Linguistic querying and quantitative analysis of large corpora

Prof. Dr. Stefan Evert
Corpus Linguistics Group
FAU Erlangen-Nürnberg
http://www.linguistik.fau.de/

CQLF Metamodel (ISO draft)
- Corpus query lingua franca (CQLF)
- ISO working document
- CQLF Part 1: metamodel of QLs

Examples of QLs
- Level 1 (linear)
  - CQP (IMS Open Corpus Workbench)
  - Manatee / SketchEngine, Poliqarp, ...
- Level 2 (hierarchical)
  - TigerSearch, tgrep, ... (2a: trees)
  - treebank.info (2b: dependency)
- Level 3 (concurrent) + 2 (hierarchical)
  - NQL (Nite XML Toolkit)
  - AnnisQL

A slightly biased view …
IMS Open Corpus Workbench

- Original development 1993–1996 (IMS Stuttgart)
  - research project on text corpora & exploration tools
    (also produced TreeTagger)
  - applications: statistical NLP, lexicography, corpus linguistics
  - Unix-based, ISO-8859-1 character set, up to ca. 200M words
- Maintenance at IMS 1998–2004
- IMS Open Corpus Workbench 2005 (GPL licence)
- Current: CWB 3.5 to be released soon
  - UTF-8 / ISO-8859, 64-bit (up to 2G words), Windows support
  - full backward compatibility

![CWB data model](http://cwb.sf.net/)

### CWB data model

<table>
<thead>
<tr>
<th>#</th>
<th>word</th>
<th>pos</th>
<th>lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>DET</td>
<td>a</td>
</tr>
<tr>
<td>1</td>
<td>fine</td>
<td>ADJ</td>
<td>fine</td>
</tr>
<tr>
<td>2</td>
<td>example</td>
<td>NN</td>
<td>example</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>PUN</td>
<td>.</td>
</tr>
<tr>
<td>4</td>
<td>Very</td>
<td>ADV</td>
<td>very</td>
</tr>
<tr>
<td>5</td>
<td>fine</td>
<td>ADJ</td>
<td>fine</td>
</tr>
<tr>
<td>6</td>
<td>examples</td>
<td>NN</td>
<td>example</td>
</tr>
<tr>
<td>7</td>
<td>!</td>
<td>PUN</td>
<td>!</td>
</tr>
</tbody>
</table>

### CWB data model

A fine example. Very fine examples!

```xml
<text id="42" lang="English">
<s>
A/DET/a fine/ADJ/fine example/NN/example ./PUN/.
</s>
<s>
Very/ADV/very fine/ADJ/fine examples/NN/example !/PUN/!
</s>
</text>
```

XML tags inserted as "invisible" tokens

<table>
<thead>
<tr>
<th>#</th>
<th>word</th>
<th>pos</th>
<th>lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0)</td>
<td>&lt;text id=&quot;42&quot; lang=&quot;English&quot;&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(0)</td>
<td>&lt;s&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>A</td>
<td>DET</td>
<td>a</td>
</tr>
<tr>
<td>1</td>
<td>fine</td>
<td>ADJ</td>
<td>fine</td>
</tr>
<tr>
<td>2</td>
<td>example</td>
<td>NN</td>
<td>example</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>PUN</td>
<td>.</td>
</tr>
<tr>
<td>(3)</td>
<td>&lt;/s&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4)</td>
<td>&lt;s&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Very</td>
<td>ADV</td>
<td>very</td>
</tr>
<tr>
<td>5</td>
<td>fine</td>
<td>ADJ</td>
<td>fine</td>
</tr>
<tr>
<td>6</td>
<td>examples</td>
<td>NN</td>
<td>example</td>
</tr>
<tr>
<td>7</td>
<td>!</td>
<td>PUN</td>
<td>!</td>
</tr>
<tr>
<td>(7)</td>
<td>&lt;/s&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(7)</td>
<td>&lt;/text&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Corpus position ("cpos")
CWB data model

<table>
<thead>
<tr>
<th>#</th>
<th>word</th>
<th>pos</th>
<th>lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0)</td>
<td>&lt;text id=&quot;42&quot; lang=&quot;English&quot;&gt;</td>
<td>&lt;s&gt;</td>
<td></td>
</tr>
<tr>
<td>(0)</td>
<td></td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>fine</td>
<td>ADJ</td>
<td>fine</td>
</tr>
<tr>
<td>2</td>
<td>example</td>
<td>NN</td>
<td>example</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>PUN</td>
<td>.</td>
</tr>
<tr>
<td>(3)</td>
<td>&lt;/s&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4)</td>
<td>&lt;s&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Very</td>
<td>ADV</td>
<td>very</td>
</tr>
<tr>
<td>5</td>
<td>fine</td>
<td>ADJ</td>
<td>fine</td>
</tr>
<tr>
<td>6</td>
<td>examples</td>
<td>NN</td>
<td>example</td>
</tr>
<tr>
<td>7</td>
<td>!</td>
<td>PUN</td>
<td>!</td>
</tr>
<tr>
<td>(7)</td>
<td>&lt;/s&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(7)</td>
<td>&lt;/text&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

XML regions represented internally as ranges of tokens, i.e. start/end #

CWB software architecture

- Static column store
  - independent p-attributes can be added/deleted
  - token sequence can't be changed/extended after encoding
- Inverted index for value lookup
  - retrieve all occurrences of (set of) annotation string(s)
- Aggressive data compression
  - bit streams using Huffman / Golomb coding
- Query language based on regular expressions
  - at character level for annotation strings
  - query = regular expression over tokens

lexicon IDs for annotation strings (per column)

CQP

original contribution

Witten/Moffat/Bell (1999)
Managing Gigabytes. 2nd ed.
CWB software architecture

CWB inverted index (p-attribute)

# | word | pos | lemma
---|------|-----|------
0 | <text id="42" lang="English"> |
0 | <s> |
1 | A | DET | 0 | a | 0 |
1 | fine | ADJ | 1 | fine | 1 |
2 | example | NN | 2 | example | 2 |
3 | . | PUN | 3 | . | 3 |
(3) | </s> |
(4) | <s> |
4 | Very | ADV | 4 | very | 4 |
5 | fine | ADJ | 1 | fine | 1 |
6 | examples | NN | 2 | example | 2 |
7 | ! | PUN | 3 | ! | 5 |
(7) | </s> |
(7) | </text> |

CWB inverted index (p-attribute)

<table>
<thead>
<tr>
<th>cpos</th>
<th>tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>id</th>
<th>lexicon</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DET</td>
</tr>
<tr>
<td>1</td>
<td>ADJ</td>
</tr>
<tr>
<td>2</td>
<td>NN</td>
</tr>
<tr>
<td>3</td>
<td>PUN</td>
</tr>
<tr>
<td>4</td>
<td>ADV</td>
</tr>
<tr>
<td>5</td>
<td>...</td>
</tr>
<tr>
<td>6</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>id</th>
<th>freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>id</th>
<th>occurrences (cpos)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0, ...</td>
</tr>
<tr>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>2, 6, ...</td>
</tr>
<tr>
<td>3</td>
<td>3, 7, ...</td>
</tr>
<tr>
<td>4</td>
<td>4, ...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

CWB storage of s-attributes

<table>
<thead>
<tr>
<th>&lt;s&gt;</th>
<th>start</th>
<th>end</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>&lt;text&gt;</th>
<th>start</th>
<th>end</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

annotation

id="42" lang="English"
Huffman coding (token sequence)

- Huffman code = optimal compression for independent coding of individual tokens with static codebook
  - similar to Morse code: use short bit patterns for frequent items
- Sample Huffman codes for part-of-speech tags
  - NN  110  JJ  0100
  - IN  111  JJR  000000100
  - DT  101  JJS  0000000001
  - PP  1001
  - VB  00111
- Encoding of POS tags for *go to the prettiest beach*:
  - 0 0 1 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0 1 1 0

Data compression rates

Typical data sizes of p-attributes for 100 M word corpus
- Plain text: ca. 400-600 MB
- Uncompressed attribute: ca. 800 MB (including all index & lexicon files)
- Word form, lemma, etc.: ca. 320-360 MB
- POS, morphological features: ca. 100-150 MB
- Binary attribute: ca. 50 MB

Golomb coding (inverted index)

- Encode distances between occurrences of lexicon entry
  - assumption: occurrences randomly distributed across corpus
- Golomb codes = mixed unary/binary representation
  - fixed size of binary part = average distance value
  - optimal for random distribution, good worst-case bounds
- Golomb code example:
  - distance to next occurrence = 26 tokens
  - e.g. 3-bit binary representation: 26 = 3 * 8 + 2
  - Golomb code: 000 1 010

CWB software architecture
**Regular expressions**

- **Generalisations over characters**
  - arbitrary character: . ("matchall")
  - character ranges, e.g. [A-Z], [aeiou]

- **Alternatives**: (...) | (...)
  - e.g. (over|under)us(es|es|ed|ing)

- **Repetition of subexpression**
  - optionality ( .. )? (0 or 1)
  - Kleene star ( .. )* (0 or more); ( .. )+ (1 or more)
  - repetition count ( .. ){4} (exactly 4)
  - ( .. ){1,10} (between 1 and 10); ( .. ){5,} (at least 5)

**CQP query language**

- **Regular expressions for words and annotations**
  - "([A-Z0-9]+)+[a-z]+"
  - \( \rightarrow 24\text{-}hour, A\text{-}levels, MS\text{-}DOS\text{-}compatible, ... \)

- **Boolean combination of annotations**
  - \([\text{pos} = \text{"RB"}] \& \text{(word} \neq \text{".*ly"%c})\]
  - adverbs that don't end in -ly
  - ".+ly" \( \Leftrightarrow [\text{word} = \text{".+ly"}]\)

---

**CQP query language**

- **Regular expressions over token descriptions allow search for flexible lexico-grammatical patterns**
  - \([\text{lemma} = \text{"give"}] [\text{pos} = \text{"PRP"}] \text{"the"}\)
  - \[\text{pos} = \text{"JJ.*"}] [\text{pos} = \text{"N.*"}]\]
  - gives me the creeps, give him the grand tour, ...

- **Implementation as finite-state automaton**

---

**Query optimization problems**

\([\text{pos} = \text{"DT"}]? [\text{pos} = \text{"JJ.*|RB"}] + [\text{pos} = \text{"NN.*"}]\)
\[\text{pos} = \text{"PP.*"}]\]
\[\text{pos} = \text{"VB.*"}]* [\text{lemma} = \text{"dance"} \& \text{pos} = \text{"VB.*"}]\)

---

possible start positions looked up in inverted index
Query optimization problems

[ pos="DT" ] [ pos="JJ.*" ]*
("cleverest" | "most" "intelligent")
[ pos="NN.*" ]

Corpuscle search engine attempts to find most efficient
"cut" of FSA graph

neither search direction allows efficient index lookup

Matching strategy for repetitions

Pattern: DET? ADJ* NN (PREP DET? ADJ* NN)*

the old book on the table in the room

This is can be useful for the extraction of cooccurrence data, e.g.

[ pos="ADJ" ] [ ]{0,5}, [ pos="NN" ]

Matching strategy for repetitions

Pattern: DET? ADJ* NN (PREP DET? ADJ* NN)*

the old book on the table in the room

> set MatchingStrategy "traditional";

Matching strategy for repetitions

Pattern: DET? ADJ* NN (PREP DET? ADJ* NN)*

the old book on the table in the room

> set MatchingStrategy "shortest";
Matching strategy for repetitions

Pattern: DET? ADJ* NN (PREP DET? ADJ* NN)*

the old book on the table in the room

the old book on the table in the room

MatchingStrategy "longest";

