Introduction to "Grammatical Framework" – a programming language for writing grammars

Rainer Osswald and Yulia Zinova

SoSe 2016

Rules for this class

- This course includes both theoretical material and practice
- 14 sessions (final session 18.07, no class on 16.05)
- Attendance is not enforced
- Homework is due by the next session, unless specified otherwise
- Late submission = 50% points
- Assignments will be graded, 60% of the total amount of points is needed to get a BN
- When submitting programming assignments, include source code and several screenshots showing how you tested it
- For an AP, you need to do the assignments and a project after the course (writing your own grammar)

For an AP

- You will have to describe a part of grammar of a language not described in GF or extend an existing description (harder).
- Each student doing an AP should be describing a separate piece of morphology, no joint submissions.
- To find material for the project, go to the library and study the shelves with grammars of languages you don't know.
- You have to show us the material you want to work with and receive our approval prior to programming the grammar.

AP – Grades

- 30% of your grade are the assignments and 70% is the project you do after the class
- Formal grading criteria for the project will be available later
- The correspondence between the percentage of points and grades is the following:
  - 1.0: 95 – 100%
  - 1.3: 91 – 94%
  - 1.7: 87 – 90%
  - 2.0: 83 – 86%
  - 2.3: 80 – 82%
  - 2.7: 75 – 79%
  - 3.0: 70 – 74%
  - 3.3: 65 – 69%
  - 3.7: 60 – 65%
  - 4.0: 50 – 59%

Important links

- Course website: `https://user.phil-fak.uni-duesseldorf.de/~zinova/teaching/GF/index.html`
- Our emails: `osswald@phil.hhu.de`, `zinova@phil.hhu.de`
- We will follow the book by Aarne Ranta "Programming with Multilingual Grammars" (Ranta, 2011)
- GF website: `http://www.grammaticalframework.org/`

Grammars or statistics?

- Two approaches: symbolic approach vs. statistical approach
- Symbolic approach: give computers grammar (rules)
- Statistical approach: give computers data and use statistics/machine learning
- Pros and cons of these methods? What do **you** think? What about learning languages?

Current situation

- In practice, statistical approaches dominate in such areas as information retrieval, machine translation, speech recognition.
- In this course, however, we will show how to use grammars for practical purposes and how they can do the job statistics cannot.

Formal and natural languages

- Both formal and natural languages have grammars
- What is the difference between those grammars?

Formal and natural languages

- Both formal and natural languages have grammars
- What is the difference between those grammars?
- Formal languages are *defined* by grammars.
- Natural languages exist independently of the grammar, grammars are theories, artifacts.
- Consequences?

Formal and natural languages

- Both formal and natural languages have grammars
- What is the difference between those grammars?
- Formal languages are *defined* by grammars.
- Natural languages exist independently of the grammar, grammars are theories, artifacts.
- Consequences?
- Incompleteness/overgeneration

Do we need grammars at all?

- Grammars are never sufficient to describe a language completely.
- In some systems, all the rules are induced and no hand-written rules are used.
- So why bother?

Do we need grammars at all?

- Grammars are never sufficient to describe a language completely.
- In some systems, all the rules are induced and no hand-written rules are used.
- So why bother?
- For humans, grammars provide a *shortcut* and replace a lot of data
- Knowing grammar usually improves *quality* of the produced text/speech
- The more structurally complex is the language, the more data you need to replace grammar

Long-distance dependencies

- Google translate, English to French (15.09.2010):
  my father immediately became very worried
  mon père est immédiatement devenu très inquiet
- Correct! Change *father* to *mother*:
  ma mère est immédiatement devenu très inquiet
- Correct would be:
  *ma mère est immédiatement devenu***e** *très inquiet***e**
- Problem: with *n*-gram models, either *n* is too small to capture long-distance dependencies at all, or the data is not sufficient to find the needed *n*-gramm

Discontinuous constituents

- Try translating with Google translate (German to English):
  Ich bringe ihn um. Ich bringe meinen besten Freund um.

Find some balance

- A possible solution is to use grammars when they are available.
- Manage **precision** and **coverage** by using **hybrid systems**
- Apply **smoothing** techniques if something is not described in the grammar
- Manage the **cost of grammars** by finding efficient ways to describe them: Grammatical Framework!

