Large Scale Indexing

- Key decision in block merge indexing is block size
- In practice, spidering often interlaced with indexing
- Spidering bottlenecked by WAN speed and other factors
Single-Pass In-Memory Indexing
Overview

- Introduction
- Hardware
- BSBI - Block sort-based indexing
- SPIMI - Single Pass in-memory indexing
- Distributed indexing
- Dynamic indexing
- Miscellaneous topics
Distributed Indexing

• Web-scale indexing
  • Must use a distributed computing cluster
  • “Cloud computing”
• Individual machines are fault-prone
  • They slow down unpredictably or fail
    • Automatic maintenance
    • Software bugs
    • Transient network conditions
    • A truck crashing into the pole outside
    • Hardware fatigue and then failure
Distributed Indexing - Architecture

- The design of Google’s indexing as of 2004
Distributed Indexing - Architecture

- Use two classes of parallel tasks
  - Parsing
  - Inverting
- Corpus is split broken into **splits**
  - Each split is a subset of documents
  - analogous to distributed crawling
- Master assigns a split to an idle machine
  - Parser will read a document and output \((t,d)\) pairs
  - Inverter will sort and write postings
• Use an instance of MapReduce
  • An general architecture for distributed computing
  • Manages interactions among clusters of
    • cheap commodity compute servers
    • aka nodes
  • Uses Key-Value pairs as primary object of computation
• Use an instance of MapReduce
  • There is a map phase
    • This takes splits and makes key-value pairs
    • this is the “parse” phase of BSBI and SPIMI
  • The map phase writes intermediate files
  • Results are bucketed into R buckets
  • There is a reduce phase
    • This is the “invert” phase of BSBI and SPIMI
  • There are R inverters
Distributed Indexing - Architecture

Corpus

Parsers

Inverters

Postings

A-F
G-P
Q-Z

A-F
G-P
Q-Z

A-F
G-P
Q-Z

A-F
G-P
Q-Z

A-F
G-P
Q-Z

A-F
G-P
Q-Z

A-F
G-P
Q-Z

A-F
G-P
Q-Z

A-F
G-P
Q-Z
Parsers and Inverters are not separate machines
- They are both assigned from a pool
- It is separate software

Intermediate files are stored on a local disk
- Part of the “invert” task is to talk to the parser machine and get the data. (master coordinates)

MapReduce has different architectures for different data manipulation tasks besides this one.
Overview

- Introduction
- Hardware
- BSBI - Block sort-based indexing
- SPIMI - Single Pass in-memory indexing
- Distributed indexing
- Dynamic indexing
- Miscellaneous topics
Dynamic Indexing

- Documents come in over time
  - Postings need to be updated for terms already in dictionary
  - New terms need to get added to dictionary
- Documents go away
  - Get deleted, etc.
Dynamic Indexing

- Overview of solution
  - Maintain your “big” main index on disk
    - (or distributed disk)
  - Continuous crawling creates “small” indices in memory
  - Search queries are applied to both
    - Results merged
Dynamic Indexing

- Overview of solution
  - Document deletions
    - Invalidation bit for deleted documents
    - Just like contextual filtering,
      - results are filtered to remove invalidated docs
      - according to bit vector.
    - Periodically merge “small” index into “big” index.
Dynamic Indexing

Inverters

Small Indices

Big Indices

Invalidation Bits

Query

Filter

Result
Issues with big *and* small indexes

- Corpus wide statistics are hard to maintain
  - Typical solution is to ignore small indices when computing stats
- Frequent merges required
- Poor performance during merge
  - unless well engineered
- Logarithmic merging
Got to about slide 17 of cons.pdf
And image cons18.eps or so