Quantitative analysis

- Frequency breakdown
  - count matching strings (variable length)
  - based on in-memory or disk-based sort algorithm
- Metadata distribution
  - counts for fixed set of categories
- Frequency counts
  - frequency counts for (combinations of) marked tokens
  - uses associative array (hash)
  - disk-based merge of sorted partial results for large data

→ various statistical analyses can be applied to frequency data (frequency comparison, keywords, type-token distributions, ...)
Quantitative analysis

- **Co-occurrence frequency**
  - contingency table
  - within span of \( n \) words
  - within sentence, article, ...
  - syntactic dependency
- **Multivariate analysis**
  - multiple linguistic features for each text (rel. freq.)
  - experimental support in CQPweb
- **Per-text frequency data, n-grams, patterns, ...**

→ various statistical analyses applied to frequency data (collocation identification, latent dimensions, regression, ...)

### Beyond CQLF level 1

- Large corpora with rich annotation becoming available
  - XML text structure & metadata (e.g. TEI)
  - constituency structure or syntactic dependencies
  - parallel corpora with word or phrase alignment
  - multi-layer annotation (intersecting hierarchies, time-aligned)
- Extended data model & indexing requirements
  - support for non-tabular data structures
- Regular query languages not sufficient
  - target structures need not be linear sequences

### The NXT Object Model

(Evert et al. 2003)

- Token sequence
- Constituency (tree)
- Pointers (graph)
- Intersecting hierarchies
- Time-aligned layers

### The NXT Object Model

(Evert et al. 2003)

- Semantics.xml
- Syntax.xml
- Tokens.xml
- Gtypes.xml
- Gestures.xml
NXT Query Language

- \((s\ syntax): s@cat == "PP"\)
- \((s1\ syntax) (s2\ syntax): \ s1@cat == \ s2@cat\)

- \((w\ word): s ^ w\)
- \((a)\ (b): a ^ 1 \ b\)

- \((s\ syntax) (t\ word|syntax): \ s > \ t\)
- \((g\ gesture) (exists \ t\ gtype): \ (s > \ t) \ \&\ \& \ (t@name == "deictic")\)

- \((g1\ gesture) (g2\ gesture): \ g1@hand == "left" \ \&\ \& \ g2@hand == "right" \ \&\ \& \ g1 \ # \ g2\)
Query languages for CQLF L2 + L3

- Declared anchor elements + constraints / relations
  - examples: NQL, TigerSearch, AnnisQL, ...
- Simple implementation: nested loops
  - can be extremely slow
  - index-based optimization similar to SQL
  - most engines still not fast enough for large corpora
- Naïve queries can have excessive match counts
  - \((s, s), (w_1, \text{word}), (w_2, \text{word}): s \land w_1 \land s \land w_2\)
  - ca. 8.5M matches in Tiger Treebank (800k words)
- Cannot search for variable-length patterns (≠ CQP)

Treebank.info

(Proisl & Uhrig 2012)

- Dedicated query engine for dependency graphs
- Translate graph fragment into SQL
  - MySQL, PostgreSQL
- Use specialized graph database
  - Neo4j („the world’s leading graph DB“)
- Roll your own: CWB-treebank
  - store dependency graph in CWB corpus
  - CQP query + Perl code for subgraph matching
Quantitative analysis for CQLF L2+L3

- Trivial: frequency counts, metadata distribution, ...
- Temporal association
- Complex contingencies
  - methods from data mining community?
- Co-occurrence of tree or graph fragments
  - harder than it looks
  - ongoing PhD project by Thomas Proisl
Problem: combinatorial matches

Outlook: CWB 4

- Data model: sequential annotation layers
  - layer = sequence of annotation units (tabular format)
  - base layer represents tokenized text
  - segmentation layer, XML layer, graph layer link annotation units to their base layer (units or ranges)
- Indexing engine: similar to CWB 3
  - sort index, inverted index; compressed byte-streams
- CQP4 query language (speculative)
  - tree of regular patterns with index lookup for root
  - multiple axes: precedence, dominance, dependency graph