Introduction to Lucene

CS 510

Information Retrieval on the Internet

Lucene

- An open source project, part of Apache
- A set of libraries, or a toolkit, for **building** a search engine
- Written in Java; ported to some other languages
  - Only the Java version will be supported in this course
- API available on the web

Lucene

- http://lucene.apache.org/java/docs/index.html
- http://lucene.apache.org/java/docs/features.html
- http://wiki.apache.org/jakarta-lucene/LuceneFAQ
- http://lucene.apache.org/java/3_0_1/gettingstarted.html
- http://lucene.apache.org/java/3_0_1/api/demo/index.html
- http://lucene.apache.org/java/3_0_1/api/

Organization of Lucene

- Core libraries
  - Contain code for dealing with text processing, document indexing, parsing queries, and searching the index
  - http://lucene.apache.org/java/3_0_1/api/
- External resources that others have contributed

Building a Search Engine

Need

1. A collection of documents to be indexed
2. A program to build an index
3. A user interface
4. A program to search the index for documents to match the query

Numbers 3 & 4 can be combined

Lucene Demo Programs

- Described in detail in the “Getting Started” pages of the Lucene website
- The demo programs and the source code come with the Lucene download
- IndexFiles.java
  - Basic code to index all the files in a given directory
- SearchFiles.java
  - Basic code to get a user query from the command line, search the indexed documents, and return results

```
 try {
     IndexWriter writer = new IndexWriter(FSDirectory.open(INDEX_DIR), new
StandardAnalyzer(Version.LUCENE_CURRENT), true, IndexWriter.MaxFieldLength.LIMITED);
     System.out.println("Indexing to directory '" +INDEX_DIR+ "'...");
     indexDocs(writer, docDir);
....
}
....
  static void indexDocs(IndexWriter writer, File file)
    throws IOException {
    // do not try to index files that cannot be read
    if (file.canRead()) {
      if (file.isDirectory()) {
        String[] files = file.list();
        // an IO error could occur
        if (files != null) {
          for (int i = 0; i < files.length; i++) {
            indexDocs(writer, new File(file, files[i]));
          }
        }
      } else {
        System.out.println("adding " + file);
        try {
```

Excerpt from IndexFiles.java in the demo

```
 IndexReader reader = IndexReader.open(FSDirectory.open(new File(index)), true); // only
searching, so read-only=true

   if (normsField != null)
     reader = new OneNormsReader(reader, normsField);

   Searcher searcher = new IndexSearcher(reader);
   Analyzer analyzer = new StandardAnalyzer(Version.LUCENE_CURRENT);

   BufferedReader in = null;
   if (queries != null) {
     in = new BufferedReader(new FileReader(queries));
   } else {
     in = new BufferedReader(new InputStreamReader(System.in, "UTF-8"));
   }
   QueryParser parser = new QueryParser(Version.LUCENE_CURRENT, field, analyzer);
   while (true) {
     if (queries == null)                    // prompt the user
       System.out.println("Enter query: ");

     String line = in.readLine();

     if (line == null || line.length() == -1)
       break;
```

Excerpt from SearchFiles.java in the demo

Project I

- Preliminaries
  - Download Lucene 3.0 and the source code
  - Look at, run, the demo programs
  - Use the source code for the demos and the Lucene API to understand what's happening

Project I

- Assignment
  - Create your own simple search engine
    (You can base your code on the demos and copy from the demo source code)
  - Index the provided files
  - Answer some questions
  - Modify your search engine
  - Answer more questions
  - Submit your answers and your code