Tech Review : Apache Lucene

Introduction:

Apache Lucene is an open source tool used which was written in java originally and is supported by Apache Software Foundation. It has been now ported to various other languages as well like C , C# , Ruby, Python, PHP. It is suitable for indexing and searching. One main reason that Apache Lucene is considered in such high regard is that it can return search responses quickly.

Body :

Lucene is used for full text indexing and searching. It is majorly utilized for internet search engines. Lucene can also be used in recommender systems. Lucene has a class names 'MoreLikeThis' which can be used create recommendations for similar documents. Lucene it self is for indexing and searching but it can be extended to provide crawling and html parsing. Some applications have already been implemented by extending Lucene. Following are some examples:

- Apache Nutch - provides crawling and parsing
- Apache Solr - for enterprise search
- CrateDB - opensource sql built on Lucene
- DocFetcher - desktop search application

Other thank indexing and search Lucene provides other capabilities like spell checking, hit highlighting and advance tokenization capabilities.

Features of Lucene :

- High performance and scalable indexing
  - 150gb/hr on modern hardware
  - Small RAM requirements
  - Incremental indexing which is fast as batch indexing
  - Index size is greater than text indexed.

- Accurate and efficient search algorithms
  - Ranked searching
  - Many query types - wildcard queries, phrase queries, proximity queried and range queries.
  - Sorting available by all fields.
  - Field searching.
  - Multiple - index search
  - Memory - efficient
  - Typo - tolerant suggesters
  - Pluggable ranking models. Includes VSM and Okapi BM25

- Cross - Platform Solution
  - Available as open source which allows us to use Lucene in both commercial and open source program.
  - Pure Java.
  - Implementations in other language also available.

Lucene Basic Concepts:
- Searching and Indexing

Lucene gives fast search responses because instead of searching text as it is it searches based on index. This is similar to retrieving pages in a book based on the keyword on the index page given at the back of the book as opposed to searching word on each page. In short Lucene uses inverted index .

- Document

An index contains one or more documents. Indexing in Lucene involves adding the documents to Index writer and searching involves retrieving the documents from the index using searcher. In Lucene document doesn't really mean document ins English usage. It can be something else for example if the index is for the database each record in the database can be referred as document.

- Fields

A document in Lucene may have one or more fields. A field is a name - value pair. For eg : Title can be considered as a field with title as name and the title content as the value.

- Searching

Before searching can be done index should be build. It involves creating Query and then giving this query to the Searcher which returns list of Hits.

- Queries

Lucene can have own mini - language performing searches. It allows user to tell which feels to search on and which field are more important. It also has the ability to perform boolean query and other functionality.

Query Syntax :
fieldName : text

Range search example:
timestamp:[1509909322,1572981321]

Can also search for wildcard

- Analysis

There are many built in analyzers in Lucene :

- StandardAnalyzer - analysis based on basic grammar . Converts to lowercase and removes stop words.
- SimpleAnalyzer - breaks text based on no - letter character and converts it to lower case.
- WhiteSpaceAnalyzer - breaks text based on white space.

Lucene makes it easy to add full - text search to the application. It is simple and can be one by doing the following steps:

- Index : create an in memory index from some strings.
- Query : read the query from stdin parse it and build a Lucene Query from it.
- Search : create a searcher based on the query to search the index. Then use TopScoreDoc collector to collect the top n pages.
- Display : Display the results we got from the search to the user.

Lucene can be used along with maven as well by adding following dependencies :

```
<dependency>
   <groupId>org.apache.lucene</groupId>
   <artifactId>lucene-core</artifactId>
   <version>7.1.0</version>
</dependency>

<dependency>
   <groupId>org.apache.lucene</groupId>
   <artifactId>lucene-queryparser</artifactId>
   <version>7.1.0</version>
</dependency>
```

Conclusion:
All in all Lucene is a very good tool for searching and indexing but it lacks some functionalities like crawling and parsing. It has many success stories. It is continuing to make rapid progress. It is a class library with enough customization and optimization space. But yet additional development work is required. All extension, distribution, reliability etc need to be implemented by themselves, non - real time , there is a time delay from indexing to searchable.

Resources Used:

- https://www.lucenetutorial.com/basic-concepts.html
- http://www.lucenetutorial.com/lucene-in-5-minutes.html
- https://www.baeldung.com/lucene
- https://lucene.apache.org
- https://en.wikipedia.org/wiki/Apache_Lucene