struct> = nil
S_f.t:<comp> = nil
S_f.t:<comp> = S_r.b:<comp>
S_f.t:<conj> = S_r.b:<conj>
S_f.t:<extracted> = S_r.b:<extracted>
S_f.t:<assign-comp> = S_r.b:<assign-comp>
S_f.t:<tense> = S_r.b:<tense>
S_f.t:<wh> = S_r.b:<wh>
S_f.t:<inv> = S_r.b:<inv>
S_f.t:<invlink> = S_r.b:<invlink>
S_f.t:<mode> = ind/imp
S_f.t:<mode> = S_r.b:<mode>
S_f.t:<assign-case> = S_r.b:<assign-case>
S_f.t:<agr> = S_r.b:<agr>

Punct.t:<punct struct> = comma/dash
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

44 Tree "betapuARBPpuvx"

44.1 graphe



44.2 comments

Tree for multi-word subordinating conjunctions, introducing a sentential adjunct which occurs between the subject and VP of the matrix clause:

    Mary, even if she does not find a job, will move to LA.

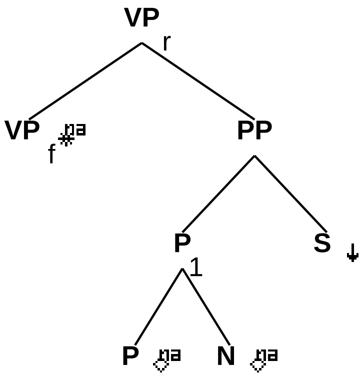
```
This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. The adjunct clause must be separated from the main
clause by a pair of commas or dashes. In general, adjuncts may be
indicative, subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.
```

44.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
Punct_1.t:<punct struct> = Punct_2.t:<punct struct>
Punct_1.t:<punct struct> = comma/dash
VP_r.b:<punct struct> = Punct_1.t:<punct struct>
VP.t:<punct struct> = nil
VP.t:<punct bal> = nil
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<passive> = VP.t:<passive>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
```

45 Tree "betavxPNs"

45.1 graphe



45.2 comments

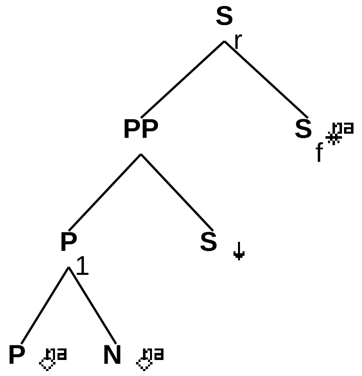
```
Tree for a multi-word subordinating conjunction introducing a
sentential adjunct which follows the main clause:
      Mary bought a spare bike in case her old one is stolen.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a
substitution site. In general, adjuncts may be indicative, subjunctive
or participial - this and whether they allow a complementizer are
specified in the lexical entry for the subordinating
conjunction. Sentential adjuncts may not be inverted, extracted or wh+
in the current analysis, although we may extend the analysis to
include inverted conditional clauses.
```

45.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
VP_r.b:<conj> = VP_f:<conj>
VP_f:<mode> = ind/imp
VP_r.b:<mode> = VP_f:<mode>
VP_f:<mode> = ind/imp
VP_r.b:<assign-comp> = VP_f:<assign-comp>
VP_r.b:<agr> = VP_f:<agr>
VP_r.b:<tense> = VP_f:<tense>
VP_r.b:<assign-case> = VP_f:<assign-case>

VP_r.b:<passive> = VP_f:<passive>
VP_f.t:<compar> = -
VP_r.b:<compar> = VP_f.t:<compar>
VP_r.b:<mainv> = VP_f.t:<mainv>
VP.t:<mainv> = +
```

46 Tree "betaPNss"

46.1 graphe



46.2 comments

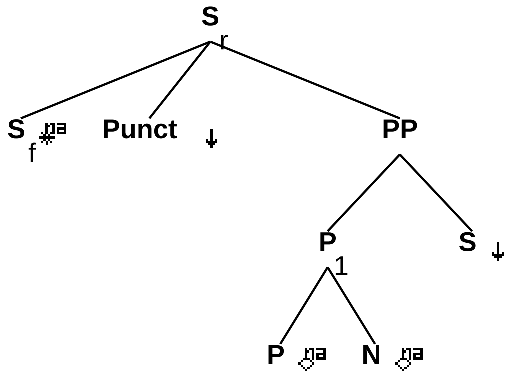
```
Tree for a multi-word subordinating conjunction introducing a
sentential adjunct which precedes the main clause:
      For all that Mary enjoys sports, she cannot abide golf.

This tree adjoins to the main clause, and the subordinate/adjunct
clause is a substitution site. Separating punctuation is optional,
adjoining using the PUs tree. In general, adjuncts may be indicative,
subjunctive or participial - this and whether they allow a
complementizer are specified in the lexical entry for the
subordinating conjunction. Sentential adjuncts may not be inverted,
extracted or wh+ in the current analysis, although we may extend the
analysis to include inverted conditional clauses.
```

46.3 features

```
S.t:<wh> = -
S.t:<extracted> = -
S.t:<inv> = -

S.t:<punct struct> = nil

S_r.b:<comp> = S_f:<comp>
S_r.b:<conj> = S_f:<conj>
S_r.b:<assign-comp> = S_f:<assign-comp>
S_r.b:<tense> = S_f:<tense>
S_r.b:<mode> = S_f:<mode>
S_f:<mode> = ind/imp
S_r.b:<assign-case> = S_f:<assign-case>
S_r.b:<agr> = S_f:<agr>

S_r.b:<wh> = S_f:<wh>
S_r.b:<comp> = nil
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

47 Tree "betaspuPNs"

47.1 graphe



47.2 comments

Tree for multi-word subordinating conjunctions, introducing a sentential adjunct which follows the main clause and is separated by a punctuation mark:

Mary bought a spare bike, in case her old one is stolen.

We typically attach post-clausal modifiers at the VP node, as you typically get scope ambiguity effects with negation ('John didn't go because he was tired' - did he go or not?). However, with post-sentential, comma-separated adjuct clauses there is no ambiguity - in 'it John didn't go, because he was tired' he definitely did not go. Thus, we have a separate tree for these adjunct, which requires the punctuation mark.

This tree adjoins to the main clause, and the subordinate/adjunct clause is a substitution site. The adjunct clause must be separated from the main clause by a comma or dash. In general, adjuncts may be indicative, subjunctive or participipal - this and whether they allow a complementizer are specified in the lexical entry for the subordinating conjunction. Sentential adjuncts may not be inverted, extracted or wh+ in the current analysis, although we may extend the analysis to include inverted conditional clauses.

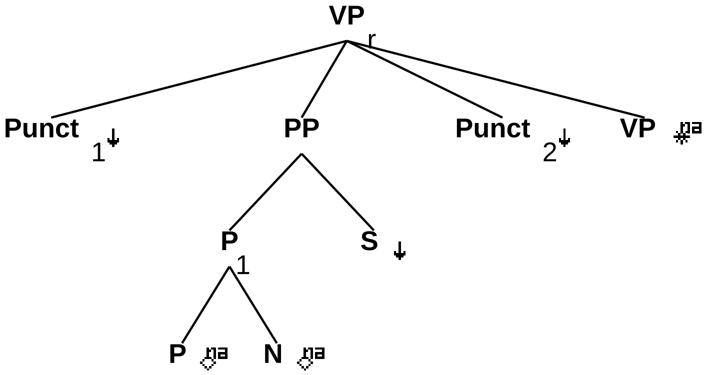
47.3 features

```
S.t:<inv> = -

S.t:<wh> = -
```

```
S.t:<extracted> = -
S.t:<punct struct> = nil

S_f.t:<punct struct> = nil
S_f.t:<comp> = nil
S_f.t:<comp> = S_r.b:<comp>
S_f.t:<conj> = S_r.b:<conj>
S_f.t:<assign-comp> = S_r.b:<assign-comp>
S_f.t:<tense> = S_r.b:<tense>
S_f.t:<wh> = S_r.b:<wh>
S_f.t:<inv> = S_r.b:<inv>
S_f.t:<invlink> = S_r.b:<invlink>
S_f.t:<mode> = ind/imp
S_f.t:<mode> = S_r.b:<mode>
S_f.t:<assign-case> = S_r.b:<assign-case>
S_f.t:<agr> = S_r.b:<agr>

Punct.t:<punct struct> = comma/dash
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```

48 Tree ”betapuPNpuvx”

48.1 graphe



48.2 comments

```
Tree for multi-word subordinating conjunctions, introducing a
sentential adjunct which occurs between the subject and VP of
the matrix clause:
   Mary, in case her old bike is stolen, bought a spare one.

This tree adjoins to the main clause, and the subordinate/adjunct
clause is a substitution site. The adjunct clause must be separated
from the main clause by a pair of commas or dashes. In general,
adjuncts may be indicative, subjunctive or participial - this and
whether they allow a complementizer are specified in the lexical entry
```

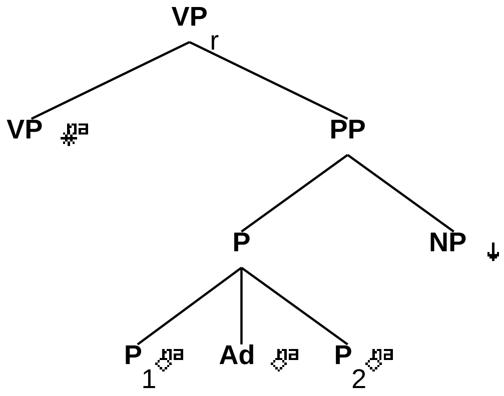
for the subordinating conjunction. Sentential adjuncts may not be inverted, extracted or wh+ in the current analysis, although we may extend the analysis to include inverted conditional clauses.

48.3 features

```
S.t:<inv> = -

S.t:<wh> = -
S.t:<extracted> = -
S.t:<punct struct> = nil
Punct_1.t:<punct struct> = Punct_2.t:<punct struct>
Punct_1.t:<punct struct> = comma/dash
VP_r.b:<punct struct> = Punct_1.t:<punct struct>
VP.t:<punct struct> = nil
VP.t:<punct bal> = nil
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<passive> = VP.t:<passive>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
```

49 Tree ”betavxPARBPnx”

49.1 graphe



49.2 comments

PP modifying VP
Three-word P, as well/recently as

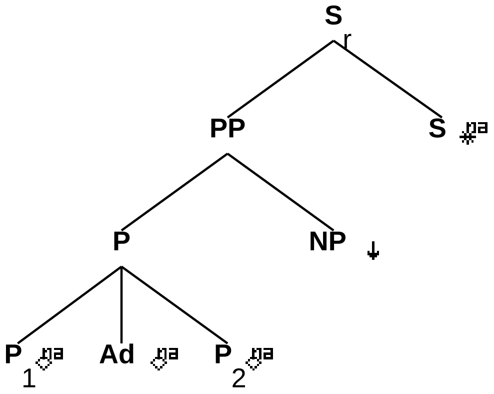
She went to the doctor as recently as yesterday.

49.3 features

```
VP_r.b:<mode> = VP.t:<mode>
VP_r.b:<agr> = VP.t:<agr>
VP_r.b:<tense> = VP.t:<tense>
VP_r.b:<assign-case> = VP.t:<assign-case>
VP_r.b:<assign-comp> = VP.t:<assign-comp>
VP_r.b:<passive> = VP.t:<passive>
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP.t:<case>

VP.t:<compar> = -
VP_r.b:<compar> = VP.t:<compar>
VP_r.b:<mainv> = VP_f.t:<mainv>
VP.t:<mainv> = +
```

50 Tree "betaPARBPnxs"

50.1 graphe



50.2 comments

```
PP modifying S
Three-word P, as well/recently as

'As well as the 1.2 million dollars in taxes, we will be gaining a good portion
of our budget from ticket fees.'
```

50.3 features

```
S.t:<comp> = nil
S.t:<comp> = S_r.b:<comp>
S.t:<conj> = S_r.b:<conj>
S.t:<assign-comp> = S_r.b:<assign-comp>
S.t:<tense> = S_r.b:<tense>
S.t:<extracted> = S_r.b:<extracted>
```

```
S.t:<mode> = S_r.b:<mode>
S.t:<assign-case> = S_r.b:<assign-case>
S.t:<agr> = S_r.b:<agr>

S_r.b:<wh> = PP.t:<wh>
PP.b:<wh> = NP:<wh>
S_r.b:<inv> = S.t:<inv>
S_r.b:<invlink> = S_r.b:<inv>
S.b:<comp> = nil
PP.b:<assign-case> = P.t:<assign-case>
PP.b:<assign-case> = NP:<case>
S_r.b:<nocomp-mode> = S.t:<nocomp-mode>
```