StaticIndexPruning

Static Index Pruning

Static index pruning is a method of reducing the size of the index and increasing the search performance by removing in advance certain terms and certain posting data, while retaining good enough quality of top-N results.

"Static" refers to the fact that this pruning is done in advance, ie. not during the query execution. There are also methods of "dynamic pruning" described in the literature, and they refer to pruning that is applied during query execution, to reduce the number of postings / terms examined when collecting top-N results.

It would be nice not to put this superfluous information in the index in the first place, but most of the time this is not possible, because most static pruning techniques rely also on some corpus-wide statistics.

LUCENE-1812 JIRA issue is a patch that implements this static pruning that works on existing Lucene indexes. pruning.pdf are the slides of a presentation I made during Lucene Meet{{`Up}}{{ at Apache}}`Con` 09 that contain more details on this. The implementation of static pruning in LUCENE-1812 does not require any changes to the Lucene core.

Applications

Multi-tiered search

One application that is mentioned in the slides above is a multi-tiered search. In this scenario we start with a single master index that contains all postings. From this full index we produce a first-tier heavily pruned index that fits completely in RAM. We can also produce a 2nd-tier index that we can put on a solid-state disk. And then we have the final 3rd-tier full index with all postings.

The 1st-tier index is optimized towards the most common queries, and results coming from this index are checked whether they satisfy certain correctnes metrics (e.g. the minimum number of results). If they don't, the query is forwarded to the 2nd-tier index, and then to the 3rd-tier index. This way for a vast majority of queries we can produce answers quickly and with minimum effort, while still being able to satisfy rare queries with the full index.