Apache Lucene - Resources - Performance Benchmarks

Kelvin Tan

Table of contents

1. Performance Benchmarks

The purpose of these user-submitted performance figures is to give current and potential users of Lucene a sense of how well Lucene scales. If the requirements for an upcoming project is similar to an existing benchmark, you will also have something to work with when designing the system architecture for the application.

If you've conducted performance tests with Lucene, we'd appreciate if you can submit these figures for display on this page. Post these figures to the lucene-user mailing list using this template.

2. Benchmark Variables

3. User-submitted Benchmarks

These benchmarks have been kindly submitted by Lucene users for reference purposes.

We make NO guarantees regarding their accuracy or validity.

We strongly recommend you conduct your own performance benchmarks before deciding on a particular hardware/software setup (and hopefully submit these figures to us).

3.1. Hamish Carpenter's benchmarks

Hamish can be contacted at hamish at catalyst.net.nz.

3.2. Justin Greene's benchmarks

Justin can be contacted at tvxh-lw4x at spamex.com.

3.3. Daniel Armbrust's benchmarks

My disclaimer is that this is a very poor "Benchmark". It was not done for raw speed, nor was the total index built in one shot. The index was created on several different machines (all with these specs, or very similar), with each machine indexing batches of 500,000 to 1 million documents per batch. Each of these small indexes was then moved to a much larger drive, where they were all merged together into a big index. This process was done manually, over the course of several months, as the sources became available.

Daniel can be contacted at Armbrust.Daniel at mayo.edu.

3.4. Geoffrey Peddle's benchmarks

I'm doing a technical evaluation of search engines for Ariba, an enterprise application software company. I compared Lucene to a commercial C language based search engine which I'll refer to as vendor A. Overall Lucene's performance was similar to vendor A and met our application's requirements. I've summarized our results below.

Search scalability:
We ran a set of 16 queries in a single thread for 20 iterations. We report below the times for the last 15 iterations (ie after the system was warmed up). The 4 sets of results below are for indexes with between 50,000 documents to 600,000 documents. Although the times for Lucene grew faster with document count than vendor A they were comparable.

50K documents Lucene 5.2 seconds A 7.2 200K Lucene 15.3 A 15.2 400K Lucene 28.2 A 25.5 600K Lucene 41 A 33

Individual Query times:
Total query times are very similar between the 2 systems but there were larger differences when you looked at individual queries.

For simple queries with small result sets Vendor A was consistently faster than Lucene. For example a single query might take vendor A 32 thousands of a second and Lucene 64 thousands of a second. Both times are however well within acceptable response times for our application.

For simple queries with large result sets Vendor A was consistently slower than Lucene. For example a single query might take vendor A 300 thousands of a second and Lucene 200 thousands of a second. For more complex queries of the form (term1 or term2 or term3) AND (term4 or term5 or term6) AND (term7 or term8) the results were more divergent. For queries with small result sets Vendor A generally had very short response times and sometimes Lucene had significantly larger response times. For example Vendor A might take 16 thousands of a second and Lucene might take 156. I do not consider it to be the case that Lucene's response time grew unexpectedly but rather that Vendor A appeared to be taking advantage of an optimization which Lucene didn't have. (I believe there's been discussions on the dev mailing list on complex queries of this sort.)

Index Size:
For our test data the size of both indexes grew linearly with the number of documents. Note that these sizes are compact sizes, not maximum size during index loading. The numbers below are from running du -k in the directory containing the index data. The larger number's below for Vendor A may be because it supports additional functionality not available in Lucene. I think it's the constant rate of growth rather than the absolute amount which is more important.

50K documents Lucene 45516 K A 63921 200K Lucene 171565 A 228370 400K Lucene 345717 A 457843 600K Lucene 511338 A 684913

Indexing Times:
These times are for reading the documents from our database, processing them, inserting them into the document search product and index compacting. Our data has a large number of fields/attributes. For this test I restricted Lucene to 24 attributes to reduce the number of files created. Doing this I was able to specify a merge width for Lucene of 60. I found in general that Lucene indexing performance to be very sensitive to changes in the merge width. Note also that our application does a full compaction after inserting every 20,000 documents. These times are just within our acceptable limits but we are interested in alternatives to increase Lucene's performance in this area.

600K documents Lucene 81 minutes A 34 minutes

(I don't have accurate results for all sizes on this measure but believe that the indexing time for both solutions grew essentially linearly with size. The time to compact the index generally grew with index size but it's a small percent of overall time at these sizes.)