Large Scale Keyword Extraction Using a Finite State Backend

Miloš Jakubíček, Pavel Šmerk

Lexical Computing
Brno, Czech Republic
milos.jakubicek@sketchengine.co.uk

NLP Centre, Masaryk University,
Brno, Czech Republic
xsmerk@fi.muni.cz

RASLAN 2016, 3. 12. 2016

Partially supported by the Czech-Norwegian Research Programme within the HaBiT Project 7F14047.

Outline

Motivation

- string operations are slow
- . . . at least slower than number operations

$\Rightarrow$ most corpus processing is performed on numbers instead of strings

$\Rightarrow$ we need to provide some string-to-number and number-to-string mappings

- . . . but sometimes this does not help, e.g. in cross-corpora comparisons such as for keyword extraction

Keyword extraction

- contrastive approach: one (sub)corpus vs. another (sub)corpus
- many algorithms available, but here: simplemath
- input one of:
  - positional attributes (word forms, lemmas, tags)
  - word sketch collocations (headword-relation-collocate triples)
  - terms (according to a term grammar)
- comparison must be performed on strings

KwEx on positional attributes

e.g. BNC vs. enTenTen13 or BNC written ws. enTenTen13

KwEx on collocations

e.g. enTenTen15 spam subcorpus vs. whole enTenTen15

KwEx on terms

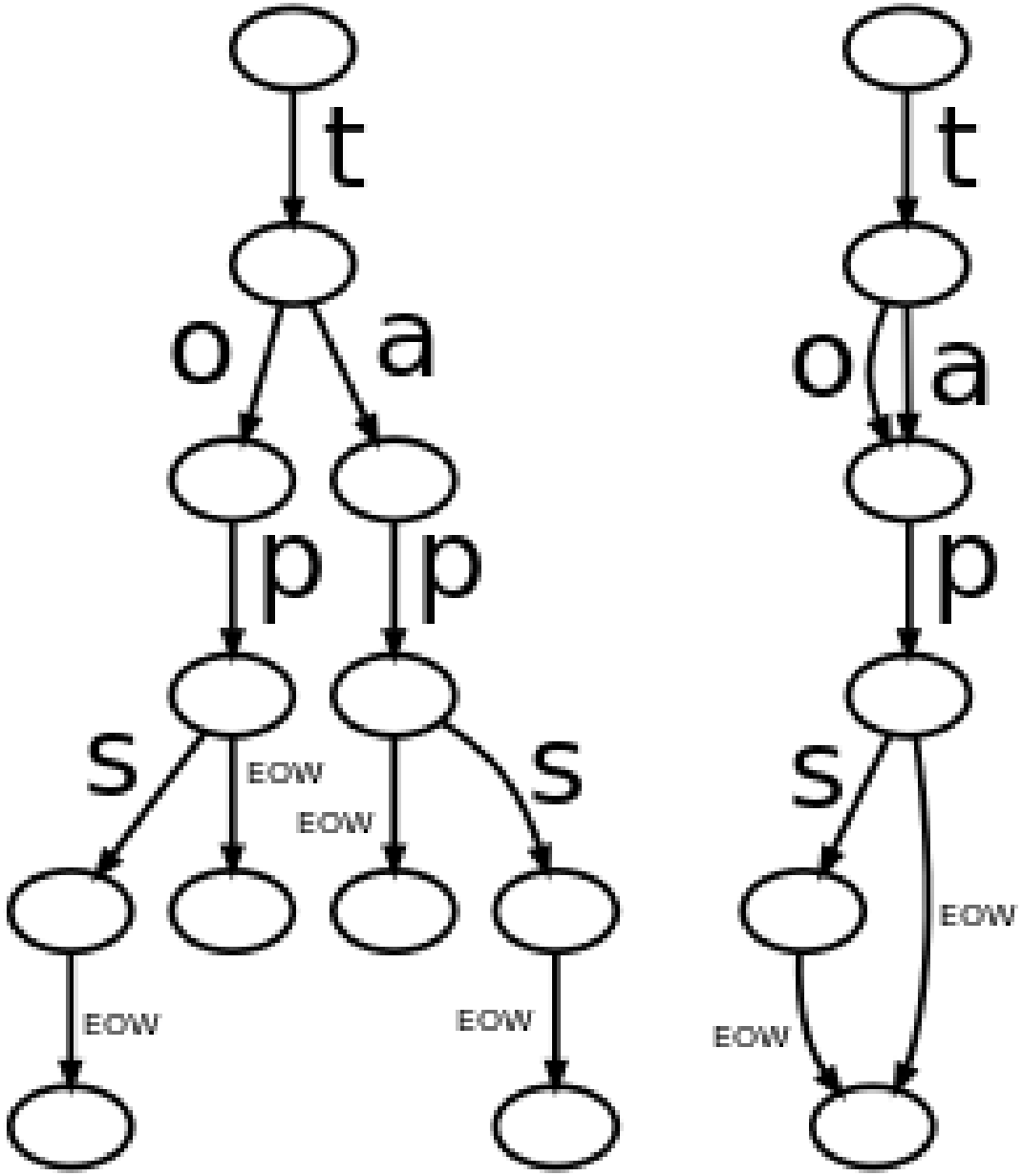
e.g. English Accounting Corpus vs. enTenTen12

Issues

- big corpora yield big data files, comparison on strings inevitably slow
- especially with cold disk cache
- current solution exploits pre-hashing strings for the case of collocations and terms, and comparing hashes instead of full strings
- still slow, also lot of preprocessing needed

Improvements

- use a minimal finite-state automaton to store strings
- benefit from easy intersection of two automata
- get: smaller data files, unified approach for any input, faster preproceessing and runtime execution
- all thanks to the FSA3 package made by Pavel Šmerk


t
o
a
p
p
s
EOW
EOW
s
EOW
EOW
t
o
a
p
s
EOW
EOW

FSA3

FSA3 is a tiny package inspired by the work of Jan Daciuk, providing

- building an FSA from (large) sorted and (with help of HAT-trie) unsorted input
- dumping an FSA
- providing number-to-string and string-to-number operations
- providing (left) intersect of two automata
- open-source, **soon** to be published at `http://corpus.tools`

Evaluation

Table: Evaluation of hash-based and FSA-based keyword extraction

string source	**corpus$_1$**	**corpus$_2$**	**FSA$_1$ size**	**FSA$_2$ size**	**page cache**	**time$_{prev}$**	**time$_{now}$**	**speedup**
lemma	BNC	enTenTen15	556k items 4 MB	26,426k items 340 MB	cold	80.2s	51.4s	1.56x
					hot	6.3s	0.7s	9x
term list	Brown family	enTenTen12	320k items 4 MB	164,189k items 2 GB	cold	1m11s	2m10.2s	0.54x
					hot	4s	1.2s	3.3x

Conclusions

- FSA represent a great structure for storing strings
- . . . if you have good tools to work with them
- keyword and term extraction made simpler and faster thanks to FSA3
- many other related changes in SkE to come

Conclusions...continued