Common question answering (QA) systems are based on the extraction of answers from large document collections. The task of the IR component in QA is to retrieve relevant segments in order to reduce the search space. The performance (especially in terms of recall) of this component is crucial for such QA systems. We compared seven off-the-shelf IR engines using the test set from the CLEF 2003 competition on Dutch question answering.

## Open source IR engines

**Amberfish:** [http://www.etymon.com/tr.html](http://www.etymon.com/tr.html)  
GPL, C/C++, plain text, semi-structured/XML (with nested fields), wildcard search, phrase search, boolean queries, relevance ranking

**Lucene:** [http://jakarta.apache.org/lucene/docs/index.html](http://jakarta.apache.org/lucene/docs/index.html)  
Apache License, Java, plain/semi-structured documents, snowball stemmers, phrase search, boolean queries, relevance ranking

GPL, C, plaintext, images, boolean or ranked queries

**Swish-e:** [http://swish-e.org/](http://swish-e.org/)  
GPL, C, plain/semi-structured documents, snowball stemmers, wildcard search, phrase search, fuzzy search (soundex, metaphone), flexible configuration (input/output, tokenisation etc.), boolean queries, relevance ranking, Perl bindings

**Xapian:** [http://www.xapian.org/](http://www.xapian.org/)  
GPL, C++, plain text, snowball stemmers, phrase search, proximity search, relevance feedback, wide range of boolean operators, relevance ranking, Perl/SWIG bindings

**Zebra:** [http://www.indexdata.dk/zebra/](http://www.indexdata.dk/zebra/)  
GPL, C, structured (XML), phrase search, boolean queries, relevance ranking, wild-card search, Z39.50 protocol, client-server implementation

BSD-style license, C, plain, semi-structured (TREC), phrase search, boolean queries, relevance ranking, summary function

## IR results (CLEF 2003 data, 200 retrieved documents)

<table>
<thead>
<tr>
<th>Engine</th>
<th>MRR (in%)</th>
<th>Documents</th>
<th>Paragraphs</th>
<th>Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swish-e</td>
<td>26.02</td>
<td>54.01</td>
<td>28.86</td>
<td>43.52</td>
</tr>
<tr>
<td>Zettair</td>
<td>32.10</td>
<td>52.69</td>
<td>29.90</td>
<td>42.09</td>
</tr>
<tr>
<td>Xapian</td>
<td>28.25</td>
<td>50.49</td>
<td>30.11</td>
<td>41.41</td>
</tr>
<tr>
<td>Zebra</td>
<td>26.50</td>
<td>45.06</td>
<td>27.79</td>
<td>37.53</td>
</tr>
<tr>
<td>Lucene</td>
<td>29.74</td>
<td>47.87</td>
<td>30.14</td>
<td>36.48</td>
</tr>
<tr>
<td>Amberfish</td>
<td>21.05</td>
<td>44.31</td>
<td>20.67</td>
<td>28.05</td>
</tr>
<tr>
<td>MG</td>
<td>20.86</td>
<td>39.98</td>
<td>20.98</td>
<td>22.53</td>
</tr>
</tbody>
</table>

Evaluation methodology

Performance is measured in terms of mean reciprocal ranks (MRR).

$$MRR = \frac{1}{x} \sum_{i=1}^{x} \frac{1}{\text{rank}(\text{first answer})}$$

Two types of scores are distinguished: document MRR and answer MRR:  
- **doc MRR:** mean reciprocal rank of relevant documents retrieved, i.e. documents listed in the gold standard  
- **answer MRR:** mean reciprocal rank of relevant answers retrieved, i.e. documents which include the answer string

## Conclusions

- QA may gain a lot from appropriate IR  
- there is large performance differences between open-source IR engines  
- IR performance is not (always) correlated to QA performance

## Future Work

- NLP in IR (compound analysis, dependency relations, multi-word-units/phrases)  
- IR voting  
- different IR engines  
- different index types  
- parameter optimisation