GF Runtime System

Grégoire Détrez
Ramona Enache

GF is

- more expressive than CF(context free) and MCS (mildly context sensitive) grammars
- equivalent to PMCFG(parallel multiple context free grammars)
    - □ sufficient expressive power
    - □ “almost” linear parsing time

PMCFG

CFG +

- multidimensional categories(except the start category)
- productions are linear combinations of terminals and projections of arguments

PMCFG

Example : $a^n b^n c^n$, n>0

$$S \rightarrow c\ [N]$$
$$N \rightarrow s\ [N]$$
$$N \rightarrow z\ []$$

where

$c = \langle 1;1\rangle \langle 1;2\rangle\ \ \langle 1;3\rangle$

$s = \langle a \langle 1;1\rangle,\ \ b \langle 1;2\rangle,\ \ c \langle 1;3\rangle\rangle$

$z = \langle a, b, c\rangle$

$\dim(N) = 3$, $\dim(S) = 1$

PGF

- runtime binary format of GF
- equivalent to PMCFG + extra for advanced type features and metavariables
- encodes information about the grammar – abstract and concrete syntaxes and their constituents
- used for parsing and linearization
- compact representation of the grammar

PGF

- predefined types :
  - integers(signed, theoretically unbound)
  - strings(in UTF-8 format)
  - floats
  - lists

Parsing

(Angelov, 2009)

- polynomial complexity (linear empirical complexity for the resource grammars)
- incremental – for word completion and authoring of complex syntactic constructions
- idea – the grammar is extended incrementally at runtime, by approximation with a CFG grammar, by predicting possible continuations

Linearization

(Angelov, 2010)

- key phrase – loosely coupled synchronous grammar
- idea: an abstract syntax tree is linearized by top-down analysis on the constructing function and arguments

Linearization

Example

Abstract syntax :

CN → AdjN N A ;
N → song_N ;
A → beautiful_A ;

Linearization

Example

Italian concrete syntax :

CN_fem → AdjN_fem [N_fem, A]

CN_masc → AdjN_masc [N_masc, A]

N_fem → song_N_fem[]

A → beautiful_A[]

where

AdjN_fem = <<2;2><1;1>, <2;4><1;2>>

AdjN_masc = <<2;1><1;1>, <2;3><1;2>>

song_N_fem = <”canzone”, “canzone”>

beautiful_A = <”bello”, “bella”,”belli”,”belle”>

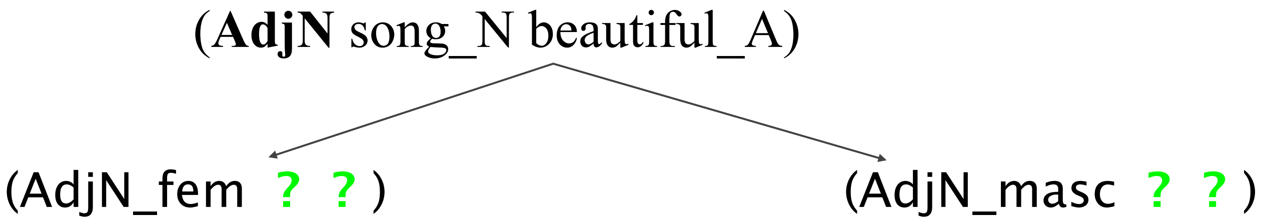
Linearization

Example

(**AdjN** song_N beautiful_A)