Cost of grammars

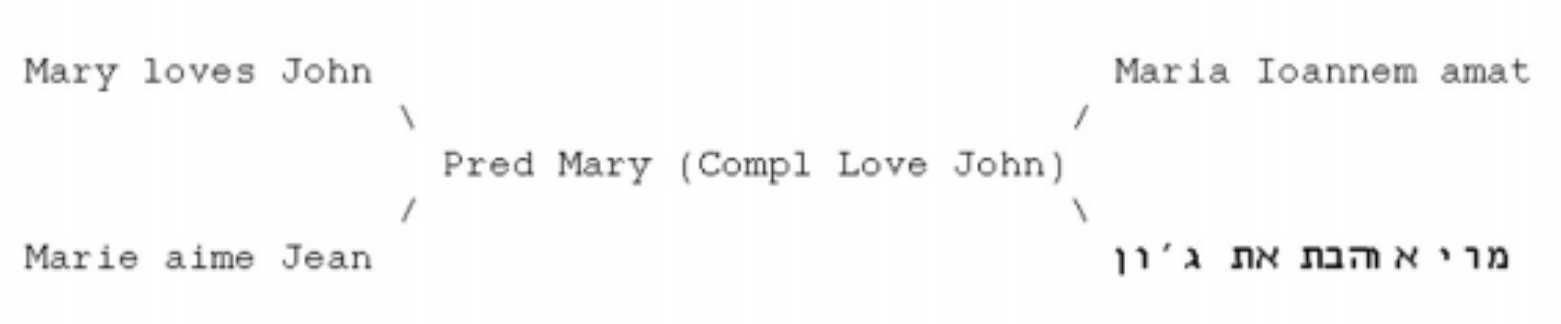
- 2 types of costs: to write and to run
- Worst-case parsing complexity for a mildly context-sensitive grammar is $O(n^6)$
- How GF lowers costs:
  - **static type system** detects many programming errors automatically
  - **module system** supports division of labor
  - **functional programming** enables powerful abstractions
  - **libraries** allow to build on earlier grammars
  - **compilers** convert GF grammars to other formats
  - **tools for information extraction** convert other linguistic resources to GF

Multilinguality

- GF can deal with several languages at one time
- Two components: **abstract syntax** and **concrete syntax**
- Abstract syntax is a tree-like representation that captures the semantically relevant structure of language
- Concrete syntax relates the tree structure with linear string representations
- One abstract syntax is equipped with various concrete syntaxes
- Abstract syntax is independent of features like word order and inflection, it only cares about constituency

Multilingual grammars

Main idea: represent many languages related by a common abstract syntax



Semantic actions

- So far we have talked about two actions: parsing and linearization
- Semantics actions are operations on the syntax tree
- **Transfer-based translation**: a tree received from parsing the source is transformed into another tree before it is linearized
- **Question answering system**: semantic action converting question trees into answer trees
- **Dialogue system**: questions can be given in pieces (search for flight prices)
- GF uses **embedded grammars**: grammars can be accessed from code in other programming languages, which can be used to perform semantic actions

Application grammars

- Application grammar: abstract syntax functions as a semantic model
- Applications have specific domains. How does knowing the domain help?

Application grammars

- Application grammar: abstract syntax functions as a semantic model
- Applications have specific domains. How does knowing the domain help?
- Domain facilitates word sense disambiguation
- Domain may have idiomatic sentence (mathematical exercises are written in imperative in English and in infinitive in French)
- Application grammars are also called semantic grammars, as abstract syntax is modeled after semantic structure
- Semantic structure may be defined in mathematical logic o in an ontology and then ported to GF

Resource grammars

- Syntactic grammar: capture syntactic structures, or grammatical operations
- Syntactic grammars deal with what is meant by *syntax* in linguistics and cover larger parts of language
- When parsing, we can use both types of grammars and receive different trees: tree with the syntactic structure and with semantic structure
- Syntactic grammars can be used as **libraries** for writing semantic grammars
- Grammar composition: use of abstract syntax trees of one grammar in the concrete syntax of another grammar

Division of labour

- The resource grammar is written by a linguist, who knows the rules of agreement, word order, etc.
- The application grammar is written by a domain expert, wo knows the terminology, domain idioms, etc.
- Resource Grammar Library: 200 grammatical functions implemented for 16 languages
- For each of these languages, the library has a complete inflectional morphology (set of functions that are capable of producing all forms of all words)

References:

Ranta, A. (2011). *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications.