# LSA 125 – Psycholinguistics and Syntactic Corpora

## Today: *Extracting and importing data from syntactic corpora into a database (2)*

LSA Summer Institute 2009, UC Berkeley

**Florian Jaeger**

**TAs: Judith Degen, Alex Fine, and Peter Graff**

# Log log log, log log around, …

- Our corpus server: (see website for most up-to-date info)
  - 174.129.5.193 (faster CPU)
  - 174.129.205.212 (maybe less traffic)
    - Login: *lsaXX,* e.g. *lsa18*
    - Password: only you know

# Q & A

- Do you have questions about UID described in the previous lecture?
- Questions about TGrep2?

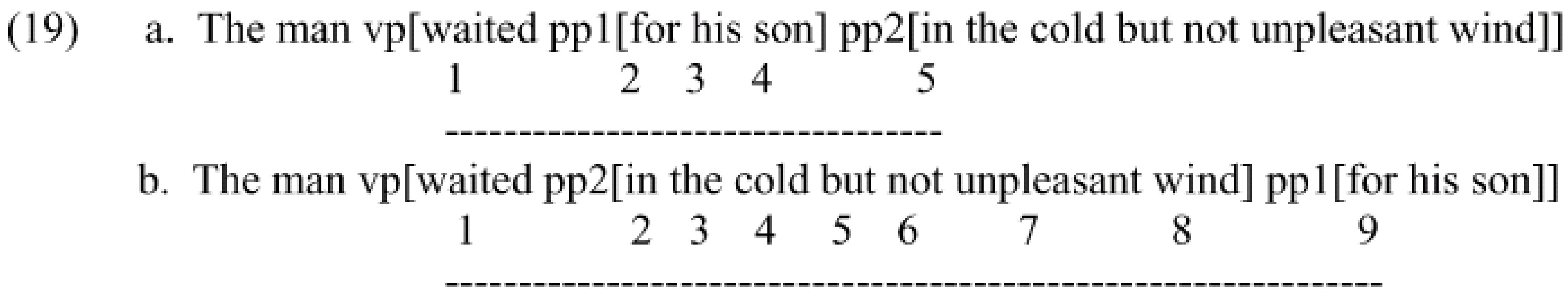
# Today

- More on:
  - **TGrep2** :: a tool to search syntactically-annotated corpora

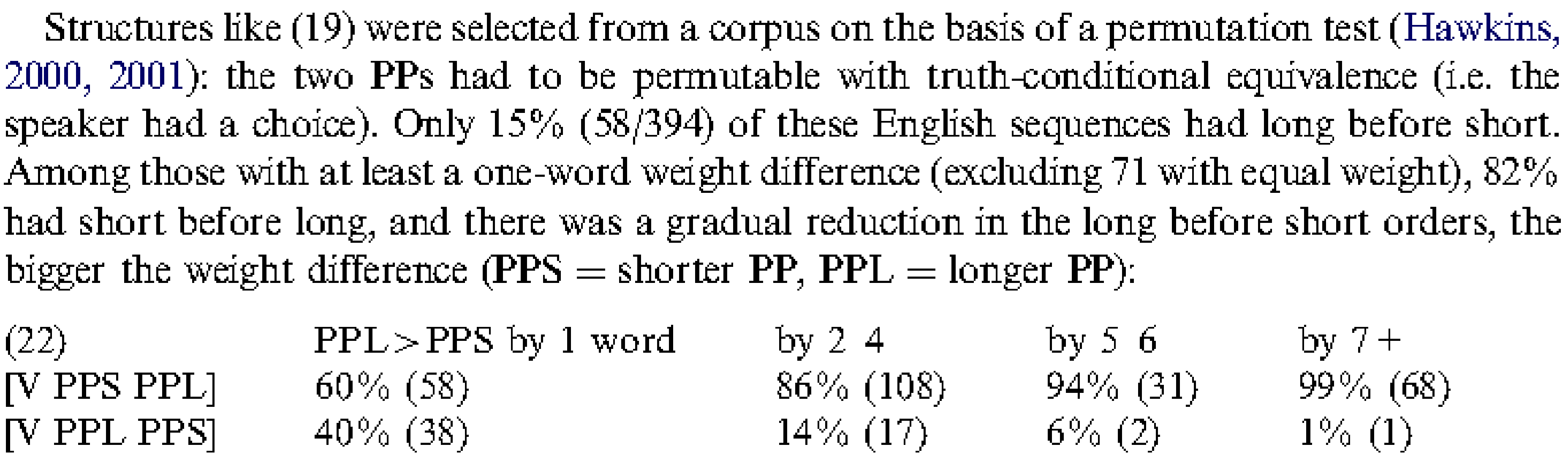
- Introduction to:
  - **TDT*lite*** :: a set of scripts we wrote to combine TGrep2 output into a database that can be handed to Excel or a stats program of your choice (e.g. R).

# Time to get real: PP-ordering in English

(Hawkins, 1999; taken from Hawkins, 2007:97)



```
(19)    a. The man vp[waited pp1[for his son] pp2[in the cold but not unpleasant wind]]
                        1           2   3   4          5
                        ----------------------------------
        b. The man vp[waited pp2[in the cold but not unpleasant wind] pp1[for his son]]
                        1           2   3   4   5   6      7        8          9
                        -------------------------------------------------------------
```



Structures like (19) were selected from a corpus on the basis of a permutation test (Hawkins, 2000, 2001): the two **PPs** had to be permutable with truth-conditional equivalence (i.e. the speaker had a choice). Only 15% (58/394) of these English sequences had long before short. Among those with at least a one-word weight difference (excluding 71 with equal weight), 82% had short before long, and there was a gradual reduction in the long before short orders, the bigger the weight difference (**PPS** = shorter **PP**, **PPL** = longer **PP**):

| (22) | PPL > PPS by 1 word | by 2 4 | by 5 6 | by 7 + |
|---|---|---|---|---|
| [V PPS PPL] | 60% (58) | 86% (108) | 94% (31) | 99% (68) |
| [V PPL PPS] | 40% (38) | 14% (17) | 6% (2) | 1% (1) |

# Macros

<SPACE>

<TAB>

- Macros keep those precious fingers soft and smooth by avoiding to much typing

```
@ NP       /^NP/;
@ VP       /^VP/;
@ PP       /^PP/;
@ AP       /^(ADJ|ADV)P/;
@ WH       /^WH/;
@ PPPP     /^VP/=VP1 < (/^PP/=PP1 $.. (/^PP/=PP2  !$ (/^PP/
   != =PP1) !,, (* !< * ,, =PP1 !>> (EDITED|UH|PRN|/-UNF/
   >> =VP1))));
```