Linearization

Example



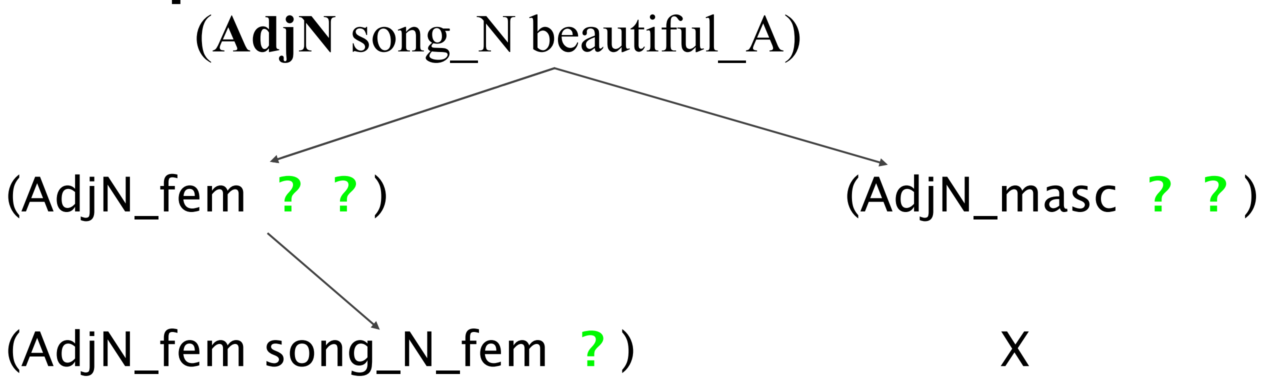
(**AdjN** song_N beautiful_A)

(AdjN_fem ? ?)

(AdjN_masc ? ?)

Linearization

Example



(**AdjN** song_N beautiful_A)

(AdjN_fem ? ?)

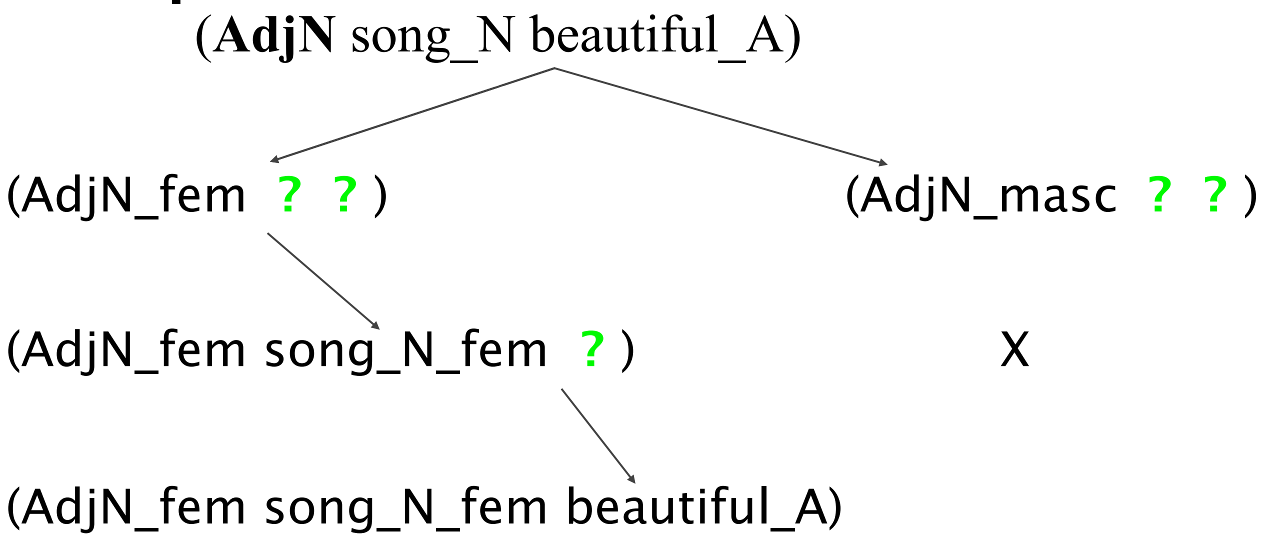
(AdjN_masc ? ?)

(AdjN_fem song_N_fem ?)

X

Linearization

Example



(**AdjN** song_N beautiful_A)

(AdjN_fem ? ?)

(AdjN_masc ? ?)

