Robust and Statistical Parsing in GF (C runtime)

Krasimir Angelov, Aarne Ranta, Shafqat Virk, Lauri Alanko

University of Gothenburg and University of Helsinki

MOLTO
Final presentation, May 2013

Overview

- The New C Runtime
  - statistical ranking
  - robust parsing
  - Python binding
- Applications in Translation
  - English Resource Grammar + Large Lexicon
  - Penn Treebank for GF
  - Prototype Online Translator

The C Runtime for GF

Initial Goals:

- small and efficient
- portable
- easy to embed in other languages (Python, Haskell, Prolog, etc)

Note: the main GF runtime is in Haskell

Using GF from Python

```
import pgf

gr = pgf.readPGF("ParseEngAbs.pgf")

try:
  for (p,e) in gr.languages["ParseEng"].parse(sent, n=5)
    sys.stdout.write("["+str(p)+"] "+str(e)+"\n")
    print gr.languages["ParseBul"].linearize(e)
except pgf.ParseError as e:
  print e.message
```

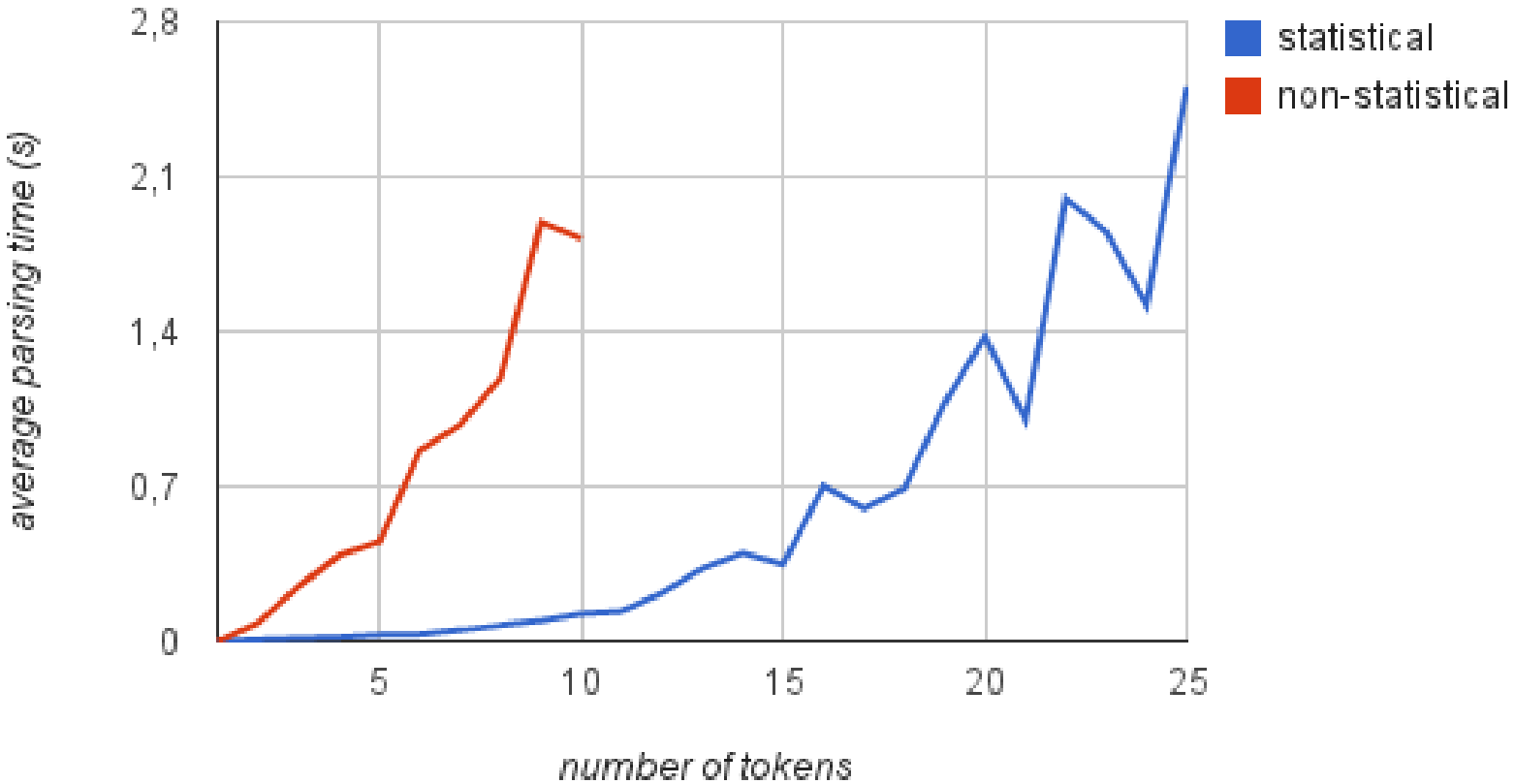
The C Runtime

Grew Into:

- statistical engine for PMCFG which is **beyond the state of the art**
  - up to **100 times faster** than state of the art alternatives
  - producing the same results
- the engine behind an experimental GF-based translator

The New vs The Old Parser

Parsing time for the English RGL + large lexicon

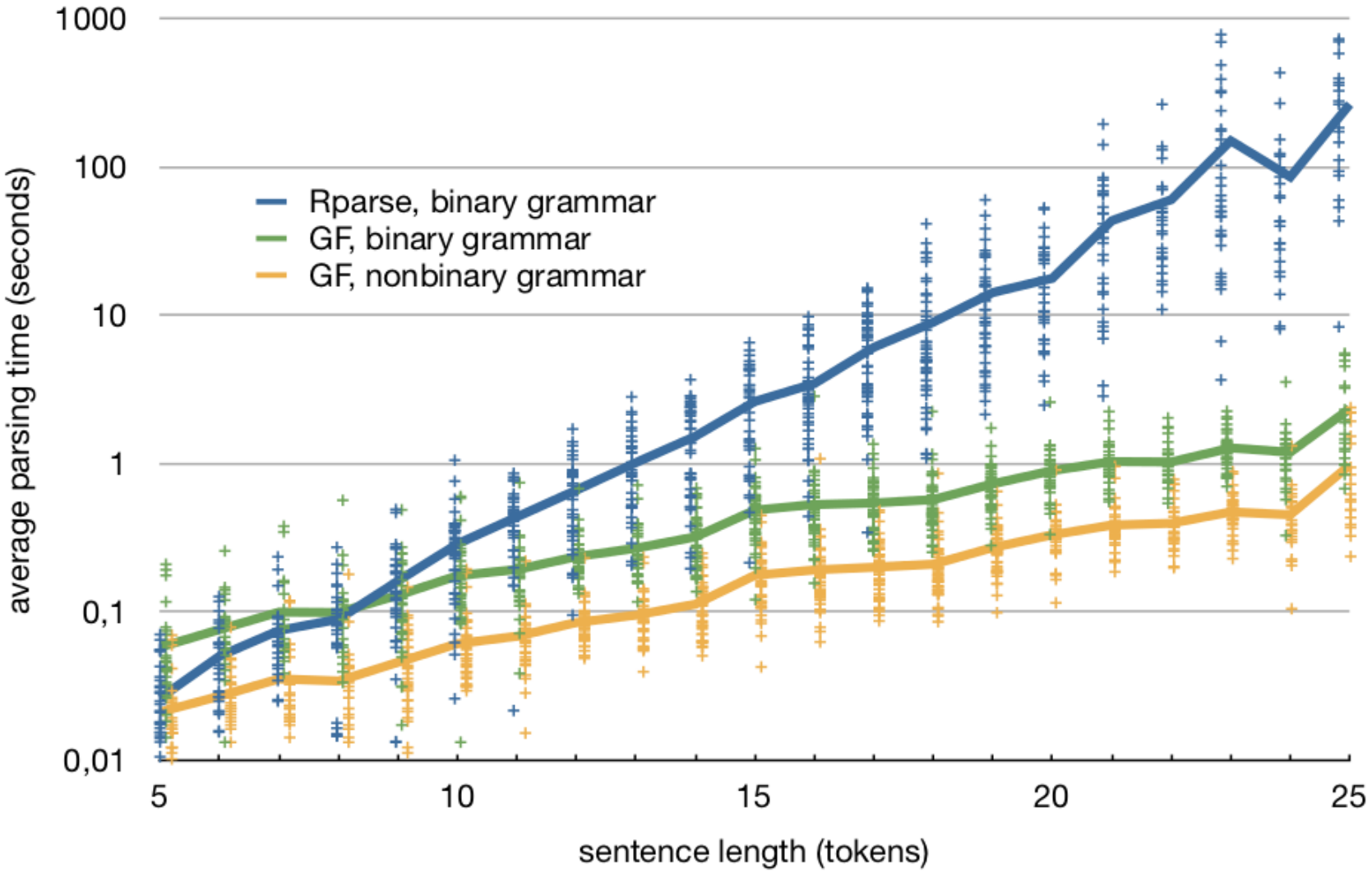


RParse

RParse is

- state of the art statistical parser for binary RCG (Kallmeyer and Maier, 2010).
- used primarily for parsing discontinuous German phrases
- using grammar learned from the German Tiger Treebank (Brants et al., 2002)

The GF Parser vs RParse



Speed up

The speed up comes from:

- the use of a different algorithm
  Angelov (2009) vs Kallmeyer and Maier (2010)
- the statistical model guides the parser to explore first the most likely branches
- if only the best tree is needed there is no need to explore the whole search space

Background

- The new parser unifies the algorithm in Angelov (2009) with the statistical ranking in Stolcke (1995)
- A baseline A^* search algorithm plus an optional non-admissible heuristic.
- Following Stolcke (1995), the parser is also made robust by allowing the construction of partial trees.
- The algorithm is lazy, i.e. it can ultimately return all parse trees, but only the necessary part of the search space is explored

Parsing Grammar

For the prototype translator we needed a large-coverage grammar

- the English grammar from RGL
- Large Lexicon of about 50 000 lemmas derived from:
  - Oxford Advanced Learners Dictionary
  - Princeton WordNet
  - Verb Valency Frames from the Penn Treebank

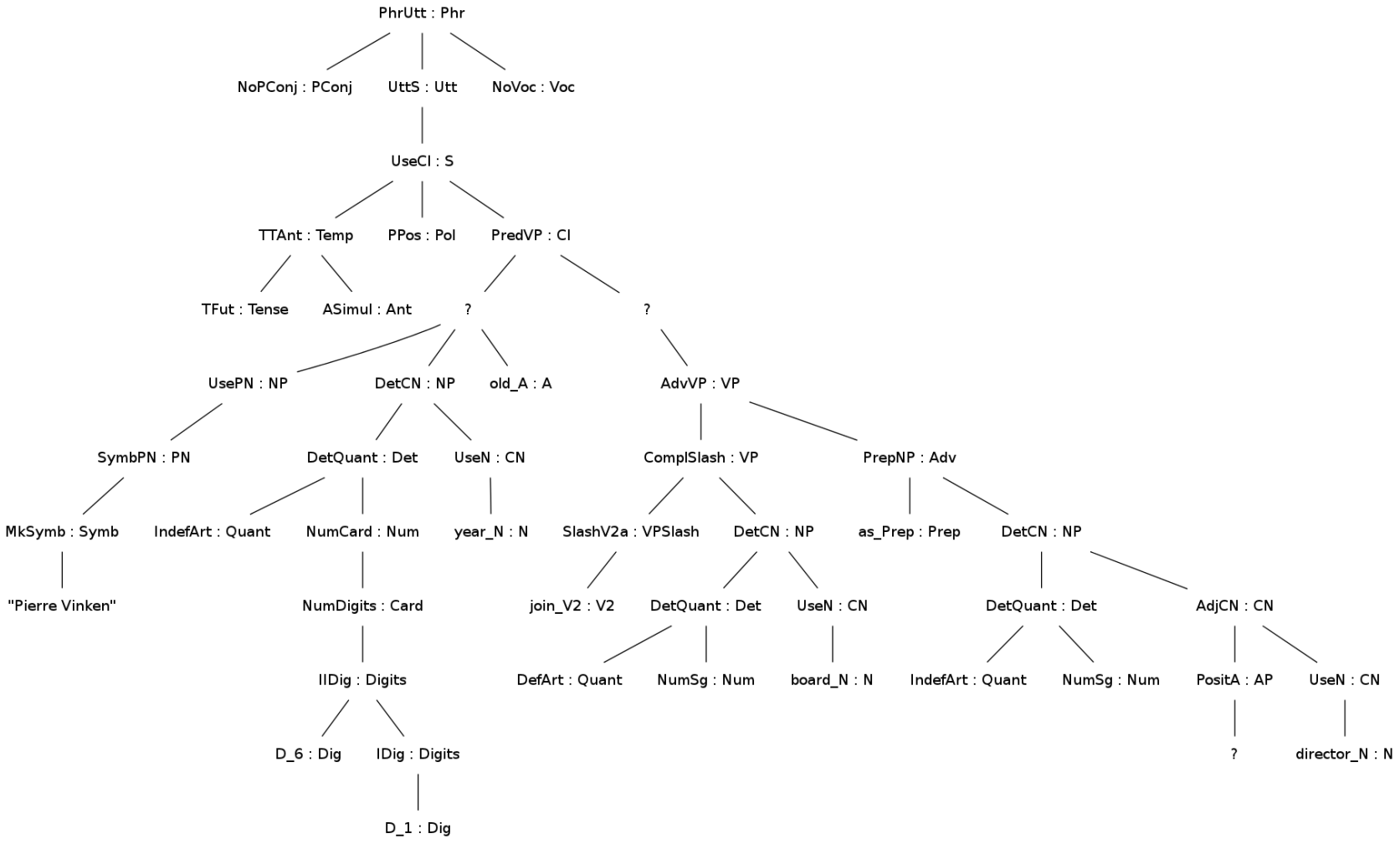
Penn Treebank

For statistical parsing we need training data

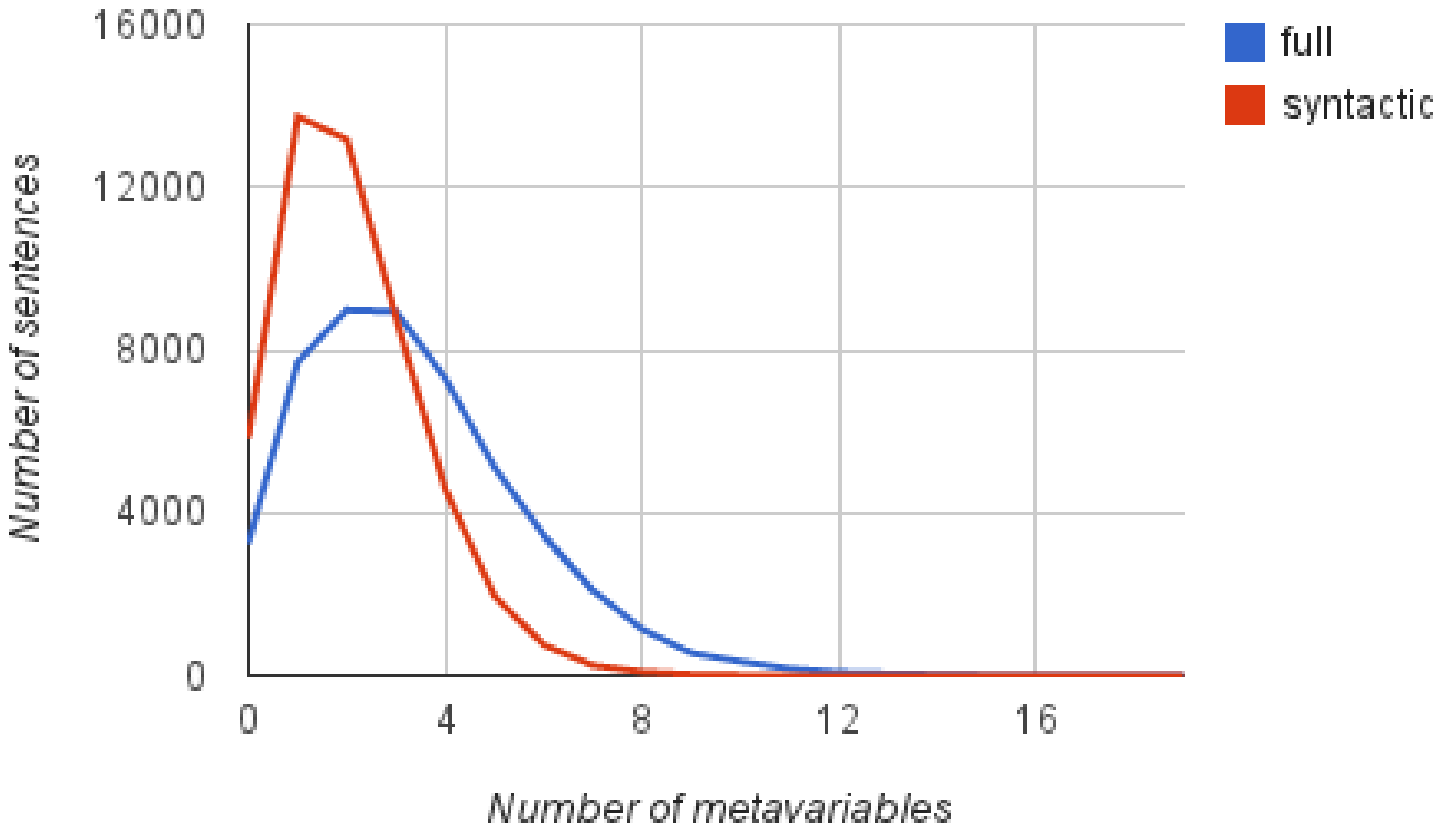
Penn Treebank:

- is **de facto standard** for training English parsers
- has **about 50 000 annotated sentences**
- the original treebank has been converted to **GF abstract trees**
  - the current coverage is 94.86% (96.81%) of the constructions

Penn Treebank



Penn Treebank



Translation Service

- An **experimental online translation service** is now available in the MOLTO translation interface
  - Currently **English** to **Bulgarian**, **Finnish**, **German** and **Hindi**
  - A **monster PGF** (39MB file / 5GB RAM)

Demo!!

Summary

- **statistical processing** in GF is a new research direction
  - the beginning of a new project and not the end
- the **translation service** is only a proof of concept
  - need for better disambiguation
  - need for better translation dictionaries
  - need dictionary of idiomatic constructions
- **scaling up**
  - in MOLTO we have promised a scale of hundreds of lemmas
  - we began scaling up to thousands, i.e. 50 0000 lemmas