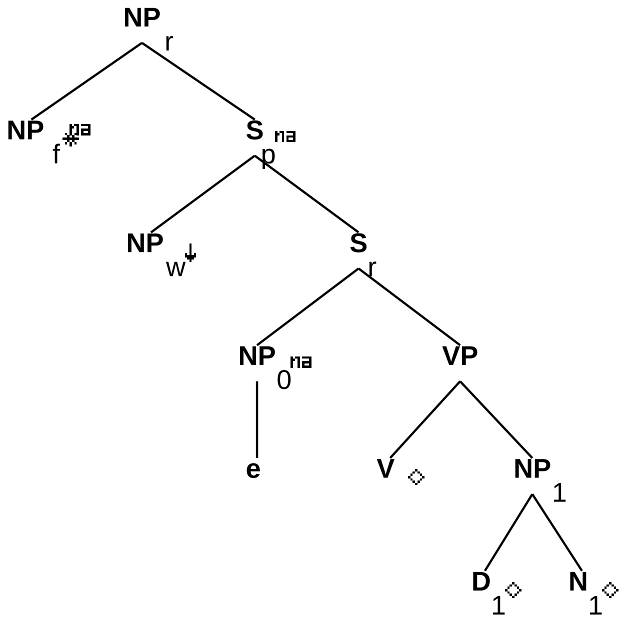
Family ”Tnx0VDN1”

March 5, 2008

1 Tree ”betaN0nx0VDN1”

1.1 graphe



1.2 comments

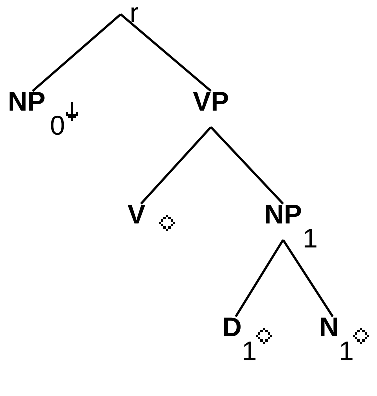
```
Transitive idiom with V, D, and N anchors.
Relative clause on the subject.

EX: [The president] who kicked the bucket...
```

1.3 features

```
S_r.t:<mode> = inf/ind
S_r.b:<comp> = nil
S_r.b:<mode> = VP.t:<mode>
S_r.b:<tense> = VP.t:<tense>
S_r.b:<assign-comp> = VP.t:<assign-comp>
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>
NP_0.t:<agr> = S_r.b:<agr>
NP_0.t:<case> = S_r.b:<assign-case>
NP_1:<case> = acc
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> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>
VP.b:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
N_1.t:<case> = nom/acc
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
```

2 Tree "alphanx0VDN1"

2.1 graphe



2.2 comments

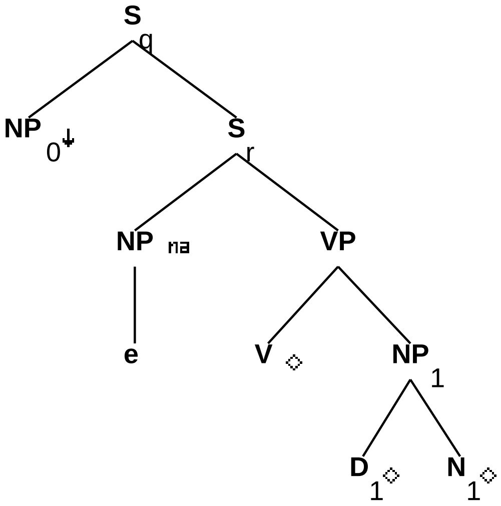
```
Transitive idiom with V, D, and N anchors.
Declarative.

EX: John buried the hatchet.
```

2.3 features

```
S_r.b:<extracted> = -



S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<tense> = VP.t:<tense>
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_1:<case> = acc
NP_0:<wh> = -
S_r.b:<wh> = NP_0:<wh>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-comp> = VP.t:<assign-comp>
S_r.b:<assign-case> = VP.t:<assign-case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
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>
VP.b:<compar> = -
S_r.b:<inv> = -
D_1:<agr> = NP_1.b:<agr>
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
S_r.b:<control> = NP_0.t:<control>
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

3 Tree ”alphaW0nx0VDN1”

3.1 graphe



3.2 comments

```
Transitive idiom with V, D, and N anchors.
Wh-question on the subject.

