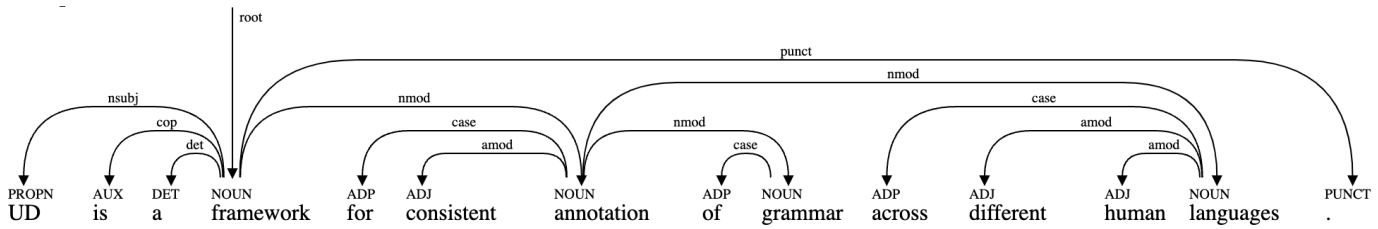
# Computational Syntax Exam 2024

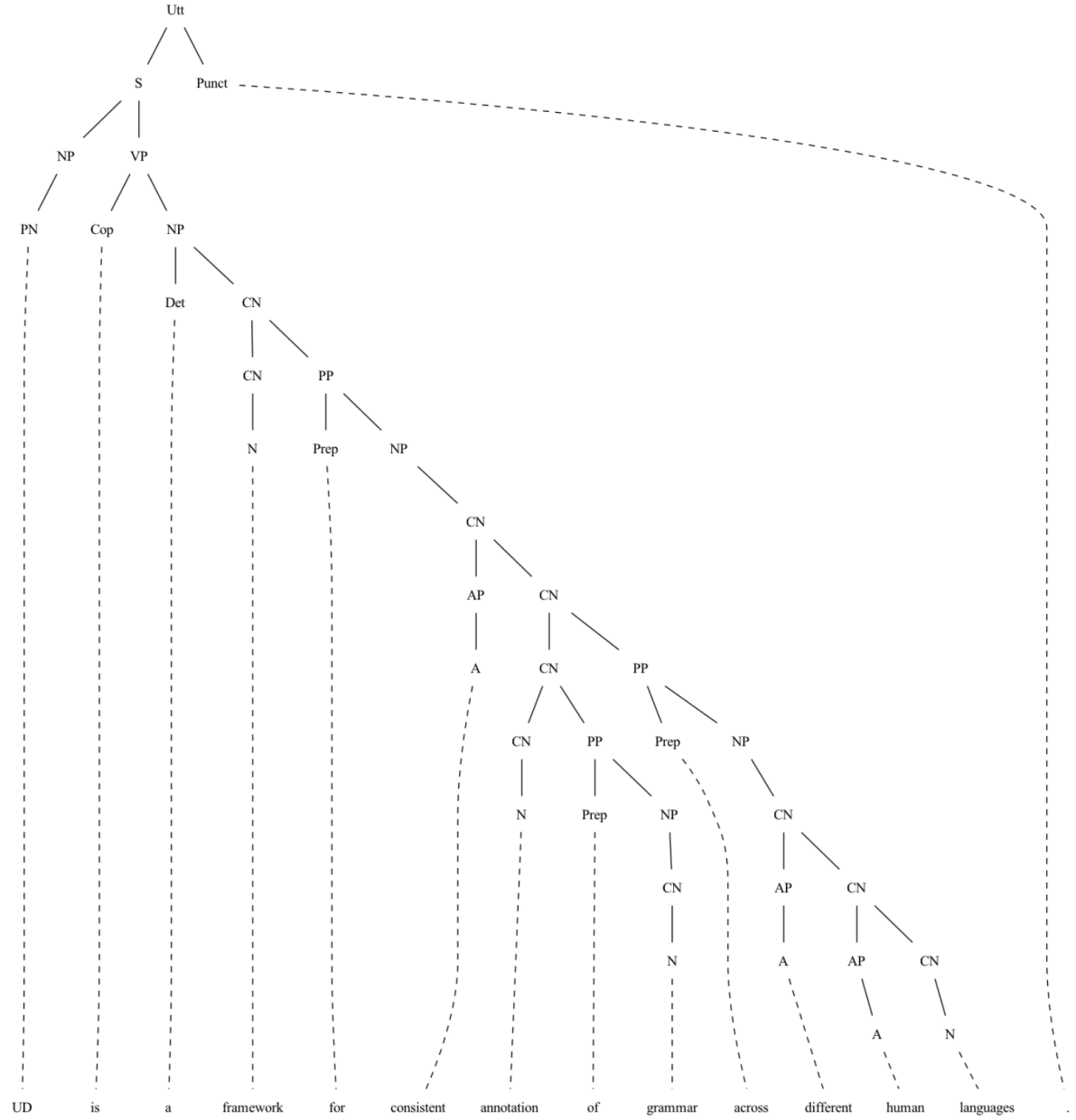
Solutions

Question 1

```
1     UD    UD    PROPN 4     nsubj
2     is    be    AUX   4     cop
3     a     a     DET   4     det
4     framework   framework   NOUN  0     root
5     for   for   ADP   7     case
6     consistent  consistent  ADJ   7     amod
7     annotation  annotation  NOUN  4     nmod
8     of    of    ADP   9     case
9     grammar     grammar     NOUN  7     nmod
10    across      across      ADP   13    case
11    different   different   ADJ   13    amod
12    human human ADJ   13    amod
13    languages   language    NOUN  7     nmod
14    .     .     PUNCT 4     punct
```



## Question 2




```
Utt ::= S Punct ;
S  ::= NP VP ;
VP ::= Cop NP ;
NP ::= PN ;
NP ::= Det CN ;
NP ::= CN ;
CN ::= CN PP ;
CN ::= AP CN ;
CN ::= N ;
AP ::= A ;
PP ::= Prep NP ;
Punct ::= "." ;
Cop ::= "is" ;
Det ::= "a" ;
Prep ::= "for" | "of" | "across" ;
PN ::= "UD" ;
N ::= "framework" | "annotation" | "grammar" | "languages" ;
A ::= "consistent" | "different" | "human" ;
```

Question 3

```
concrete SentenceSVO of Sentence = {
lincat
  S, NP, VP, V2 = Str ;
