COMP4321 Search Engines for Web and Enterprise Data – Spring (2017)

Course Project

Ng Wai Wah (20278342) Participation: 33.3333%

Ng Wai Sum (20278354) Participation: 33.3333%

Leung Kung Ming (20269688) Participation: 33.3333%

**Outline of the JAVA class:**

* Crawl.java :

Crawler which information from a webpage such as extract title, word and link.

* Extract\_db.java :

Class used to extract the information in the database for Phase 1 submission.

* Spider.java :

Spider function to extract webpage by Breath-First-Search Algorithm.

* IndexTool.java :

Indexer for indexing the database with two hash table.

* InvertedIndex.java :

Indexer for indexing the database with inverted file.

* PageInfm :

Hash table from ID to Information: URL, last-date-of-modification, size

* Porter.java :

Class storing the Porter Stemming Algorathm

* StemStop.java :

Class to take away the stopwords from the text in a webpage.

* Webpage.java :

Class of a webpage.

* Word.java :

Class of a word with string and frequency.

* SearchTool.java :

Class for searching and generating result

**Extra Libraries used:**

． htmlparser1\_6.jar ．jdbm-1.0.jar ．jsoup-1.10.2.jar

**Class used by different data:**

IndexTool

* Page : URL ⬄ PageID
* Word : Word ⬄ WordID
* Title : Title ⬄ TitleID
* ChildParent : ParentID⬄ChildID

InvertedIndex

* ParentChild : Parent => [ChildID]
* Inverted : WordID => [PageID: Frequency]
* fullWordInverted: WordID => [PageID: Postion 1, Position 2, …]
* fullWordForward: PageID => [word]
* Forward : PageID => [word]
* termW : PageID => [word : termFrequency]

PageInfm

* PageProperty : PageID => Infm

\*Infm: title, URL, last modified date, page size

Run of spider function:

1. The webpage would be run in a while loop while there are URL stored in the TodoList and the number of page extracted has not exceed the maximum.
2. Check whether the first element in TodoList is extracted. If extracted before, check for update, else add new entry to PageIndexer.
3. Extract the words in the webpage and count the frequency, store it in a forward index. Also count the max frequency and add in inverted index and maximum frequency table.
4. Extract all words in the webpage and count the frequency, sore it in full word forward index and full word inverted index.
5. Get the last update of webpage
6. Extract the page size of the webpage
7. Add required information into PageProperty.
8. Add entry for the mapping of parent link to child link and child link to parent link
9. Loop back to Step 1.

**Weight Calculation**

* Weight calculation:

wij = (tfij /max(tfij))\* idfj

tfij = frequency of term j in document i

dfj = document frequency of term j

= number of documents containing term j

idfj = inverse document frequency of term j

= log2 (N/ df j) (N: number of documents in collection)

* Cosine Similarity



\*query term is set to be equal to 1

Special remarks

* As the webpage is started from the cse page which requires login so password and username is required to be set in Crawl.java before using spider function.
* The spider function would throw exception and skip the website if HTTP response code is other than 200
* The handling of the words is handled with UTF-8