Grammar Engineering

NLP Retreat
Chelan, WA

Olga Zamaraeva
Department of Linguistics, University of Washington

September 29 2019

Grammar Engineering

- *Grammar*: a parser/generator which reflects language rules according to a linguistic (e.g. syntactic) theory
- Cross-linguistic: The same core (principles of structure) should be applicable to any language
- Precise: parses have meaningful syntactic and semantic structure; ungrammatical sentences should not be possible
- Demo: http://delph-in.github.io/delphin-viz/demo/
- Applications: Linguistic hypothesis testing (and also in industry, e.g. Grammarly)

Linguistic hypotheses in syntax

- A set of 'elegant' principles, rules, and item structures which together generate only grammatical sentences in a language

Sir Galahad, the Quest for the Holy Grail, by Arthur Hughes (1870)

Linguistic Hypothesis Testing with Paper-and-pencil Syntax

- Accounting for **all**(?) of the data 
- ...manually and by introspection

Linguistic Hypothesis Testing with Implemented Grammars

Examine coverage and overgeneration of a grammar which encodes your analyses.

Support for Constituent Questions in a Grammar Engineering Framework

- *Who did what to whom?* How to engineer models for such questions in general?
- Meta-Grammar Engineering System: The Grammar Matrix (Bender et al. 2002; 2010)
- Input: grammar specification (in typological terms), output: grammar fragment
- Using HPSG (syntax) and MRS (semantics) formalisms
- Tested broadly but on small scale (many languages, small test suites), in terms of coverage and overgeneration

Constituent (aka Wh-) Questions

- *Who did what?*
  - **What did who?*
  - **Who what did?*
- *Whose friend's book do you know that Sandy took?*
- **Whose do you know that Sandy took friend's book?*
- ...etc.

Constituent Questions Library for the Grammar Matrix

In **matrix** questions (*Who did you see?*):

- (●) A single question phrase can be fronted
- ( ) All question phrases can be fronted (only choose this if multiple questions are allowed)
- ( ) Question phrases cannot be fronted (stay in situ)

There is **obligatory** fronting:
- (●) of at least one question phrase
- ( ) of all question phrases (only choose this if multiple questions are allowed)

Pied-piping: question determiners can be extracted along with [x] nouns, [x] and it is obligatory; [ ] adpositions, [ ] and it is obligatory.

In **embedded** questions: (*Do you know who did this?*)

- (●) A single question phrase can be fronted within the embedded clause
- ( ) All question phrases can be fronted within the embedded clause (only choose this if multiple questions are allowed)
- ( ) Question phrases cannot be fronted (stay in situ) within the embedded clause

Within the embedded clause, there is obligatory fronting:
- (●) of at least one question phrase
- ( ) of all question phrases (only choose this if multiple questions are allowed)

Pied-piping: Wh-determiners can be extracted along with [ ] nouns, [ ] and it is obligatory; [ ] adpositions, [ ] and it is obligatory.

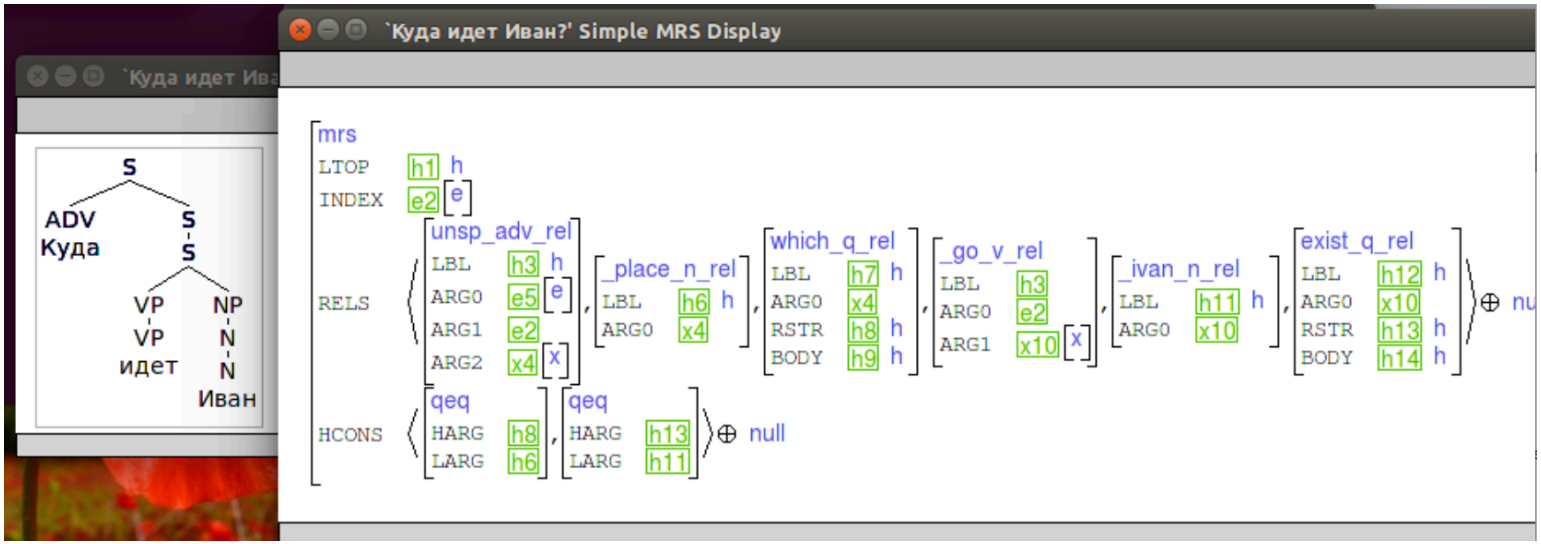
When the question phrase is extracted **from an embedded clause** and put in **the front of the matrix clause** (*Who do you think that did this?*):

[ ] All other question phrases can also be fronted

There is **obligatory** extraction from the embedded clause to the front of the matrix clause:
- (●) of at least one question phrase
- ( ) of all question phrases (only choose this if multiple questions are allowed)

Pied-piping: Wh-determiners can be extracted along with [x] nouns, [x] and it is obligatory; [ ] adpositions, [ ] and it is obligatory.

Where does Kim go? [rus] MRS



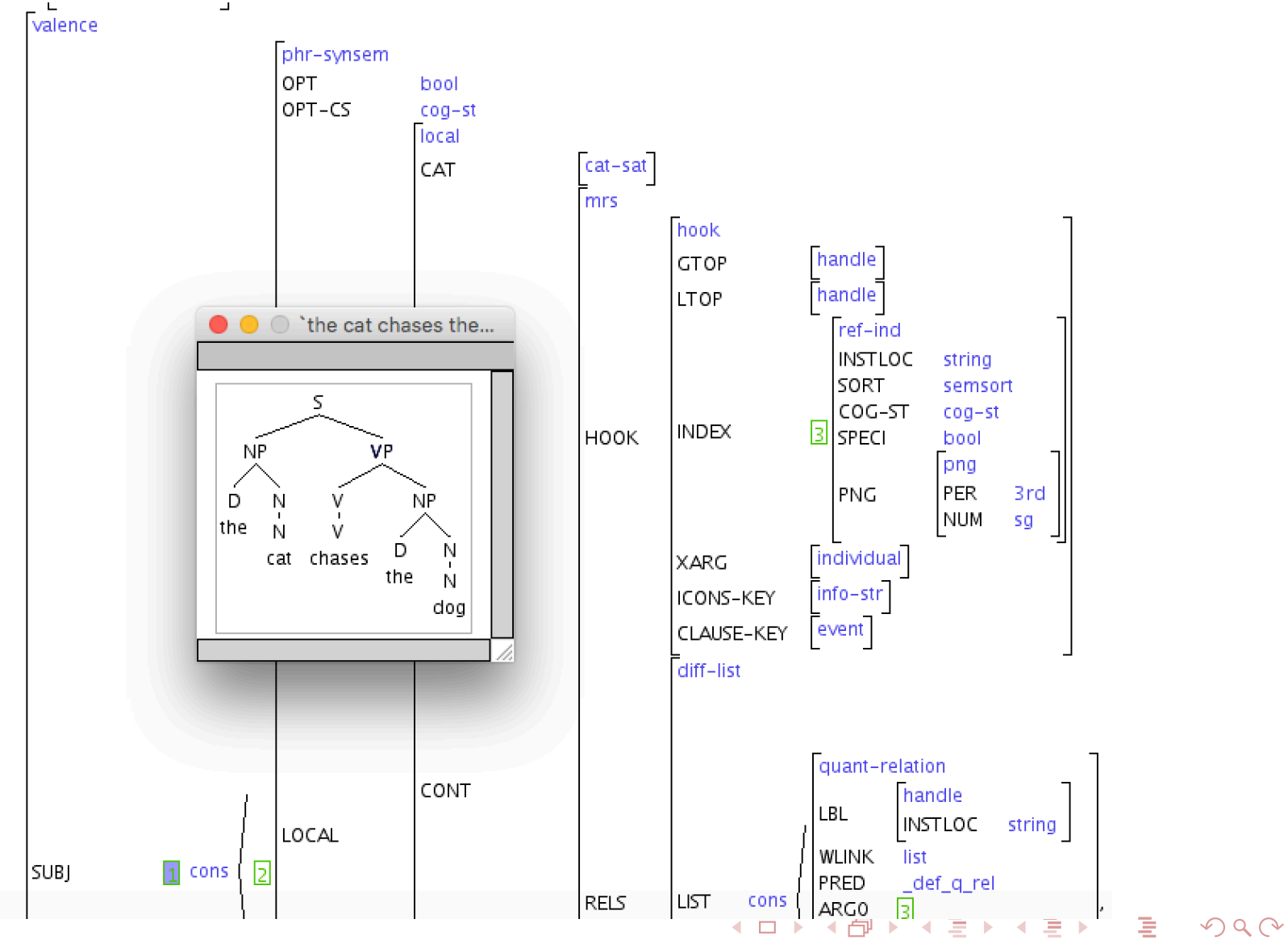
- (Bag of predications with hooks; dependency graph easily obtainable)
- The heavy ligting here is done by the syntactic analyses which are not shown (they are the approximation of the 'Holy Grail')
- E.g. what does a generic lexical entry for *where* look like? What does the adjunct extraction look like?

Structure and Evaluation

Scenario: Given a typological description  , how well does the system model a language  (in linguistic terms )?

- Questionnaire
  - Typological choices
- Generic analyses (structures) derived from development grammars
  - Russian, English, Japanese, Chukchi, Abui
- Artificial languages (as 'unit' tests)
- Evaluation on held-out language families

The big question: What should a theory of syntax really look like? Like this? :)



The big question: What should a theory of syntax really look like?

- Actually still unclear!
- No universal formalism
- Lack of distinction between "true" linguistic complexity and imposed engineered complexity
  - Insights from software engineering?
- Should an alternative formalism be...
  - Higher-level?
    - But can a grammar be 'simple'?
  - Bottom-up, motivated directly by the data?
    - insights from NLP?