HardwareRequirements

Hardware Requirements

In general, fetching and database updates require lots of disk, and searching is faster with more RAM. But the particulars depend on how big of an index you're trying to build and how much query traffic you expect.

Requirements for indexing

As a general rule, each page fetched requires around 10k of disk overall (for the page cache, its text, the index, db entries, etc.). So a terabyte of storage is required for every 100M pages.

Requirements for searching

If your peak traffic is only around 1 query/second, then you can probably stand to have 20 or more million pages per node. But if your peak traffic is higher, then things will be more cost-effective if you can fit the major index strutures in RAM. These require around 2kB/document, so a search node with 4GB of RAM can handle around 2M documents at around 20 queries/second.

For optimal performance on a search node you should merge the desired set of segments into a single index (after first performing duplicate elimination). You can then look at the size of the .f* files in the index to see how many documents are in the index.

Question and answers

Question 1

We are building out our new rack and we have 5 servers and about 2 terrabytes of disk space right now. We have a core xeon with 1.4 terrabytes that we are using right now and the rest are p4's with 200 gigs a piece and 1 gig of memory. Is there an optimial way to configure this?

Answer 1

With your current configuration, you could use the xeon for fetching and database work. Then copy sets of segments, with merged indexes, to your five xeon servers. With this hardware you should be able to easily maintain a 100+M page collection that can serve several searches per second.

More examples

Example 1

On a linux (kernel 2.4) computer with 1 GB RAM and 900 MHz we are able to run up to 200 fetcher.threads.fetch, fetching together with 5 MBits/s.

Example 2

On a linux (kernel 2.4) computer with 512 MB RAM and 2.4 GHz we decreased fetcher.threads.output down to "2". Seems that we get for each fetcher.thread.output a system load of a little bit more then one. This system is fetching with 25 fetcher.threads.fetch and around 4 MBits/s.