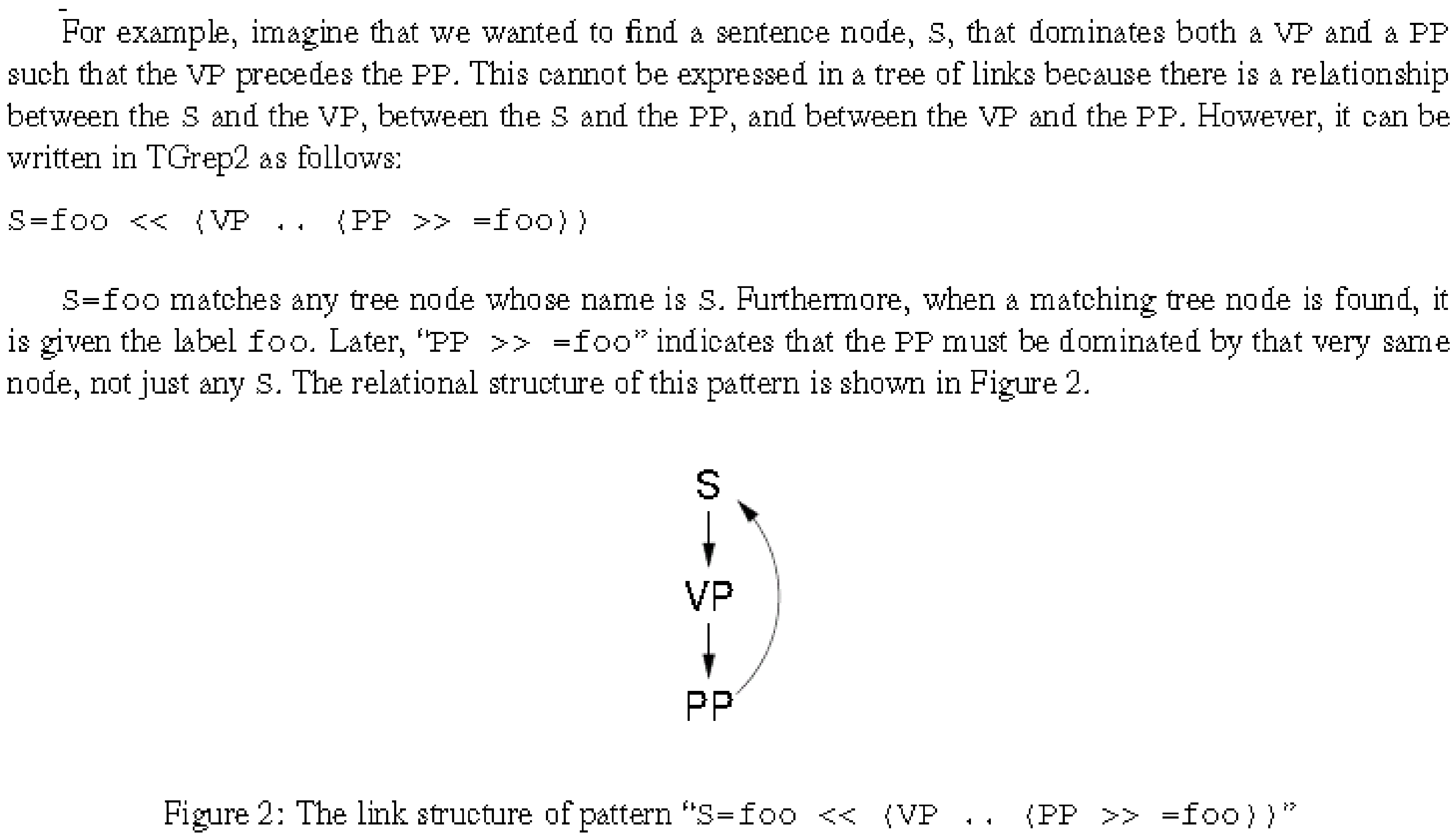
No quotes

# More on TGrep2

- All relations
- Negation
- Disjunction (conjunction=default)

- Head vs. match
  - Multiple marked nodes

# Node labeling and links to nodes



For example, imagine that we wanted to find a sentence node, S, that dominates both a VP and a PP such that the VP precedes the PP. This cannot be expressed in a tree of links because there is a relationship between the S and the VP, between the S and the PP, and between the VP and the PP. However, it can be written in TGrep2 as follows:

```
S=foo << (VP .. (PP >> =foo))
```

S=foo matches any tree node whose name is S. Furthermore, when a matching tree node is found, it is given the label foo. Later, “PP >> =foo” indicates that the PP must be dominated by that very same node, not just any S. The relational structure of this pattern is shown in Figure 2.

Figure 2: The link structure of pattern “S=foo << (VP .. (PP >> =foo))”

# Two types of links

- Back-link
  - If reference to label is an ancestor of labeled node → node identity (as in example on previous page)

- Crossing-link: not permitted → node is copied (i.e. not node-identity but node-description identity)

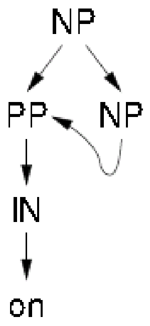


Figure 3: The initial structure of pattern "NP < (PP=pp < (IN < on)) | < (NP < =pp)"

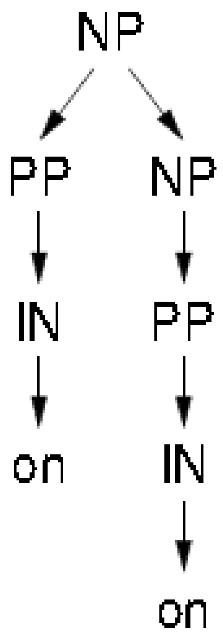


Figure 4: The final structure of pattern "NP < (PP=pp < (IN < on)) | < (NP < =pp)"

- To see the difference, use *tgrep2 -z* with the two patterns

# Output options

- -m vs. -s
- subtrees
  - %h
  - %m
  - %w
  - %=nodelabel=
  - %*N*b, %*N*a

# Output options

- subtree style
  - t, l, u, n, x
  - k, d, y, z

- Additional formatting:
  - \t, \n
  - Anything else (but escape % as %%)

# A *totally* different project: Macros

- Macros keep those precious fingers soft and smooth by avoiding to much typing

```
@ NP      /^NP/;
@ VP      /^VP/;
@ PP      /^PP/;
@ AP      /^(ADJ|ADV)P/;
@ WH      /^WH/;
@ PPPP    /^VP/=VP1 < ((/^AD(J|V)/=PP1 !<< -NONE-) $.
   (/^AD(J|V)/=PP2 !<< -NONE- !$ (/^PP/ != =PP1)));
```

# Directory structure

- Project dir
  - /ptn
  - /data
  - /results
  - /shellscripts

# /data, /results, /ptn

- /results: final results
- /data:  usually just one corpus, *.t2o
- /ptn: tgrep2 pattern files *.ptn
  - CatVar
  - StringVar
  - ContVar
  - CountVar
  - POSVar
  - ParseVar
  - CtxtVar

# /shellscripts

- macro-file for project,
  - general macros
  - macros used in project
- two scripts (*run*, *getOptions.py*): you only need to use *run*
- configuration file (*options*): needed by run
- Environment variables:
  - TDTlite, TDT_DATABASES, TGREP2ABLE
  - Path: add TDTlite directory