Cross-Lingual Syntax: Relating Grammatical Framework with Universal Dependencies

Aarne Ranta
University of Gothenburg
aarne@chalmers.se

Prasanth Kolachina
University of Gothenburg
prasanth.kolachina@gu.se

Thomas Hallgren
University of Gothenburg
hallgren@chalmers.se

Abstract

GF (Grammatical Framework) and UD (Universal Dependencies) are two different approaches using shared syntactic descriptions for multiple languages. GF is a categorial grammar approach using abstract syntax trees and hand-written grammars, which define both generation and parsing. UD is a dependency approach driven by annotated treebanks and statistical parsers. In closer study, the grammatical descriptions in these two approaches have turned out to be very similar, so that it is possible to map between them, to the benefit of both. The demo presents a recent addition to the GF web demo, which enables the construction and visualization of UD trees in 32 languages. The demo exploits another new functionality, also usable as a command-line tool, which converts dependency trees in the CoNLL format to high-resolution LaTeXand SVG graphics.

1 Introduction

GF (Grammatical Framework, (Ranta, 2011) and UD (Universal Dependencies, (Nivre et al., 2016)) are two attempts to use shared syntactic descriptions for multiple languages. In GF, this is achieved by using **abstract syntax trees**, similar to the internal representations used in compilers and to Curry's tectogrammatical formulas (Curry, 1961). The trees can be converted to strings in different languages by **linearization functions**, similar to pretty-printing rules in compilers and to Curry's phenogrammatical rules. Linearization rules are reversible to parsers (Ljunglöf, 2004).

In UD, the shared descriptions are dependency labels, part of speech tags and morphological features used in dependency trees. The words in the leaves of UD trees are language-specific, and each language can extend the core descriptions to have a set of its own tags and labels. The relation between trees and strings is not defined by grammar rules, but by constructing a set of example trees — a treebank. From a treebank, a parser is typically built using statistical methods (Nivre, 2006).

The abstract syntax trees of GF can be automatically converted to UD trees (Kolachina and Ranta, 2016), by utilizing the shared abstract syntax of GF to allow simultaneous generation of UD trees in many languages. The proposed demo shows tools that use this conversion. An example is shown in Figure 3, whose contents are produced by these tools. The configurations used are defined for the GF Resource Grammar Library (GF-RGL) (Ranta, 2009), which currently contains 32 languages. An inverse conversion from UD to GF is work in progress (Ranta and Kolachina, 2017) and can also be shown in the demo.
All grammars, configurations, and software are available from the GF homepage. [1]

2 Command-line functionalities

The richest set of functionalities is available in the GF shell (the program launched by the command `gf`). Figure 1 shows a part of the help entry for these functionalities.

The `-conll2latex` option can be independently interesting for dependency parser community. It is the only tool known to us that converts CoNLL to standard LaTeXcode (lines, ovals, etc) with no extra packages required. The same code base also produces SVG graphics from CoNLL.

3 UD trees in the incremental parser interface

This functionality is the easiest one to test, since it does not require any software to be installed, other

[1] www.grammaticalframework.org

Proceedings of the 21st Nordic Conference of Computational Linguistics, pages 322–325,
Gothenburg, Sweden, 23-24 May 2017.

```
options:
 -v              show extra information
 -conll2latex    convert conll to latex

flags:
 -file           configuration file for labels, format per line 'fun label*'
 -output         output format of graph source (dot, latex, conll)
 -view           program to open the resulting file

examples:
  gr | vd -view=open -output=latex                      --generate tree, show on Mac
  gr -number=1000 | vd -file=dep.labels -output=conll   --generate random treebank
  rf -file=ex.conll | vd -conll2latex | wf -file=ex.tex --convert conll file to latex
```

Figure 1: The usage of GF shell command `vd` = `visualize_dependency`.

