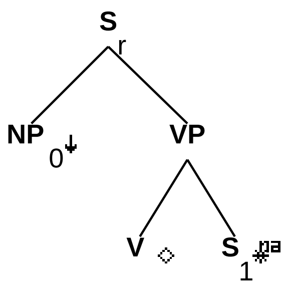
Family ”TXnx0Vs1”

March 5, 2008

1 Tree ”betaXnx0Vs1”

1.1 graphe



1.2 comments

```
ECM verbs:
John expects Bill to eat beans
John sees Bill eat beans

Parallel to nx0Vs1 tree except S foot has assign-case=acc,
comp=nil, and assign-comp=for.  Lexical entries selects
wheter it should take mode inf or base (for bare infinitives)
```

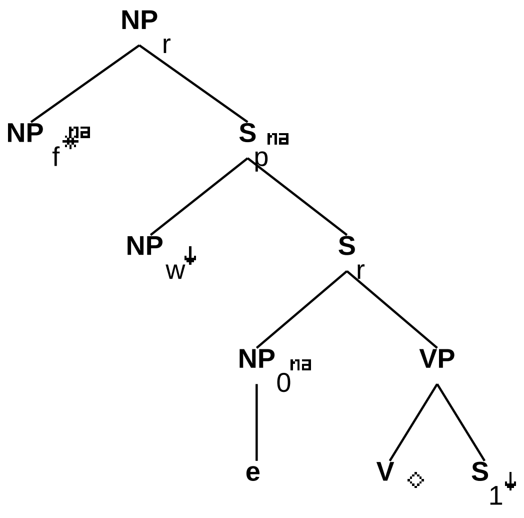
1.3 features

```
S_r.b:<extracted> = -
S_r.b:<wh> = NP_0.t:<wh>



VP.b:<compar> = -
S_r.b:<comp> = nil
S_r.b:<inv> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<assign-comp> = VP.t:<assign-comp>
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_0:<wh> = -
```

```
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
V.t:<contr> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>

S_1.t:<assign-comp> = ecm
S_1.t:<inv> = -
S_1.t:<extracted> = -
S_r.b:<control> = NP_0.t:<control>
S_1.t:<control> = NP_0.t:<control>
S_r.b:<punct contains> = VP.t:<punct contains>
VP.b:<punct contains> = S_1.t:<punct contains>
S_1.t:<comp> = nil
```

2 Tree ”betaXN0nx0Vs1”

2.1 graphe



2.2 comments

```
The ECM parallel of N0nx0Vs1 -

Relative clauses with subject gap
```

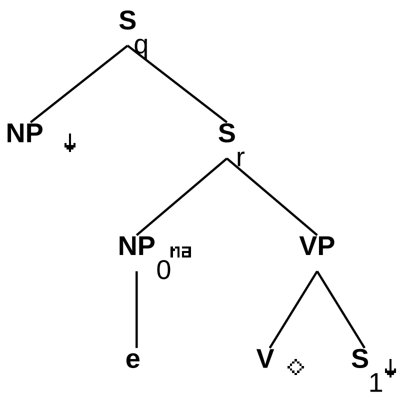
```
The person who expects John to leave
```

2.3 features

```
S_r.b:<assign-comp> = VP.t:<assign-comp>


VP.b:<compar> = -
S_r.b:<mode> = VP.t:<mode>
S_r.t:<mode> = ind/inf
S_r.b:<comp> = nil
S_r.t:<inv> = -
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
S_r.b:<agr> = NP_0.t:<agr>
S_r.b:<assign-case> = NP_0.t:<case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
V.t:<contr> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>
S_1.t:<assign-comp> = ecm

S_1.t:<inv> = -
S_1.t:<extracted> = -
S_1.t:<comp> = nil
S_r.t:<conj> = nil

NP_w.t:<trace> = NP_0.b:<trace>
NP_w.t:<case> = NP_0.b:<case>
NP_w.t:<agr> = NP_0.b:<agr>
NP_w.t:<wh> = +
S_r.t:<comp> = nil
NP_r.b:<rel-clause> = +
NP_f.b:<case> = nom/acc
NP_r.b:<pron> = NP_f.t:<pron>
```

3 Tree "alphaXW0nx0Vs1"

3.1 graphe



3.2 comments

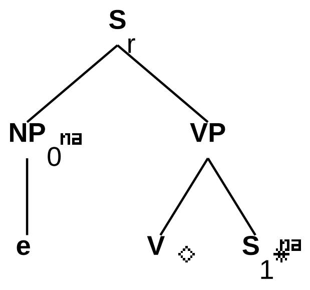
```
ECM Parallel to N0nx0Vs1

Sentential complement verbs with subject extracted:
        Who expects John to eat bean?
```

3.3 features

```
S_q.b:<extracted> = +

S_q.b:<inv> = S_r.t:<inv>
S_q.b:<wh> = S_r.t:<wh>

S_r.t:<comp> = nil
S_r.t:<assign-comp> = inf_nil/ind_nil
S_r.b:<assign-comp> = VP.t:<assign-comp>


VP.b:<compar> = -
S_q.b:<wh> = NP:<wh>
NP_0:<wh> = NP:<wh>
NP:<wh> = +
S_q.b:<mode> = S_r.t:<mode>
S_q.b:<comp> = nil
S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<inv> = -
NP:<trace> = NP_0:<trace>
NP:<agr> = NP_0:<agr>
NP:<case> = NP_0:<case>
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
S_r.b:<agr> = NP_0:<agr>
S_r.b:<assign-case> = NP_0:<case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
V.t:<contr> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>
S_1.t:<assign-comp> = ecm

S_1.t:<inv> = -
S_1.t:<extracted> = -
S_r.t:<conj> = nil
S_1.t:<control> = NP_0:<control>
S_1.t:<comp> = nil
S_r.b:<assign-comp> = inf_nil/ind_nil/ecm
```

4 Tree "betaXInx0Vs1"

4.1 graphe



4.2 comments

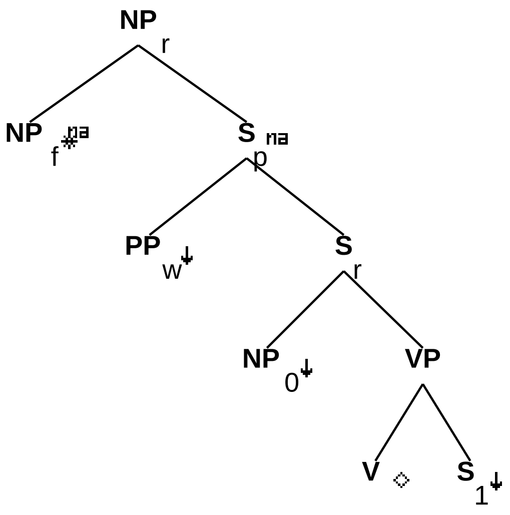
```
ECM PArallel to Inx0Vs1

Imperative tree

Expect Bill to eat beans
See Bill eat beans
```

4.3 features

```
S_r.b:<extracted> = -
S_r.b:<inv> = -
S_r.t:<assign-comp> = inf_nil/ind_nil
S_r.b:<assign-comp> = VP.t:<assign-comp>


S_r.b:<comp> = nil
S_r.b:<mode> = imp
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_0:<wh> = -
NP_0:<agr pers> = 2
NP_0:<agr 3rdsing> = -
NP_0:<agr num> = plur/sing
NP_0:<case> = nom
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
V.t:<contr> = -
VP.t:<tense> = pres
VP.t:<neg> = -
VP.t:<mode> = base
VP.b:<mode> = V.t:<mode>
VP.b:<agr> = V.t:<agr>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>
VP.b:<compar> = -
S_1.t:<assign-comp> = ecm

S_1.t:<inv> = -
S_1.t:<extracted> = -
S_1.t:<control> = NP_0.t:<control>
S_1.t:<comp> = nil
```

5 Tree "betaXNpxnx0Vs1"

5.1 graphe



5.2 comments

```
ECM verbs:
John expects Bill to eat beans
John sees Bill eat beans

Parallel to nx0Vs1 tree except S foot has assign-case=acc,
comp=nil, and assign-comp=for.  Lexical entries selects
wheter it should take mode inf or base (for bare infinitives)
```

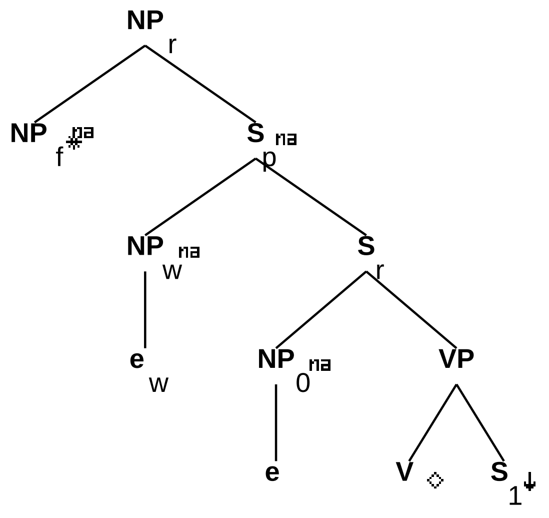
5.3 features

```
S_r.b:<extracted> = -
S_r.b:<wh> = NP_0.t:<wh>



VP.b:<compar> = -
S_r.b:<comp> = nil
S_r.b:<inv> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<assign-comp> = VP.t:<assign-comp>
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_0:<wh> = -
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
VP.b:<passive> = V.t:<passive>
```

```
V.t:<passive> = -
V.t:<contr> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>

S_1.t:<assign-comp> = ecm
S_1.t:<inv> = -
S_1.t:<extracted> = -
S_r.b:<control> = NP_0.t:<control>
S_1.t:<control> = NP_0.t:<control>
S_r.b:<punct contains> = VP.t:<punct contains>
VP.b:<punct contains> = S_1.t:<punct contains>
S_1.t:<comp> = nil
S_r.t:<inv> = -
PP_w.t:<wh> = +
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_f.b:<case> = acc/nom
S_r.t:<comp> = nil
NP_r.b:<rel-clause> = +
NP_f.b:<case> = nom/acc
NP_r.b:<pron> = NP_f.t:<pron>
```

6 Tree "betaXNc0nx0Vs1"

6.1 graphe



6.2 comments

```
The ECM parallel of N0nx0Vs1 -

Relative clauses with subject gap

The person who expects John to leave
```

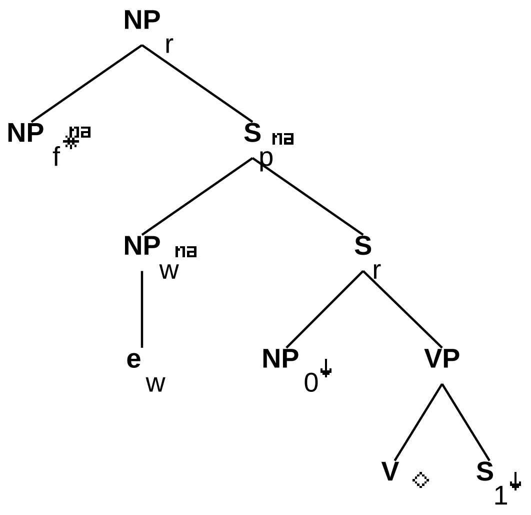
6.3 features

```
S_r.b:<assign-comp> = VP.t:<assign-comp>


VP.b:<compar> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.t:<inv> = -
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
S_r.b:<agr> = NP_0.t:<agr>
S_r.b:<assign-case> = NP_0.t:<case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
V.t:<contr> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>
S_1.t:<assign-comp> = ecm

S_1.t:<inv> = -
S_1.t:<extracted> = -
S_1.t:<comp> = nil
S_r.t:<conj> = nil

NP_w.t:<trace> = NP_0.b:<trace>
NP_w.t:<case> = NP_0.b:<case>
NP_w.t:<agr> = NP_0.b:<agr>
NP_r.b:<rel-clause> = +
S_r.t:<mode> = inf/ger/ind
S_r.t:<nocomp-mode> = inf/ger
VP.t:<assign-comp> = that/ind_nil/inf_nil/ecm
S_r.b:<nocomp-mode> = S_r.b:<mode>
NP_f.b:<case> = nom/acc
```

```
NP_r.b:<pron> = NP_f.t:<pron>
```

7 Tree "betaXNcnx0Vs1"

7.1 graphe



7.2 comments

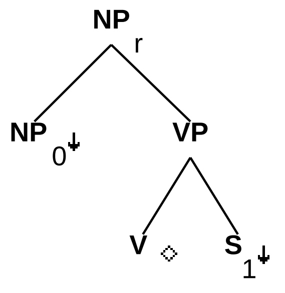
```
ECM verbs:
John expects Bill to eat beans
John sees Bill eat beans

Parallel to nx0Vs1 tree except S foot has assign-case=acc,
comp=nil, and assign-comp=for.  Lexical entries selects
wheter it should take mode inf or base (for bare infinitives)
```

7.3 features

```
S_r.b:<extracted> = -
S_r.b:<wh> = NP_0.t:<wh>



S_r.b:<comp> = nil
S_r.b:<inv> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<assign-comp> = VP.t:<assign-comp>
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_0:<wh> = -
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
VP.b:<passive> = V.t:<passive>
VP.b:<compar> = -
V.t:<passive> = -
V.t:<contr> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>

S_1.t:<assign-comp> = ecm
S_1.t:<inv> = -
S_1.t:<extracted> = -
S_r.b:<control> = NP_0.t:<control>
S_1.t:<control> = NP_0.t:<control>
S_r.b:<punct contains> = VP.t:<punct contains>
VP.b:<punct contains> = S_1.t:<punct contains>
S_1.t:<comp> = nil
NP_r.b:<wh> = NP_f.t:<wh>
NP_r.b:<agr> = NP_f.t:<agr>
NP_r.b:<case> = NP_f.t:<case>
NP_f.b:<case> = acc/nom
S_r.t:<inv> = -
S_r.t:<mode> = ind/inf
S_r.t:<nocomp-mode> = ind
VP.t:<assign-comp> = that/for/ind_nil
S_r.b:<nocomp-mode> = S_r.b:<mode>
NP_r.b:<rel-clause> = +
NP_f.b:<case> = nom/acc
NP_r.b:<pron> = NP_f.t:<pron>
```

8 Tree "alphaXGnx0Vs1"

8.1 graphe



8.2 comments

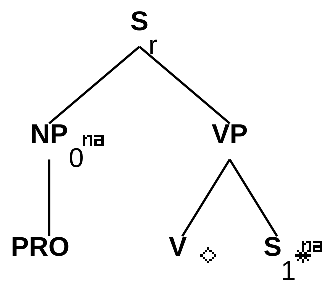
```
ECM Parallel to Gnx0Vs1

Gerund of ECM verbs:

...John('s) expecting Bill to eat beans...
...John('s) seeing Bill eat beans...
```

8.3 features

```
NP_0:<wh> = NP_r.b:<wh>
VP.t:<mode> = ger
NP_r.b:<case> = nom/acc
NP_r.b:<agr num> = sing
NP_r.b:<agr pers> = 3
NP_r.b:<agr 3rdsing> = +
S_1.t:<assign-comp> = ecm
S_1.t:<comp> = nil

S_1.t:<inv> = -
S_1.t:<extracted> = -
NP_r.b:<gerund> = +
VP.b:<mode> = V.t:<mode>
VP.b:<passive> = V.t:<passive>
VP.b:<compar> = -
V.t:<passive> = -
NP_0:<case> = acc/gen
```

9 Tree "betaXnx0Vs1-PRO"

9.1 graphe



9.2 comments

```
ECM verb w/ PRO subject

John wants [PRO to see Bill eat beans].
While [PRO watching Bill eat beans] John got hungry.
```

9.3 features

```
S_r.b:<extracted> = -
S_r.b:<comp> = nil
S_r.b:<inv> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<assign-comp> = VP.t:<assign-comp>
S_r.b:<control> = NP_0.t:<control>
S_r.b:<assign-case> = NP_0.t:<case>
S_r.b:<wh> = NP_0.t:<wh>
NP_0:<agr> = S_r.b:<agr>
NP_0:<wh> = -
NP_0.t:<case> = none
S_r.b:<tense> = VP.t:<tense>
S_r.b:<agr> = VP.t:<agr>
VP.t:<mode> = inf/ger
VP.b:<compar> = -
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
V.t:<contr> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>

S_1.t:<assign-comp> = ecm
S_1.t:<inv> = -
S_1.t:<extracted> = -
S_1.t:<control> = NP_0.t:<control>
S_r.b:<punct contains> = VP.t:<punct contains>
VP.b:<punct contains> = S_1.t:<punct contains>
S_1.t:<comp> = nil
```