

# Grammar Development with LFG and XLE

Miriam Butt  
University of Konstanz

# Last Time

- Lexical Rules (Passive, Dative Shift)
- Different Types of Functional Equations
  - defining
  - constraining
  - negative
  - existential
- Testsuites

# This Time: Lesson 4

1. LFG Treatment of Adjuncts
  - Adjectives and Adverbs
  - PPs
2. PP OBL vs. PP Adjunct
3. Semantically heavy vs. light Ps
4. Tokenization: Sentence Punctuation

# Tokenization

- So far we have parsed sentences that do not contain any punctuation.
- But this needs to be integrated into the grammar.
- Dealing with punctuation is part of **tokenization**.
  - capitalization
  - multiword items like *San Francisco*
  - abbreviations like *Mr.*, *Dr.*
  - *etc.*

# Tokenization

- Tokenizers in the ParGram grammars are generally implemented via finite-state transducers.
- They interact with the finite-state morphological analyzers (cascades of transducers).
- Creating good tokenizers is an art of its own.
  - Not part of this course.
  - The default tokenizers provided with XLE are very good.
  - But if working with different languages, will eventually have to build your own.

# Adjuncts vs. Arguments

- So far we have parsed sentences that consisted of:
  - a verbal head/predicate
  - NP arguments of that verbal head
- Arguments have the property that each type may only appear once.
  - A clause does not have two subjects
  - or two objects, etc.

# Adjuncts vs. Arguments

- But consider sentences like:
  - » The small, grey, intelligent dog devoured bones.
  - » The dog devoured large, delicious, juicy bones.
- Adjectives of the same type can appear multiple times.
- In principle, they can appear **infinitively** many times.
- In LFG adjectives are treated as Adjuncts.

# Types of Adjuncts

- Other common types of adjuncts are adverbs and PPs.
  - » The **very, very, very** large, **absolutely totally** black dog appeared.
  - » The dog barked [**in the garden**] [**under the tree**] [**on a rock**].
- Recall that so far we have treated PPs as OBL.
- An OBL is a governable grammatical function.
- PPs can function both as OBL or as an Adjunct.
- Sometimes it is not easy to tell the difference.



# Various kinds of PPs

- There are two types of OBL PPs
- Both types are characterized by the fact that the PP is required by the verb as an argument.
  - In one type the P expresses a (generally spatial) meaning:
    - » the gorilla put the banana in the tree
    - » \*the gorilla put the banana
  - In the other type the P has no separate or only a weak/light meaning:
  - Ex: the meaning of serving somebody in a restaurant
    - » the maitre'd waited on the customer
    - » \*the maitre'd waited on the customer on the boss
    - » the maitre'd waited (does not mean serve)

# Various kinds of PPs

- PPs that function as Adjuncts:
  - the P expresses a meaning of its own
  - any number of PPs can be used
  - the verb does not require the PPs
- Adjunct Example
  - spatial sense of 'on'
  - *wait* with *on* not in the sense of serving a customer
    - » the maitre'd waited on the bench on the bank of the river
    - » the maitre'd waited

# Adjunct PPs

- The Ps in Adjuncts are assumed to have a PRED.
- This is because they contribute their own semantics.
- In many languages, they also impose case restrictions on their arguments (like verbs do).

# Case Requirements

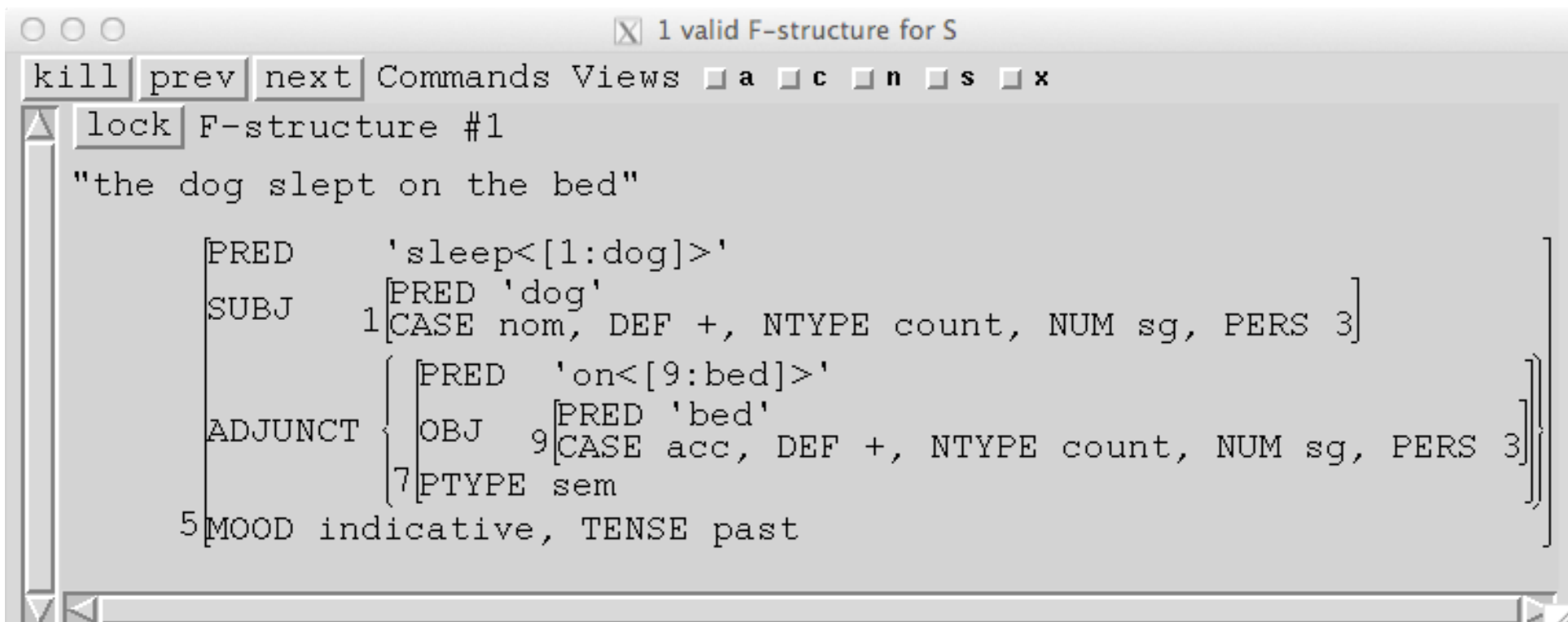
- In German, Ps require their objects to have a specific case.
- Some Ps allow for more than one case and this can give rise to meaning differences.

Der Hund rannte in dem Garten.  
the.Nom dog ran in the.**Dat** garden  
'The dog ran around in the garden.'

Der Hund rannte in den Garten.  
the.Nom dog ran in the.**Acc** garden  
'The dog ran into the garden.'

# Adjunct PPs

- The Ps in Adjuncts are assumed to have a PRED.
- The PRED subcategorizes for an object.
- For example:
  - (<sup>^</sup> PRED) = 'on<(<sup>^</sup>OBJ)>'



The screenshot shows a window titled "1 valid F-structure for S". Inside, there are buttons for "kill", "prev", "next", "Commands", "Views", and checkboxes for "a", "c", "n", "s", and "x". Below these is a section labeled "lock F-structure #1" containing the sentence "the dog slept on the bed". The F-structure is displayed as a nested list:

```
[PRED      'sleep<[1:dog]>'
 SUBJ      1[PRED 'dog'
              [CASE nom, DEF +, NTYPE count, NUM sg, PERS 3]
 ADJUNCT   { [PRED 'on<[9:bed]>'
               OBJ  9[PRED 'bed'
                       [CASE acc, DEF +, NTYPE count, NUM sg, PERS 3]
               ]
             [7]PTYPE sem
             ]
 MOOD      5[MOOD indicative, TENSE past]
```

# OBL PPs

- The non-semantic (wait *on*) Ps in OBLs are assumed not to have a PRED.
- They are simply registered via a PFORM attribute.
- For example:

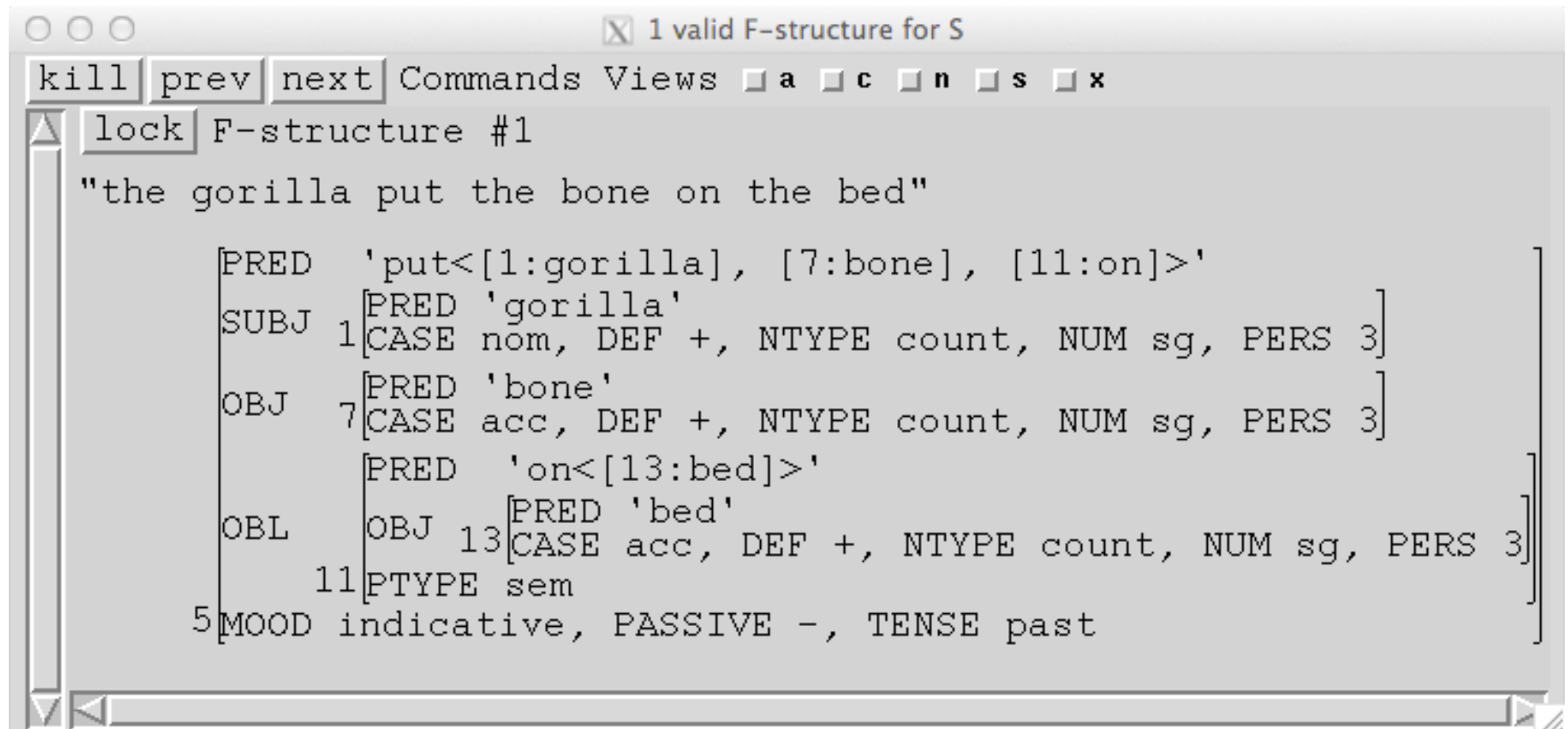
$(^{\wedge} \text{PFORM}) = \text{on}$

- The semantic Ps in OBLs are assumed to have a PRED.
- For example for *put the banana in the tree*

$(^{\wedge} \text{PRED}) = \text{'in} < (^{\wedge} \text{OBJ}) > \text{'}$

# OBL PPs

- Example for semantic P with OBL:

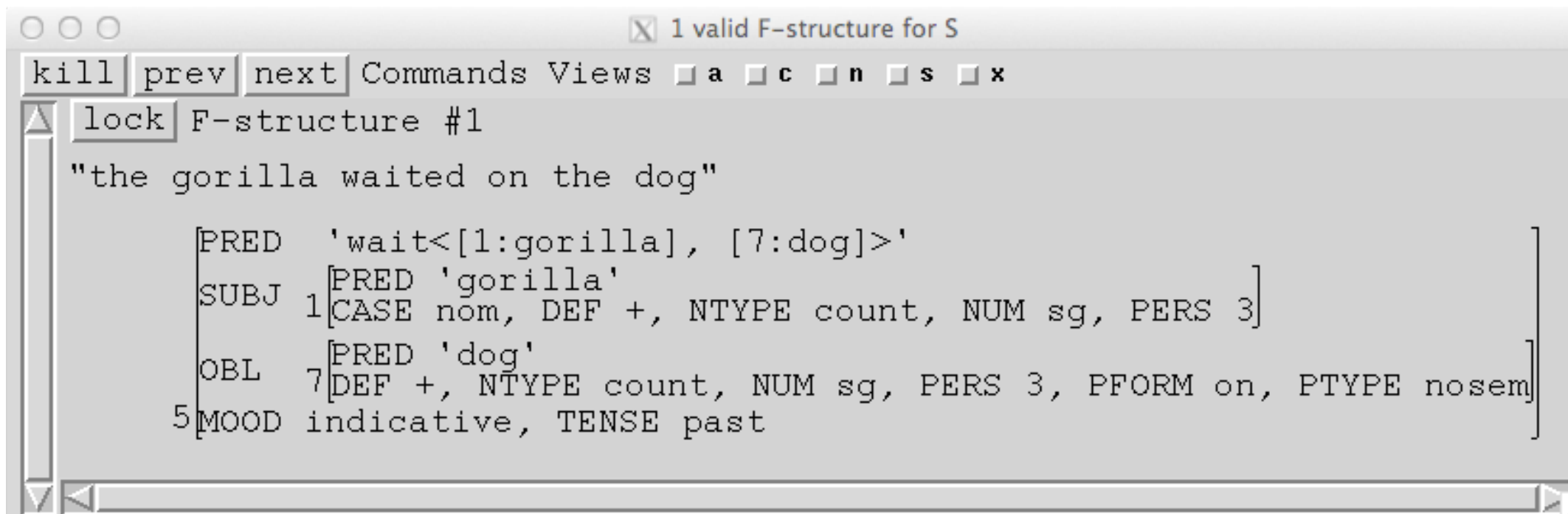


The screenshot shows a window titled "1 valid F-structure for S". The window has a menu bar with "kill", "prev", "next", "Commands", "Views", and checkboxes for "a", "c", "n", "s", and "x". Below the menu bar is a tab labeled "lock" and the text "F-structure #1". The main content area displays the sentence "the gorilla put the bone on the bed" followed by its F-structure representation. The F-structure is a nested list of semantic predicates and their arguments, with indices in brackets. The structure is as follows:

```
[PRED 'put<[1:gorilla], [7:bone], [11:on]>'
  SUBJ 1[PRED 'gorilla'
    CASE nom, DEF +, NTYPE count, NUM sg, PERS 3]
  OBJ 7[PRED 'bone'
    CASE acc, DEF +, NTYPE count, NUM sg, PERS 3]
  OBL 11[
    PRED 'on<[13:bed]>'
    OBJ 13[PRED 'bed'
      CASE acc, DEF +, NTYPE count, NUM sg, PERS 3]
    PTYPE sem
  ]
  MOOD indicative, PASSIVE -, TENSE past]
```

# OBL PPs

- Example for non-semantic P with OBL:



The screenshot shows a window titled "1 valid F-structure for S". The window has a menu bar with "kill", "prev", "next", "Commands", "Views", and checkboxes for "a", "c", "n", "s", and "x". Below the menu bar, there is a "lock" button and the text "F-structure #1". The main area of the window displays the sentence "the gorilla waited on the dog" followed by its F-structure representation in a nested list format:

```
[PRED 'wait<[1:gorilla], [7:dog]>'
SUBJ 1[PRED 'gorilla'
CASE nom, DEF +, NTYPE count, NUM sg, PERS 3]
OBL 7[PRED 'dog'
DEF +, NTYPE count, NUM sg, PERS 3, PFORM on, PTYPE nosem]
5MOOD indicative, TENSE past]
```



# Dealing with Adjuncts

- Now let us return to the status of adjuncts in LFG.
- Recall that LFG allows only one value for any given attribute.
- **But:** Adjuncts can appear multiple times.
- **Solution:** These elements are projected into elements of set-valued attributes such as ADJUNCT, MOD, etc.
- LFG Notation:  $\downarrow \in (\uparrow \text{ADJUNCT})$
- XLE Notation:  $! \$ (\wedge \text{ADJUNCT})$ 
  - down is an element of the adjunct set of the mother (up)

# Demo

**grammar3.lfg**  
**testsuite3.lfg**

**punctuation**  
**adjectives**  
**various types of PPs**

# Practical Work

- This concludes Lesson 4.
- The practical work you should do now is detailed in Exercise 4.
- You will practice with
  - adjectives and adverbs (adjuncts)
  - various types of PPs
  - punctuation