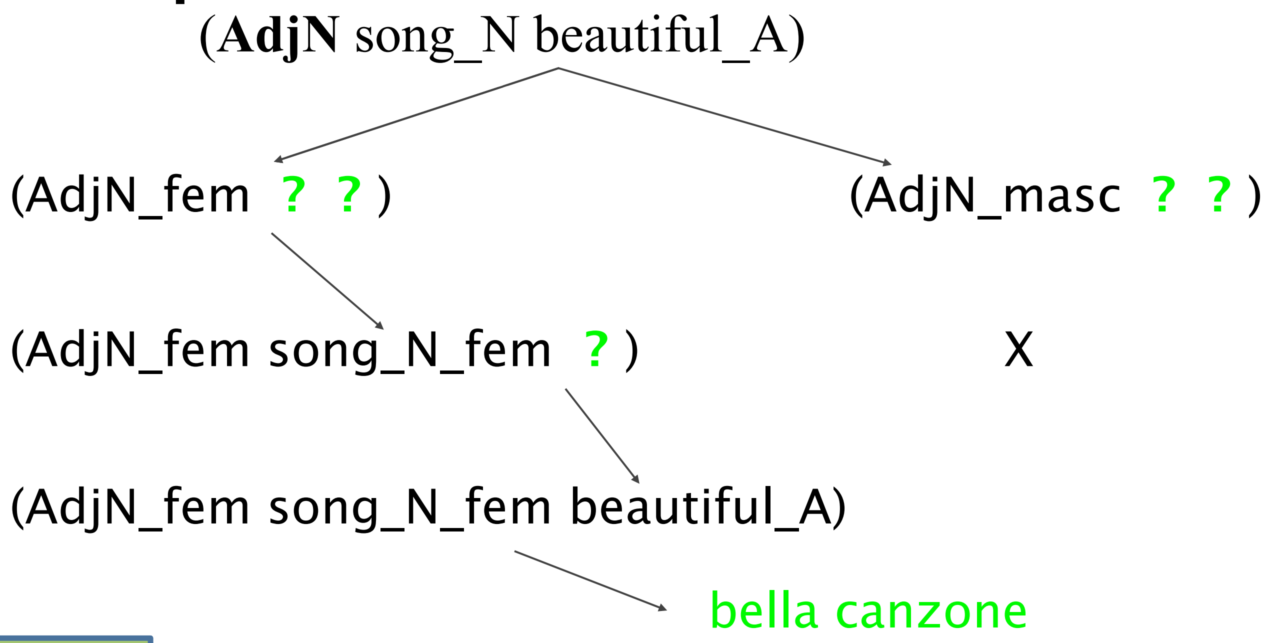
(AdjN_fem song_N_fem ?)

X

(AdjN_fem song_N_fem beautiful_A)

Linearization

Example



(**AdjN** song_N beautiful_A)

(AdjN_fem ? ?)

(AdjN_masc ? ?)

(AdjN_fem song_N_fem ?)

X

(AdjN_fem song_N_fem beautiful_A)

bella canzone

Motivation for current work

- Make GF applications run offline in specific environments - Android phones
- Make server application on Google AppEngine
- Make GF more accessible to a large number of developers and users
- Make an interpreter for PGF in an imperative language - easier to generalize to others(C, Python)
- Make GF grammars more easily embeddable in Java applications

Interpreter needed !

- existing interpreters: Haskell, JavaScript

Interpreter needed !

- existing interpreters: Haskell, JavaScript, Java (on the way)

What to do ?

- read for the PGF format and deserialize the grammar
- parse an entry with the GF predictive incremental parsing
- linearize an abstract syntax tree
- generated random constructions in the grammar
- type check GF trees

So far :

- read for the PGF format and deserialize the grammar ✓
- parse an entry with the GF predictive incremental parsing ✓
- linearize an abstract syntax tree ✓
- generated random constructions in the grammar ✍
- type check GF trees ✍

... and putting all together ✍

Inspiration :

- K. Angelov, B. Bringert and A. Ranta. PGF: A Portable Run-time Format for Type-theoretical Grammars, Journal of Logic, Language and Information, 2009.
- K. Angelov, Incremental Parsing of Parallel Multiple Context-Free Grammars. 12th Conference of the European Chapter of the Association for Computational Linguistics, 2009
- K. Angelov, Loosely Coupled Synchronous Parallel Multiple Context-Free Grammars for Machine Translation, to appear

? Questions ?