lin
  PredVP s vo = s ++ vo ;
  ComplV2 v o = v ++ o ;
}

concrete SentenceSOV of Sentence = {
lincat
  S, NP, VP, V2 = Str ;
lin
  PredVP s ov = s ++ ov ;
  ComplV2 v o = o ++ v ;
}

concrete SentenceVSO of Sentence = {
lincat
  S, NP, V2 = Str ;
  VP = {v, o : Str} ;
lin
  PredVP s vo = vo.v ++ s ++ vo.o ;
  ComplV2 v o = {v = v ; o = o} ;
}

concrete SentenceVOS of Sentence = {
lincat
  S, NP, VP, V2 = Str ;
lin
  PredVP s vo = vo ++ s ;
  ComplV2 v o = v ++ o ;
}

concrete SentenceOVS of Sentence = {
lincat
  S, NP, VP, V2 = Str ;
lin
  PredVP s ov = ov ++ s;
  ComplV2 v o = o ++ v ;
}

concrete SentenceOSV of Sentence = {
lincat
  S, NP, V2 = Str ;
  VP = {v, o : Str} ;
lin
  PredVP s vo = vo.o ++ s ++ vo.v ;
  ComplV2 v o = {v = v ; o = o} ;
}
```

Question 4

```
concrete SentenceSOV of Sentence = {
param
  Case = Nom | Acc | Gen | Dat | Abl ;
  Number = Sg | Pl ;
  Person = Per1 | Per2 | Per3 ;
lincat
  S = Str ;
  NP = {s : Case => Str ; n : Number ; p : Person} ;
  VP = {s : Number => Person => Str} ;
  V2 = {s : Number => Person => Str ; c : Case} ;
lin
  PredVP s ov = s.s ! Nom ++ ov.s ! s.n ! s.p ;
  ComplV2 v o = {s = \\n, p => o.s ! v.c ++ v.s ! n ! p} ;
}
```