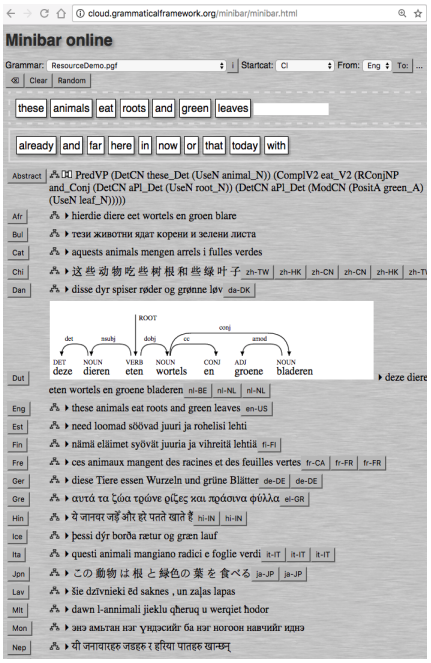

Figure 2: Dependency tree in the "minibar" incremental parser interface. Clicking the tree symbol for each language shows the UD tree. A second click shows the phrase structure tree. (The choice of words in this purely syntax-based generation can be wrong: e.g. the verb "eat" when used for animals should in German be *fressen'*.)

than a web browser. It builds on GF's web-based tool set (Ranta et al., 2010). It is accessible via the GF Cloud[2]. The currently supported grammar option is "ResourceDemo"; see Figure 2.

4 UD trees in the PGF web service

For grammars that have been equipped with a UD label configuration file, UD trees can be requested from the PGF web service. An example request: `http://cloud.grammaticalframework.org/grammars/ResourceDemo.pgf?command=deptree&format=svg&to=ResourceDemoEng&tree=...`

The GF web demo Minibar allows users to select a grammar, construct grammatical sentences in one language and see translations to the other languages supported by the grammar. UD trees can be displayed next to the translations by clicking on a tree icon (Figure 2).

5 Annotating Configurations

The example in Figures 2 and 3 is produced by the following abstract syntax configurations included in an annotation file:

```
PredVP   nsubj head
DetCN    det head
ComplV2  head dobj
RConjNP  cc head conj
ModCN    head amod
```

A configuration consists of an abstract syntax function together with a list of labels, one for each argument of the function. An extended notion of these configurations is described in (Kolachina and Ranta, 2016). The basic algorithm is a top-down tree-transducer that deterministically maps each argument of a function in the abstract syntax tree to its UD label, generating a connected dependency tree. We refer the reader to Kolachina and Ranta (2016) for more details.

