Tools for highly-inflecting languages

Apertium and Giellatekno

Francis M. Tyers

Giellatekno & CLEAR
UiT Norgga árktalaš universitehta
9018 Romsa (Norway)

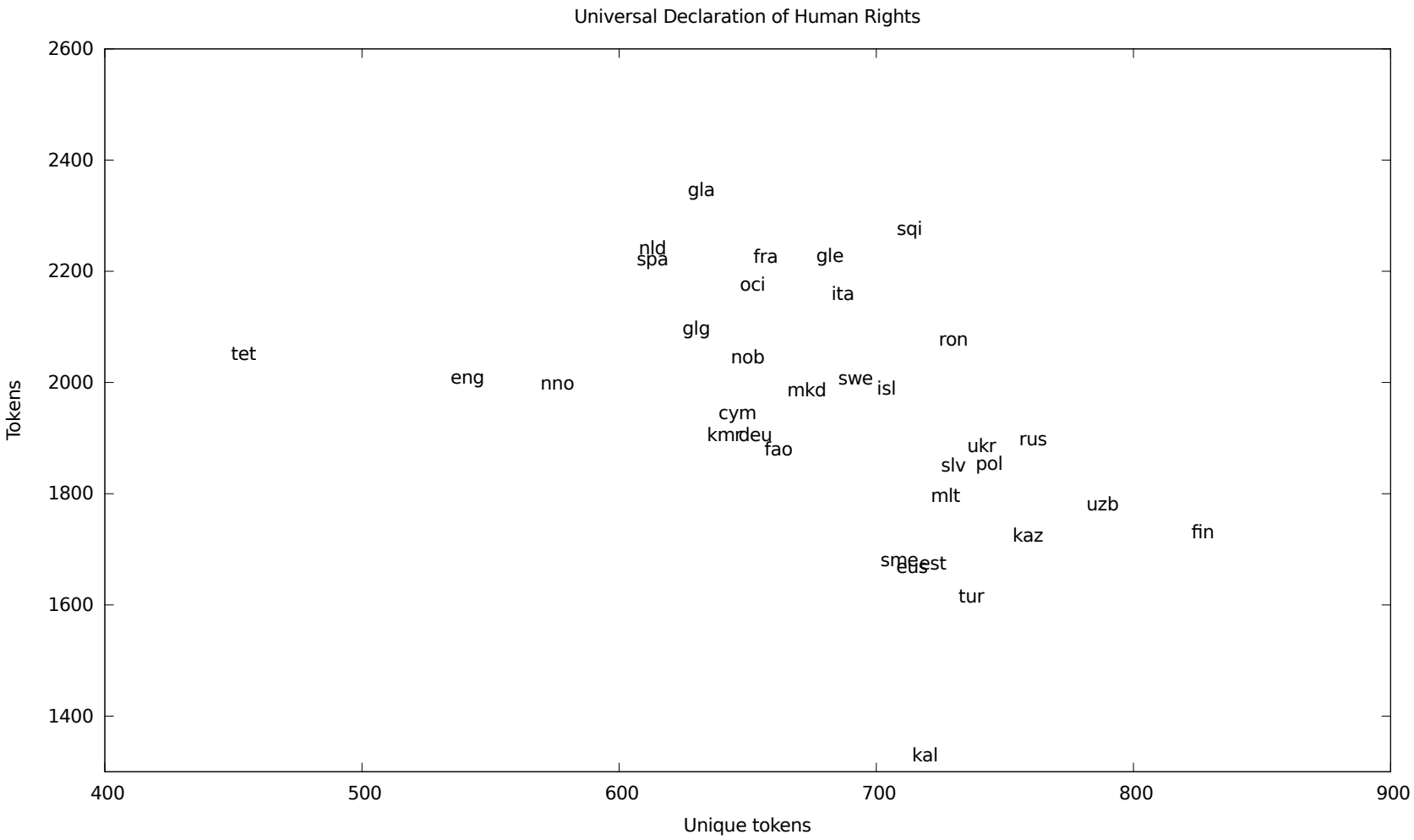
12th September, 2016

Clarifications

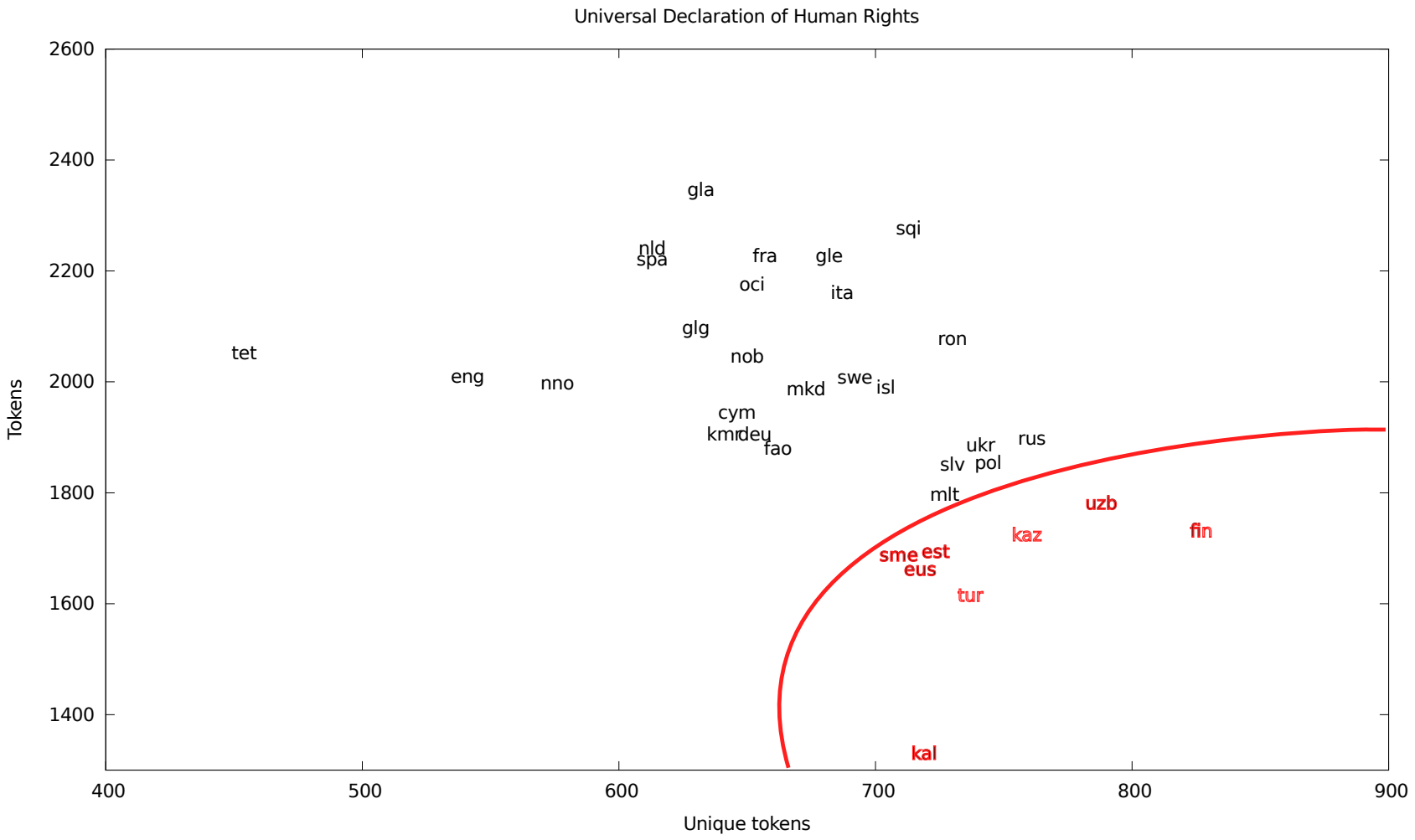
- What is “highly-inflecting” ?
  - “morphologically-rich”? $\rightarrow$ French, German, Czech ?
- Really:
  - Very productive derivation
  - Lots of syntax expressed in morphology
  - No lexemes are found[1] in all their “forms”
  - High morpheme per token ratio
  - (Compounds written without space)
- And what are tools ?
  - Finite-state models of morphology

[1] In a corpus

Highly inflecting



Highly inflecting



Apertium and Giellatekno

Apertium:

- Began as a rule-based MT project
- Now does all kinds of language technology

Giellatekno:

- Does language technology for Sámi languages
- But also languages of Russia and the High North

Commonalities:

- Primarily rule-based approach
- Finite-state transducers
- GNU `autotools`
- Constraint grammar

Languages: Apertium

Language	**Code**	**Lexemes**	**Coverage** (%)
Kazakh	`kaz`	28,109	94.5
Tuvan	`tyv`	11,682	92.7
Tatar	`tat`	12,179	91.0
Kyrgyz	`kir`	13,978	90.4
Kumyk	`kum`	4,918	90.2
Turkish†	`tur`	11,417	87.3
Avar	`ava`	4,854	86.5
Chuvash	`chv`	10,267	85.0
Uzbek	`uzb`	3,957	82.9
Karakalpak	`kaa`	5,757	81.4
Crimean Tatar	`crh`	3,466	81.8

† Maybe use TRmorph instead:
`http://coltekin.net/cagri/trmorph/`

Languages: Giellatekno

Language	**Code**	**Lexemes**	**Coverage** (%)
North Sámi	sme	145,000	>97.0
Estonian	est	43,867	95.3
Lule Sámi	smj	73,101	94.5
Ánár Sámi	smn	23,041	94.1
South Sámi	sma	88,239	93.5
Finnish†	fin	71,761	91.3
Erzya	myv	106,550	83.5
Kven	fkv	40,000	80.0
Greenlandic	kal	72,184	72.4
Moksha	mdf	59,377	61.6
Somali	som	15,837	77,9

† Maybe use OMorFI instead:
https://github.com/flammie/omorfi

Repositories

Apertium:
`https://svn.code.sf.net/p/apertium/svn/languages`

- One directory per language, `apertium-XXX`, e.g. `apertium-kaz`

Giellatekno:
`https://victorio.uit.no/langtech/trunk/`

- One directory per language, `langs/XXX`, e.g. `langs/sme`

General structure

Lexicon (`.lexc` files):

- Mappings between stem/lemma and paradigm/continuation class
- Mappings between morphemes and tags

Phonological rules (`.twol` files):[2]

- Mappings between archiphonemes and surface symbols

Analysis	**Morphotactics**	**Surface**
`машина<n><pl><abl>`	`машина>LAp>DAн`	`машиналардан`
`модель<n><pl><abl>`	`модель>LAp>DAн`	`модельдерден`
`агент<n><pl><abl>`	`агент>LAp>DAн`	`агенттерден`

[2] Or more precisely morphographemic rules

Morphological analysis and generation

Morphological analysis and generation are simple:

```
$ echo ”машиналардан” | hfst-proc -e -C kaz.automorf.hfst
”<машиналардан>”
    ”машина”    n pl abl

$ echo ”машиналардан” | hfst-proc -e kaz.automorf.hfst
^машиналардан/машина<n><pl><abl>$

$ echo ”^машина<n><pl><abl>$” | hfst-proc -g kaz.autogen.hfst
машиналардан
```

Morpheme segmentation

By composing the morphotactic transducer with the surface transducer we can get a segmenter

```
hfst-invert kaz.LR.lexc.hfst -o kaz.LR.lexc.hfst.inv
hfst-compose-intersect -2 kaz.LR.hfst -1 kaz.LR.lexc.hfst.inv
-o kaz.seg
hfst-invert kaz.seg | hfst-fst2fst -O -o kaz.segmenter

echo ”машиналарымдан” | hfst-proc kaz.segmenter
^машиналарымдан/машина>{L}{A}р>{I}м>{D}{A}н$
```

Another way of doing morphology

An alternative:

- Don't care so much about how correct the form is, or want to include it as back-off
- Use supervised training methods for generation

- But then you need data!
- There are some datasets from Wiktionary (e.g. SigMorphon)

- But you could just generate the training data using the transducer!

Generating training data

Produce a prefix of the table you want to generate:

```
$ echo ”м а ш и н а %<n%> ?* ” | hfst-regexp2fst > p.hfst
```

Compose/intersect it with the whole transducer:

```
$ zcat kaz.autogen.att.gz | hfst-txt2fst |
hfst-compose-intersect -2 - -1 p.hfst | hfst-fst2strings
машина<n><pl><px1sg><nom>:машиналарым
машина<n><pl><px1sg><loc>:машиналарымда
машина<n><pl><px1sg><abl>:машиналарымдан
...
```

Use forms to train model!

Getting help

IRC: `#_u-dep #apertium #hfst` (`irc.freenode.net`)

Wiki: `http://wiki.apertium.org`

Mailing lists: `apertium-stuff@lists.sourceforge.net`, `apertium-turkic@lists.sourceforge.net`

Private email: 1) Google name; 2) Write e-mail

At conferences: Just corner us and ask

~~**Turn up at my flat:** Rypeveg...~~

Future directions

More languages:

- We want all the languages
- But particularly:
  - Turkic languages (Apertium)
  - Uralic and the High North (Giellatekno)

Pipelines:

- Work on full-pipelines, from raw text to deep dependency analysis

Universal dependencies:

- Interchange format for analyses, tools and data

Takk! · Kiitos! · Giitu!
Teşekkürler! · Pəxmət!