EX: Who kicked the bucket?
```

3.3 features

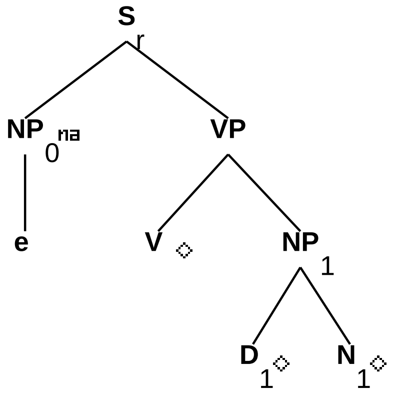
```
S_q.b:<extracted> = +
S_q.b:<inv> = S_r.t:<inv>
S_r.t:<comp> = nil
S_r.b:<assign-comp> = inf_nil/ind_nil/ecm
```

```
S_q.b:<wh> = NP_0:<wh>
S_q.b:<comp> = nil
S_q.b:<mode> = S_r.t:<mode>
S_r.b:<inv> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<tense> = VP.t:<tense>
NP.t:<trace> = NP_0.t:<trace>
NP.t:<agr> = NP_0.t:<agr>
NP.t:<case> = NP_0.t:<case>
NP.t:<wh> = NP_0.t:<wh>
```

```
NP_0:<wh> = +
NP.t:<agr> = S_r.b:<agr>
NP.t:<case> = S_r.b:<assign-case>
NP_1:<case> = acc
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
S_r.b:<assign-comp> = VP.t:<assign-comp>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
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>
VP.b:<compar> = -
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
S_r.t:<conj> = nil
```

4 Tree "alphaInx0VDN1"

4.1 graphe



4.2 comments

Transitive idiom with V, D, and N anchors.
Imperative.

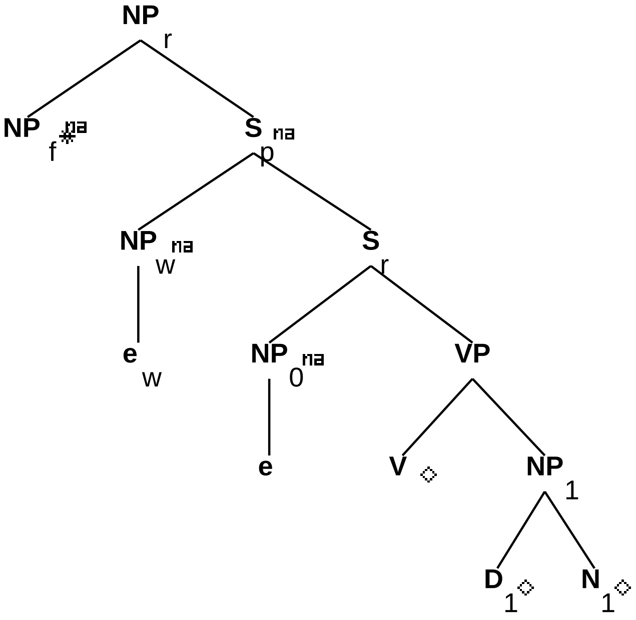
EX: Kick the bucket!

4.3 features

```
S_r.b:<extracted> = -
S_r.b:<comp> = nil



S_r.b:<inv> = -
S_r.b:<mode> = imp
S_r.b:<tense> = VP.t:<tense>
VP.t:<tense> = pres
S_r.b:<wh> = NP_0:<wh>
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_1:<case> = acc
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:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
S_r.b:<assign-comp> = VP.t:<assign-comp>
VP.t:<neg> = -
VP.t:<mode> = base
VP.b:<mode> = V.t:<mode>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
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> = -
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1:<agr> = NP_1.b:<agr>
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

5 Tree ”betaNc0nx0VDN1”

5.1 graphe



5.2 comments

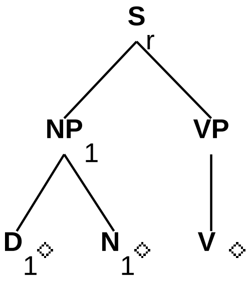
```
Transitive idiom with V, D, and N anchors.
 Relative clause on the subject, with overt Comp.

EX: [The man] that buried the hatchet...
```

5.3 features

```
S_r.b:<comp> = nil
S_r.b:<mode> = VP.t:<mode>
S_r.b:<tense> = VP.t:<tense>
S_r.b:<assign-comp> = VP.t:<assign-comp>
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>
NP_0.t:<agr> = S_r.b:<agr>
NP_0.t:<case> = S_r.b:<assign-case>
NP_1:<case> = acc
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> = -
VP.b:<agr> = V.t:<agr>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<mode> = V.t:<mode>
VP.b:<tense> = V.t:<tense>
VP.b:<mainv> = V.t:<mainv>
VP.b:<compar> = -
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_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
N_1.t:<case> = nom/acc
```

6 Tree "alphaDN1V"

6.1 graphe



6.2 comments

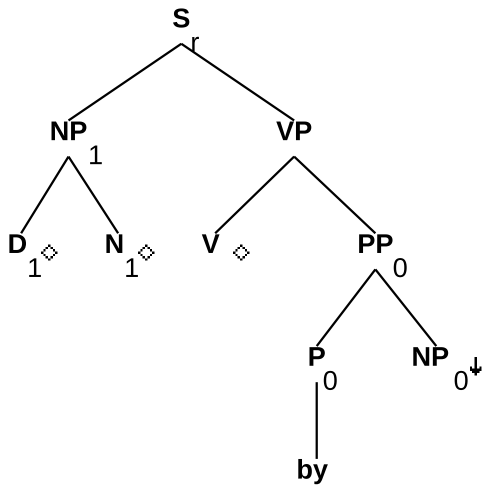
```
Transitive idiom with V, D, and N anchors.
Passive without by-phrase.

EX: The hatchet was buried.
```

6.3 features

```
S_r.b:<extracted> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<tense> = VP.t:<tense>
S_r.b:<wh> = NP_1:<wh>
NP_1:<agr> = S_r.b:<agr>
NP_1:<case> = S_r.b:<assign-case>
NP_1:<wh> = -
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
S_r.b:<assign-comp> = VP.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<tense> = V.t:<tense>
VP.b:<passive> = V.t:<passive>
VP.b:<agr> = V.t:<agr>
VP.b:<mainv> = V.t:<mainv>
VP.b:<compar> = -
V.t:<punct struct> = nil
V.t:<mode> = ppart
V.t:<passive> = +
S_r.b:<inv> = -
S_r.b:<control> = NP_1.t:<control>
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

7 Tree "alphaDN1Vbynx0"

7.1 graphe



7.2 comments

```
Transitive idiom with V, D, and N anchors.
Passive with by-phrase.

EX: The hatchet was buried by the warring parties.
```

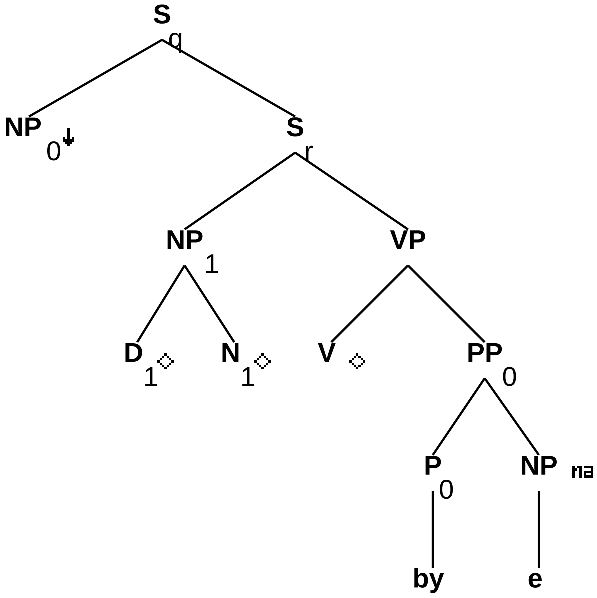
7.3 features

```
S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<extracted> = -
S_r.b:<tense> = VP.t:<tense>
S_r.b:<wh> = NP_1:<wh>
NP_1:<agr> = S_r.b:<agr>
NP_1:<case> = S_r.b:<assign-case>
NP_1.b:<case> = N_1.t:<case>
NP_1:<wh> = -
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-case> = VP.t:<assign-case>
S_r.b:<assign-comp> = VP.t:<assign-comp>
VP.b:<mode> = V.t:<mode>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<tense> = V.t:<tense>
VP.b:<passive> = V.t:<passive>
VP.b:<agr> = V.t:<agr>
VP.b:<mainv> = V.t:<mainv>
VP.b:<compar> = -
```

```
V.t:<punct struct> = nil
V.t:<mode> = ppart
V.t:<passive> = +
S_r.b:<inv> = -
PP_0.b:<assign-case> = P_0.t:<assign-case>
PP_0.b:<assign-case> = NP_0.t:<case>
P_0.b:<assign-case> = acc
S_r.b:<control> = NP_1.t:<control>
PP_0.b:<wh> = NP_0:<wh>
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

8 Tree ”alphaW0DN1Vbynx0”

8.1 graphe



8.2 comments

```
Transitive idiom with V, D, and N anchors.
Passive with wh-moved object of by-phrase.
```

```
EX: Who was the hatchet buried by?

Topicalization:

EX: Madeline the hatchet was buried by.
```

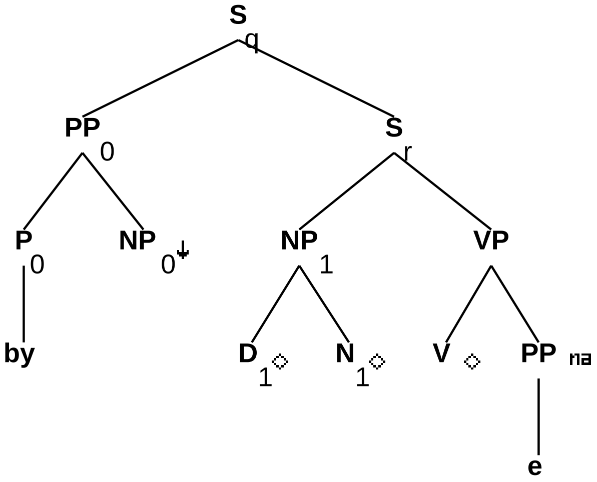
8.3 features

```
S_r.t:<comp> = nil
S_q.b:<extracted> = +




S_q.b:<wh> = NP_0:<wh>
S_q.b:<inv> = S_r.t:<inv>
S_q.b:<invlink> = S_q.b:<inv>
S_q.b:<mode> = S_r.t:<mode>
S_q.b:<comp> = nil
S_r.b:<inv> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
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:<assign-comp> = VP.t:<assign-comp>
S_r.b:<agr> = NP_1.t:<agr>
S_r.b:<assign-case> = NP_1.t:<case>
S_r.b:<control> = NP_1.t:<control>
VP.b:<passive> = +
VP.b:<mode> = V.t:<mode>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<tense> = V.t:<tense>
VP.b:<agr> = V.t:<agr>
VP.b:<mainv> = V.t:<mainv>
VP.b:<compar> = -
V.t:<mode> = ppart
V.t:<passive> = +
VP.b:<passive> = V.t:<passive>
V.t:<punct struct> = nil
NP.t:<agr> = NP_0.t:<agr>
NP.t:<case> = NP_0.t:<case>
NP.t:<trace> = NP_0.t:<trace>
NP.t:<wh> = NP_0.t:<wh>
P_0.b:<assign-case> = acc
PP_0.b:<assign-case> = P_0.t:<assign-case>
NP:<case> = PP_0.b:<assign-case>
S_r.t:<conj> = nil
PP_0.b:<wh> = NP:<wh>
```

```
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
D_1.t:<agr> = NP_1.b:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

9 Tree ”alphapW0DN1Vbynx0”

9.1 graphe



9.2 comments

```
Transitive idiom with V, D, and N anchors.
Passive with wh-moved by-phrase.

EX: By whom was the hatchet buried?

Topicalization:

EX: By Madeline the hatchet was buried.
```

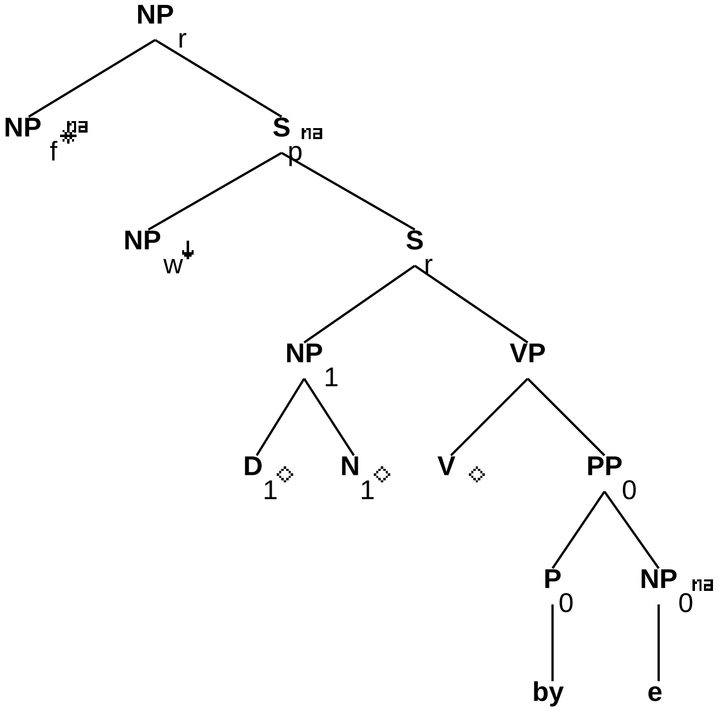
9.3 features

```
P_0.b:<assign-case> = acc
PP_0.b:<assign-case> = P_0.t:<assign-case>
```

```
S_q.b:<extracted> = +
S_q.b:<inv> = S_r.t:<inv>
S_q.b:<inv> = S_q.b:<invlink>

NP_0:<case> = PP_0.b:<assign-case>
PP_0.b:<wh> = NP_0:<wh>
S_q.b:<wh> = PP_0.t:<wh>
S_q.b:<mode> = S_r.t:<mode>
S_q.b:<comp> = nil
S_r.b:<inv> = -
S_r.b:<mode> = VP.t:<mode>
S_r.t:<comp> = nil
S_r.b:<comp> = nil
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:<assign-comp> = VP.t:<assign-comp>
S_r.b:<agr> = NP_1.t:<agr>
S_r.b:<assign-case> = NP_1.t:<case>
S_r.b:<control> = NP_1.t:<control>
VP.b:<passive> = +
VP.b:<mode> = V.t:<mode>
VP.b:<assign-case> = V.t:<assign-case>
VP.b:<assign-comp> = V.t:<assign-comp>
VP.b:<tense> = V.t:<tense>
VP.b:<agr> = V.t:<agr>
VP.b:<mainv> = V.t:<mainv>
VP.b:<compar> = -
V.t:<mode> = ppart
V.t:<passive> = +
V.t:<punct struct> = nil
VP.b:<passive> = V.t:<passive>
PP_0.t:<trace> = PP.t:<trace>
S_r.t:<conj> = nil
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

10 Tree "betaN0DN1Vbynx0"

10.1 graphe



10.2 comments

```
Transitive idiom with V, D, and N anchors.
Relative clause, extraction from by-phrase:

EX: [I saw] the man who the hatchet was buried by.
```

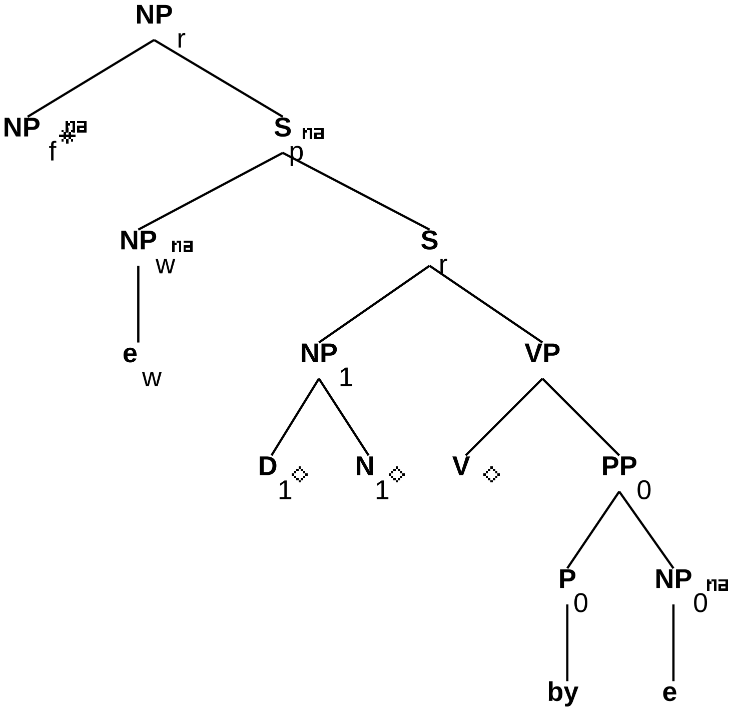
10.3 features

```
NP_f.t:<agr> = NP_r.b:<agr>
NP_f.t:<wh> = NP_r.b:<wh>
NP_f.t:<case> = NP_r.b:<case>
S_r.t:<mode> = ind/inf
S_r.b:<comp> = nil
S_r.b:<mode> = VP.t:<mode>
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:<assign-comp> = VP.t:<assign-comp>
S_r.b:<agr> = NP_1.t:<agr>
S_r.b:<assign-case> = NP_1.t:<case>
S_r.b:<control> = NP_1.t:<control>
```

```
VP.t:<mode> = ind
VP.b:<passive> = +
VP.b:<mode> = V.t:<mode>
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> = -
V.t:<mode> = ppart
V.t:<passive> = +
VP.b:<passive> = V.t:<passive>
VP.b:<agr> = V.t:<agr>
NP_f.b:<refl> = -
PP_0.b:<assign-case> = P_0.t:<assign-case>
PP_0.b:<assign-case> = NP_0.t:<case>
P_0.b:<assign-case> = acc
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
PP_0.b:<wh> = NP_0:<wh>
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
N_1.t:<case> = nom/acc
```

11 Tree "betaNc0DN1Vbynx0"

11.1 graphe



11.2 comments

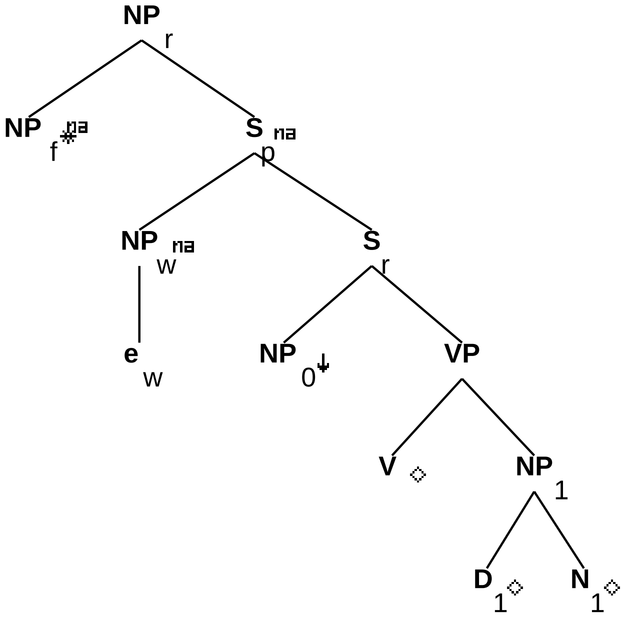
```
Transitive idiom with V, D, and N anchors.
'That' relative clause, extraction from by-phrase:

EX: [I saw] the man that the hatchet was buried by.
```

11.3 features

```
NP_f.t:<agr> = NP_r.b:<agr>
NP_f.t:<wh> = NP_r.b:<wh>
NP_f.t:<case> = NP_r.b:<case>
S_r.b:<comp> = nil
S_r.b:<mode> = VP.t:<mode>
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:<assign-comp> = VP.t:<assign-comp>
S_r.b:<agr> = NP_1.t:<agr>
S_r.b:<assign-case> = NP_1.t:<case>
S_r.b:<control> = NP_1.t:<control>
VP.t:<mode> = ind
VP.b:<passive> = +
VP.b:<mode> = V.t:<mode>
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> = -
V.t:<mode> = ppart
V.t:<passive> = +
VP.b:<passive> = V.t:<passive>
VP.b:<agr> = V.t:<agr>
NP_f.b:<refl> = -
PP_0.b:<assign-case> = P_0.t:<assign-case>
PP_0.b:<assign-case> = NP_0.t:<case>
P_0.b:<assign-case> = acc
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/ind
S_r.t:<nocomp-mode> = ind
VP.t:<assign-comp> = that/for/ind_nil
S_r.b:<nocomp-mode> = S_r.b:<mode>
NP_f.b:<case> = nom/acc
PP_0.b:<wh> = NP_0:<wh>
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
N_1.t:<case> = nom/acc
```

12 Tree ”betaNcnx0VDN1”

12.1 graphe



12.2 comments

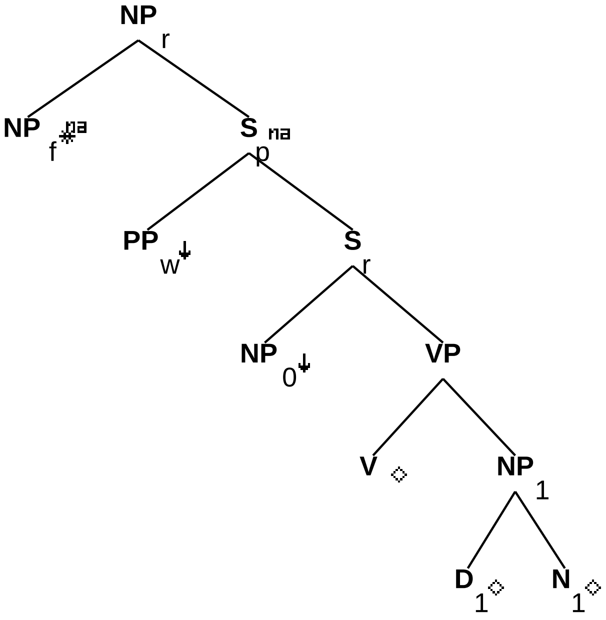
```
Transitive idiom with V, D, and N anchors.
Adjunct relative clause with overt Comp.

EX: [I saw a place] where John kicked the bucket.
```

12.3 features

```
S_r.b:<extracted> = -



S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<tense> = VP.t:<tense>
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_1:<case> = acc
NP_0:<wh> = -
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-comp> = VP.t:<assign-comp>
S_r.b:<assign-case> = VP.t:<assign-case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
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>
VP.b:<compar> = -
S_r.b:<inv> = -
S_r.b:<control> = NP_0.t:<control>
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_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
N_1.t:<case> = nom/acc
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

13 Tree ”betaNpxnx0VDN1”

13.1 graphe



13.2 comments

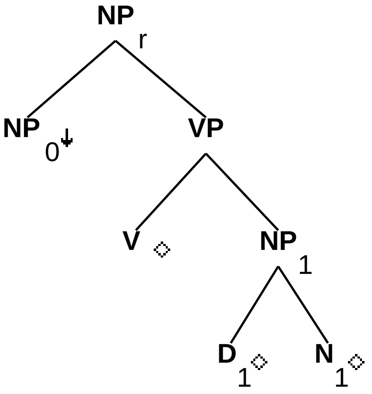
```
Transitive idiom with V, D, and N anchors.
Adjunct relative clause with PP.

EX: [I know a place] where Madeline buried the hatchet.
```

13.3 features

```
S_r.b:<extracted> = -



S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<tense> = VP.t:<tense>
NP_0:<agr> = S_r.b:<agr>
NP_0:<case> = S_r.b:<assign-case>
NP_1:<case> = acc
NP_0:<wh> = -
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-comp> = VP.t:<assign-comp>
S_r.b:<assign-case> = VP.t:<assign-case>
VP.b:<passive> = V.t:<passive>
V.t:<passive> = -
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>
VP.b:<compar> = -
S_r.b:<inv> = -
S_r.b:<control> = NP_0.t:<control>
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_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
N_1.t:<case> = nom/acc
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
```

14 Tree ”alphaGnx0VDN1”

14.1 graphe



14.2 comments

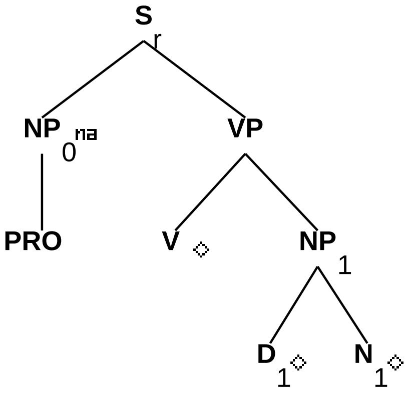
```
Transitive idiom with V, D, and N anchors - NP gerund.

[Graham('s) kicking the bucket] is the last thing we expected.
```

14.3 features

```
NP_0:<wh> = NP_r.b:<wh>
NP_r.t:<case> = nom/acc
NP_r.t:<agr num> = sing
NP_r.t:<agr pers> = 3
NP_r.t:<agr 3rdsing> = +
NP_1:<case> = acc



VP.b:<mode> = none
VP.b:<compar> = -
NP_r.b:<gerund> = +
V:<mode> = ger
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
N_1:<case> = nom/acc
NP_0:<case> = acc/gen
```

15 Tree "alphanx0VDN1-PRO"

15.1 graphe



15.2 comments

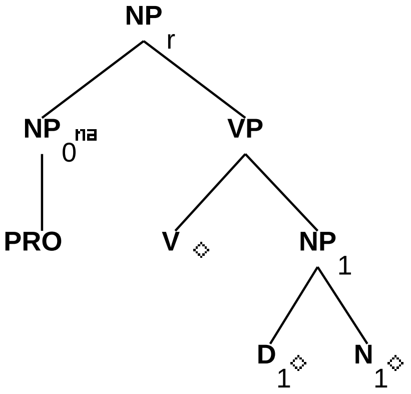
```
Transitive idiom with V, D, and N anchors, w/ PRO subject

John wanted [PRO to bury the hatchet].
While [PRO breaking the ice] the guests learned a great deal about each other.
```

15.3 features

```
S_r.b:<extracted> = -
S_r.b:<mode> = VP.t:<mode>
S_r.b:<comp> = nil
S_r.b:<tense> = VP.t:<tense>
S_r.b:<assign-case> = NP_0.t:<case>
S_r.b:<control> = NP_0.t:<control>
NP_0:<agr> = S_r.b:<agr>
NP_0:<wh> = -
NP_0.t:<case> = none
S_r.b:<wh> = NP_0:<wh>
S_r.b:<agr> = VP.t:<agr>
S_r.b:<assign-comp> = VP.t:<assign-comp>
S_r.b:<progressive> = VP.t:<progressive>
S_r.b:<perfect> = VP.t:<perfect>
S_r.b:<passive> = VP.t:<passive>
S_r.b:<mainv> = VP.t:<mainv>
VP.t:<mode> = inf/ger
VP.b:<passive> = V.t:<passive>
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>
VP.b:<compar> = -
V.t:<passive> = -
S_r.b:<inv> = -
D_1:<agr> = NP_1.b:<agr>
NP_1:<case> = acc
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<case> = N_1.t:<case>
NP_1.b:<wh> = N_1.t:<wh>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
```

16 Tree ”alphaGnx0VDN1-PRO”

16.1 graphe



16.2 comments

```
Transitive idiom with V, D, and N anchors - NP gerund w/ PRO subject

John was keen on [PRO burying the hatchet].
```

16.3 features

```
NP_0:<wh> = NP_r.b:<wh>
NP_0.t:<case> = none
NP_0.t:<wh> = -
NP_r.t:<case> = nom/acc
NP_r.t:<agr num> = sing
NP_r.t:<agr pers> = 3
NP_r.t:<agr 3rdsing> = +
NP_1:<case> = acc

VP.b:<mode> = none
VP.b:<compar> = -
NP_r.b:<gerund> = +
V:<mode> = ger
NP_1.b:<agr> = N_1.t:<agr>
NP_1.b:<compar> = N_1.t:<compar>
N_1.t:<compar> = -
D_1.t:<agr> = NP_1.b:<agr>
N_1:<case> = nom/acc
```