[2]http://cloud.grammaticalframework.org/minibar/minibar.html

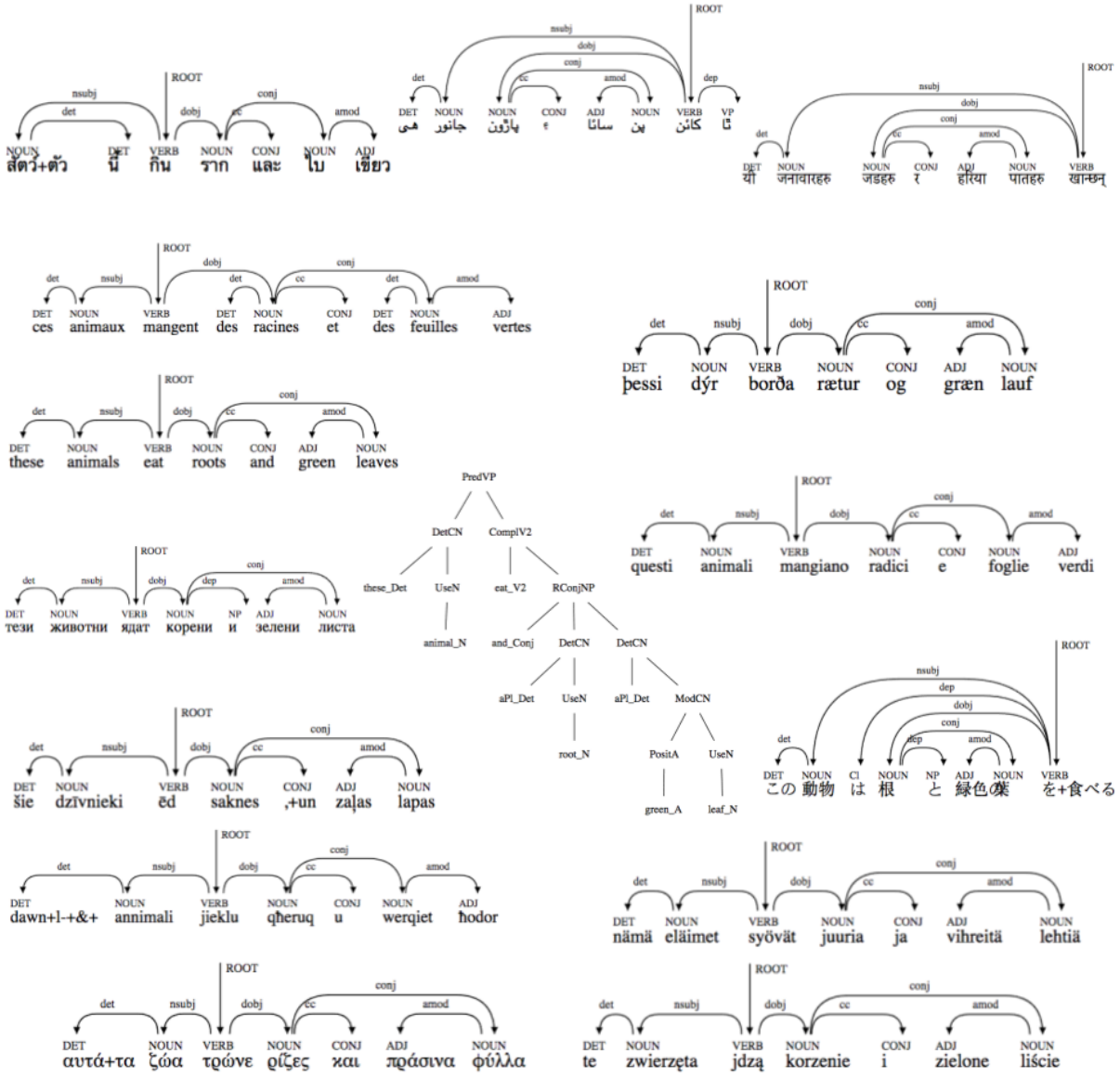

Figure 3: An abstract syntax tree and UD trees in 14 languages. The abstract syntax tree is shown in the middle. The languages corresponding to the UD trees from top-left: Thai, Sindhi, Nepali, French, Icelandic, English, Italian, Bulgarian, Latvian, Japanese, Maltese, Finnish, Greek, Polish. The gf2ud function uses language-independent configurations specified on the abstract syntax to simultaneously generate UD trees for all the languages. Here, we use png dump of the high-resolution originals, due to difficulties in rendering all the fonts in LaTeX.

References

Haskell B. Curry. 1961. Some Logical Aspects of Grammatical Structure. In *Structure of Language and its Mathematical Aspects: Proceedings of the Twelfth Symposium in Applied Mathematics*, pages 56–68. American Mathematical Society.

Prasanth Kolachina and Aarne Ranta. 2016. From Abstract Syntax to Universal Dependencies. *Linguistic Issues in Language Technology*, 13(2).

Peter Ljunglöf. 2004. *The Expressivity and Complexity of Grammatical Framework*. Ph.D. thesis, Department of Computing Science, Chalmers University of Technology and University of Gothenburg.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, May. European Language Resources Association (ELRA).

Joakim Nivre. 2006. *Inductive Dependency Parsing*. Springer.

Aarne Ranta and Prasanth Kolachina. 2017. From Universal Dependencies to Abstract Syntax. In *NoDaLiDa Workshop on Universal Dependencies (UDW 2017) (to appear)*, Gothenburg, Sweden.

Aarne Ranta, Krasimir Angelov, and Thomas Hallgren. 2010. Tools for Multilingual Grammar-Based Translation on the Web. In *Proceedings of the ACL 2010 System Demonstrations*, pages 66–71, Uppsala, Sweden, July. Association for Computational Linguistics.

Aarne Ranta. 2009. The GF Resource Grammar Library. *Linguistic Issues in Language Technology*, 2(2).

Aarne Ranta. 2011. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford.