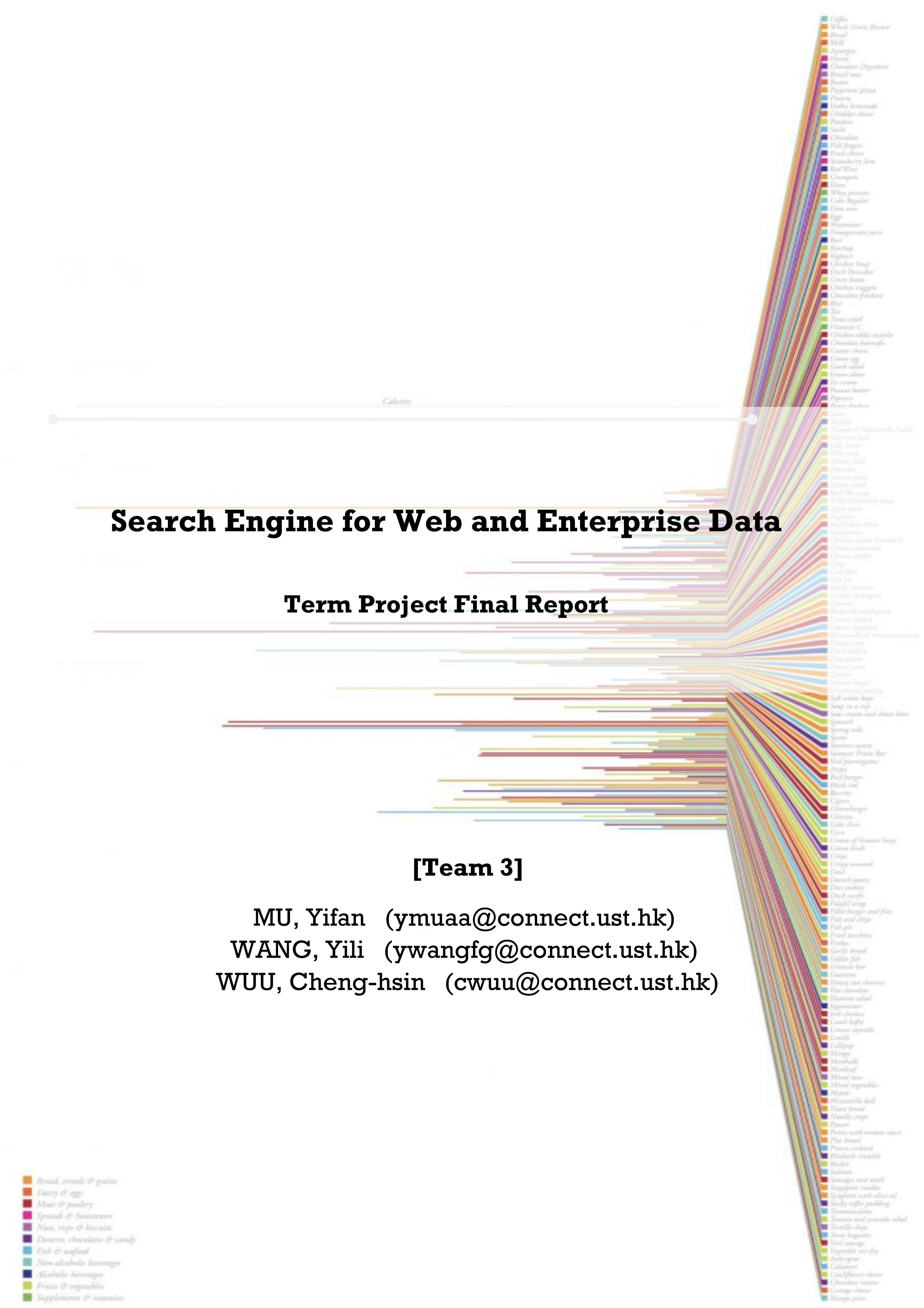


Search Engine for Web and Enterprise Data

Term Project Final Report

[Team 3]

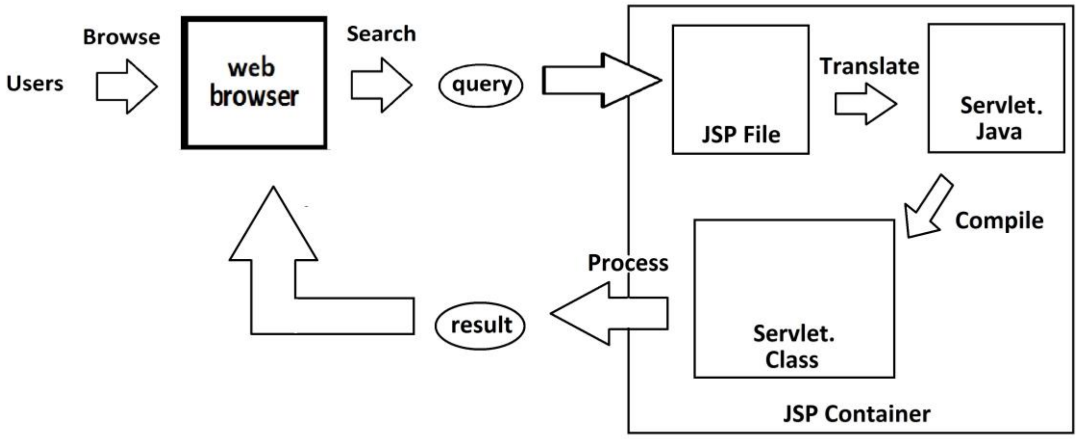
MU, Yifan (ymuaa@connect.ust.hk)
WANG, Yili (ywangfg@connect.ust.hk)
WUU, Cheng-hsin (cwuu@connect.ust.hk)

1. Introduction

In this project, we built a web search engine that allows the user to type the query on the web interface and get the relevant pages based on the database built by crawling specific URL.

2. Overall Design

1) Working Flow



Our group implemented the project according to the graph above. The crawler crawls text information from the webpage and stores them in the database according to the database schema. Users can use webpage interface to submit the query for using our search functions. The search engine will process the query by selecting the keywords for matching information with indexed pages in the database, and then returned and displayed the sorted pages to the users through the webpage.

2) Working Principle

Component	Tool	Component	Tool
Database storage	jdbm	**Data manager**	class Indexer
Crawler	htmlparser	**Query**	class SE
Stemming	Porter's Algorithm	**Web Interface**	html, css, ajax
Server	jsp		

I. Database Storage and Manager
The database is built by using jdbm, which is a comprehensive java package to handle database. It can store keys and corresponding values of all kinds of java types in the hashtable or B+ tree. It also provides users simple methods to access the database.

II. Crawler
In the crawler part, we use URL class and htmlparser package to crawl information from the webpage. URL class provides methods to connect webpages and detect potential errors and exceptions on the internet. The htmlparser package provides methods to extract text data from the webpage by analyzing html tags.

III. Stemming

For string processing, we use Porter’s algorithm to stem keywords so that different forms of words and be indexed the same. The stemming process can improve precision of the search engine. In addition, we remove stop words that are less meaningful to enhance database and information retrieval efficiency.

IV. Query

For the query part, we use vector space model to do union search and phrase search for the selected keywords. For the same query, if one page contains exactly the same phrase or have the words in searched query appear in the title will get extra bonus when calculating the final score, and thus be presented in the higher order for the user.

V. Server

In the server part, we install Tomcat in our virtual machine and use it as our webserver and JSP files are used to communicate with server directly.

VI. Web Interface

There is a HTML file used to set up the initial web interface and it mainly contains a form to submit the users’ input, a button to control similar page searching, and an area to show brief searching results dynamically through AJAX.

3. The file structures used in the index database

1) Indexer

Class	Mapping
MappingTable	url <-> pageID, word <-> wordID
InvertedIndex	wordID -> Posting list (Posting: pageID, frequency, word position list)
PageContent	pageID -> page content (Page: title, page size, modified date, url)
ForwardIndex	pageID -> parent links, pageID -> child links
Forward	pageID -> forward posting list (wordID, frequency)

Inside database, we use hash table to store keys and values

2) User-defined Class

Class	Content	Usage
Posting	pageID, frequency, word position list	Support class inverted index
fpair	wordID, frequency	Support class forward index
scoreMap	ageID, score	Support class SE

4. Algorithm

1) Porter’s algorithm (stemming)

2) Vector space model (query)

I. Term Weight Formula

$$w_{ij} = (tf_{ij} / max_l\{tf_{lj}\}) \bullet idf_j$$

II. Cosine Similarity Measure

$$\text{CosSim}(D_i, Q) = \frac{D\circ Q}{|D||Q|} = \frac{\sum_{k=1}^{t}(d_{ik}q_k)}{\underbrace{\sqrt{\sum_{k=1}^{t} d_{ik}^{2}}}_{Document\ length}\underbrace{\sqrt{\sum_{k=1}^{t} q_{k_{t_2}}^{2}}}_{Query\ length}}$$

Query length here is assumed to be 1.

III. Title favoring matches
We set an extra title bonus for those pages that has the term contains in their title.

5. Installation

1) Compile .java source code into .class files
2) Place corresponding packages under the tomcat_folder/webapps/ROOT/WEB-INF/classes/,
3) Place jdbm.jar and htmlparser.jar under tomcat_folder/webapps/ROOT/WEB-INF/lib/ folder
4) Place form.htm, jump.jsp, background.jpg, bonus3.jsp, example3.jsp, logo.jpg, similar.jsp, trans.jsp and test.jsp in tomcat_folder/webapps/ROOT/ folder.
5) Place .db file and .lg file under tomcat_folder/bin/ folder.

6. Highlight of features beyond the required specification

1) Similar Page Search
After searching the input word, we count top 5 most frequent keywords (excluding stop words) that appear in the first page returned from search engine based on the input query. By clicking “Get Similar Page!”, we will return the relevant page that searched by that top 5 most frequent keywords. It allows users to have a higher chance of searching the page that they needed as it will neglect the “less related” keyword.

2) Topic Search
After searching the input word, we count top 10 most frequent keywords (excluding stop words) that appear in the 10 pages returned from search engine based on the input query. By clicking “Topic Search” we will return the relevant page that searched by that top 10 most frequent keywords. It gives user a search result that based on the topic covers the original input query.

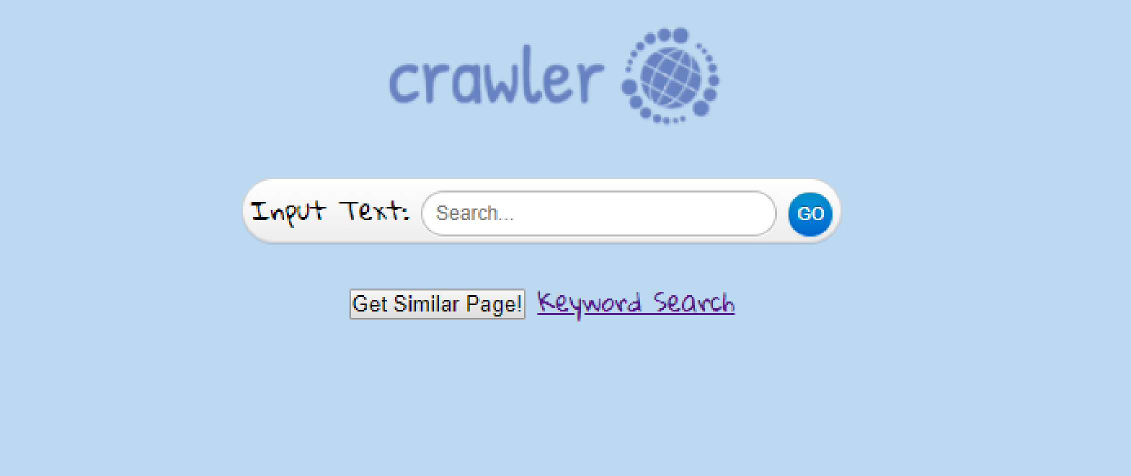
3) Selected Term Search
Selected Term Search allows users to view all the indexed term (stemmed) and select several terms from them as query keywords.

4) Instantaneous and Adventurous Interface
By using AJAX, brief searching results is shown immediately on current page while the user is inputting some words in the form. Then the user can get the relevant results dynamically and change the query according to current results if he wants.

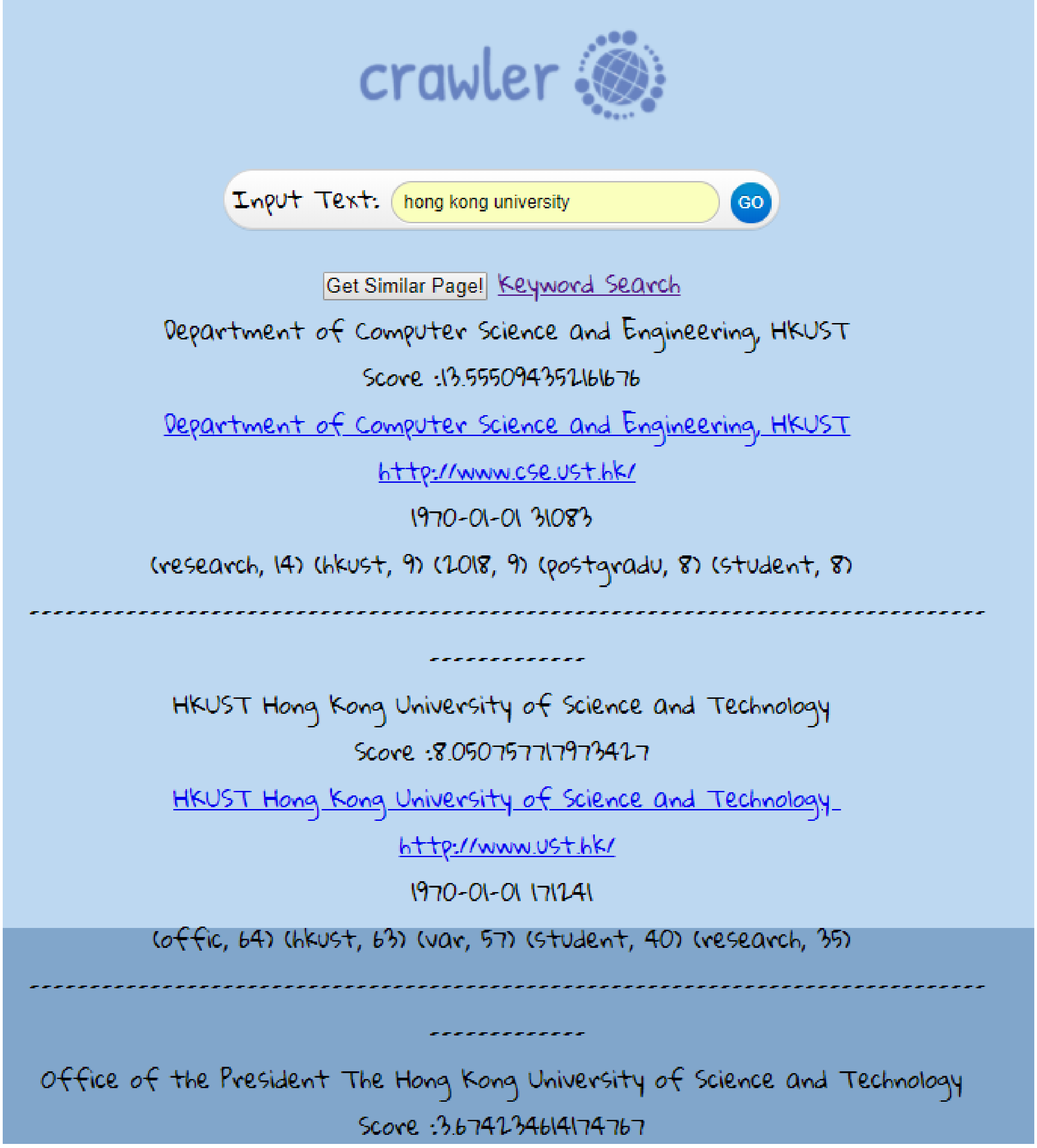
7. Testing Result

(URL: http://143.89.130.15:8080/form.htm)

1) Web interface loading page

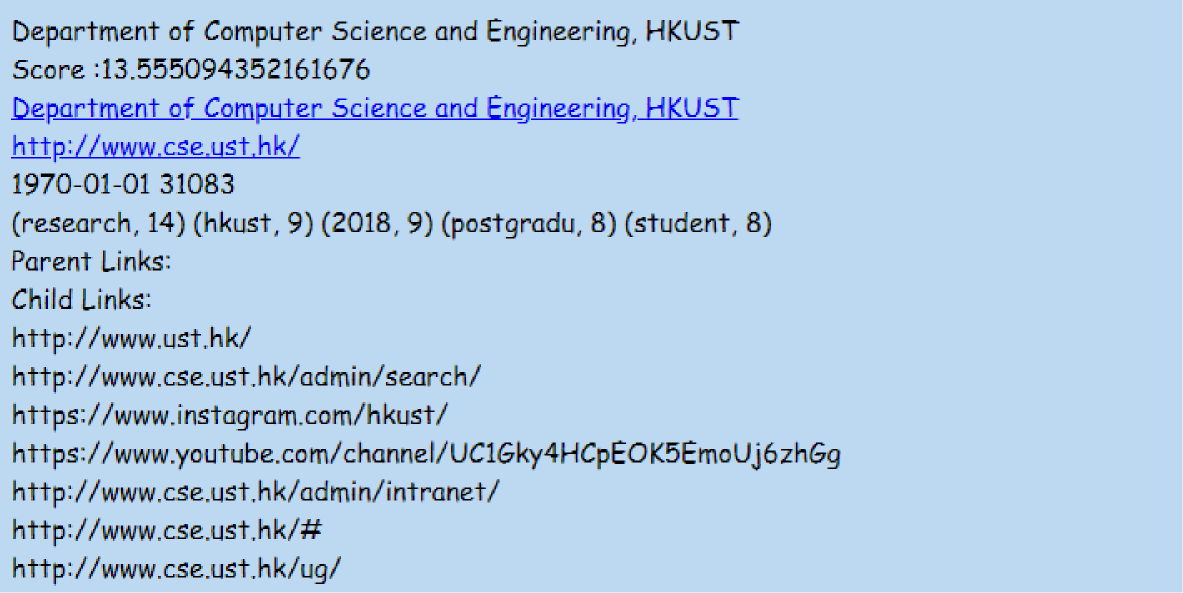


2) Search input query, interface give instantaneous interaction



```
Department of Computer Science and Engineering, HKUST
Score :13.555094352161676
Department of Computer Science and Engineering, HKUST
http://www.cse.ust.hk/
1970-01-01 31083
(research, 14) (hkust, 9) (2018, 9) (postgradu, 8) (student, 8)
--------------------------------------------------------------------------------
-------------
HKUST Hong Kong University of Science and Technology
Score :8.050757717973427
HKUST Hong Kong University of Science and Technology
http://www.ust.hk/
1970-01-01 171241
(offic, 64) (hkust, 63) (var, 57) (student, 40) (research, 35)
--------------------------------------------------------------------------------
-------------
Office of the President The Hong Kong University of Science and Technology
Score :3.674234614174767
```

3) Press “Go” to get more details of the pages returned by search engine



Department of Computer Science and Engineering, HKUST

Score :13.555094352161676

[Department of Computer Science and Engineering, HKUST](http://www.cse.ust.hk/)

http://www.cse.ust.hk/

1970-01-01 31083

(research, 14) (hkust, 9) (2018, 9) (postgradu, 8) (student, 8)

Parent Links:

Child Links:

http://www.ust.hk/

http://www.cse.ust.hk/admin/search/

https://www.instagram.com/hkust/

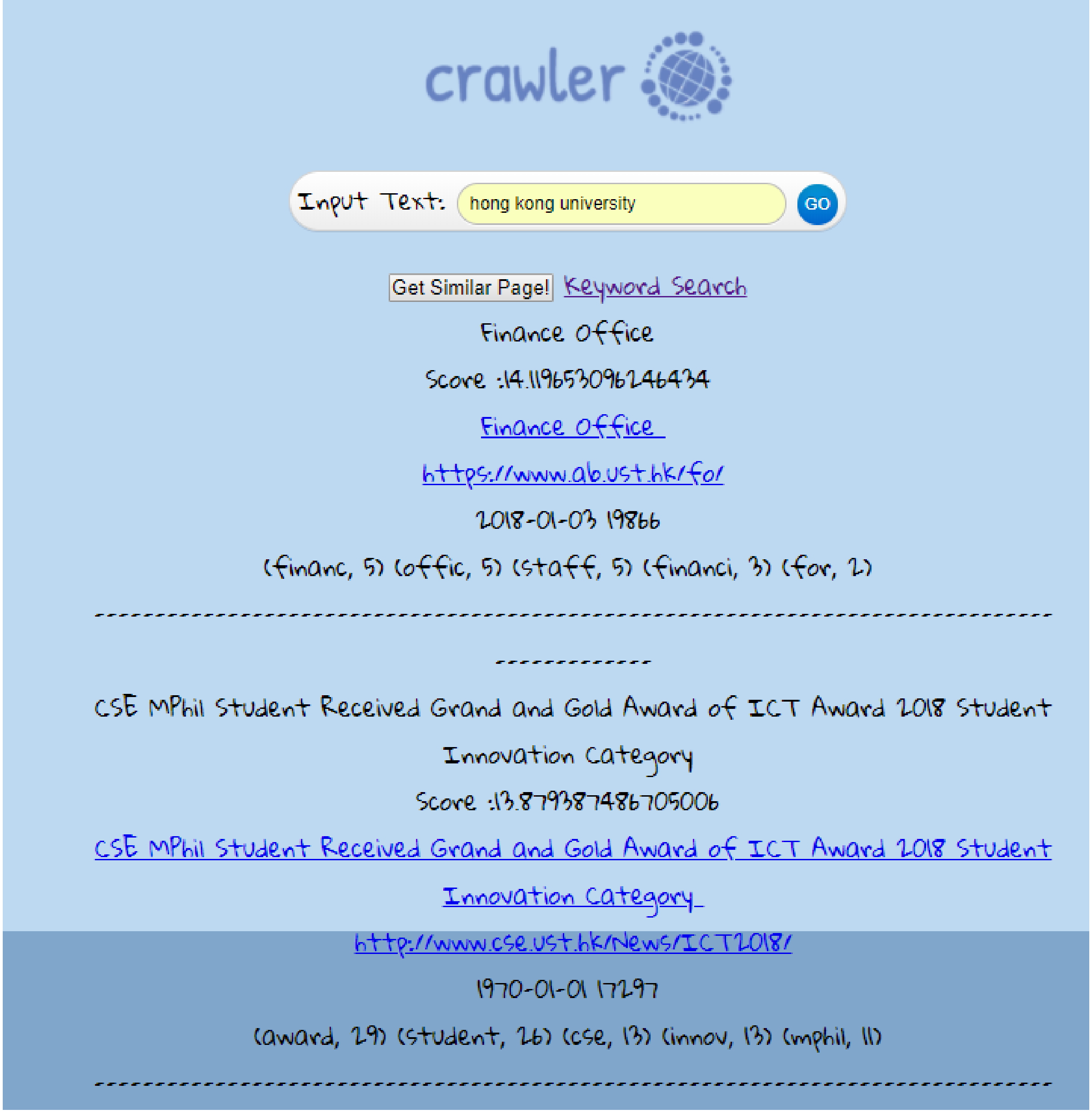
https://www.youtube.com/channel/UC1Gky4HCpEOK5EmoUj6zhGg

http://www.cse.ust.hk/admin/intranet/

http://www.cse.ust.hk/#

http://www.cse.ust.hk/ug/

4) Press “Get Similar Page” to get the pages by searching more relevant keywords



Input Text: hong kong university

Finance Office

Score :14.119653096246434

Finance Office

https://www.ab.ust.hk/fo/

2018-01-03 19866

(financ, 5) (offic, 5) (staff, 5) (financi, 3) (for, 2)

CSE MPhil Student Received Grand and Gold Award of ICT Award 2018 Student Innovation Category

Score :13.879387486705006

CSE MPhil Student Received Grand and Gold Award of ICT Award 2018 Student Innovation Category

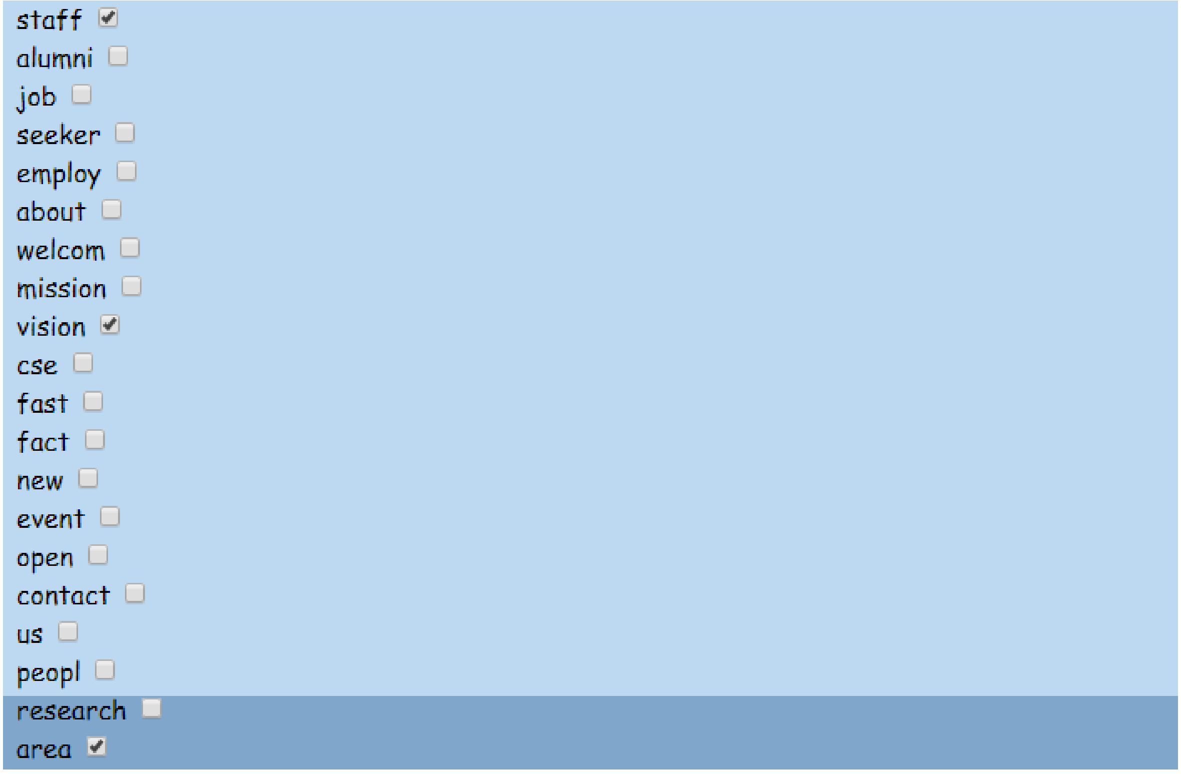
http://www.cse.ust.hk/News/ICT2018/

1970-01-01 17297

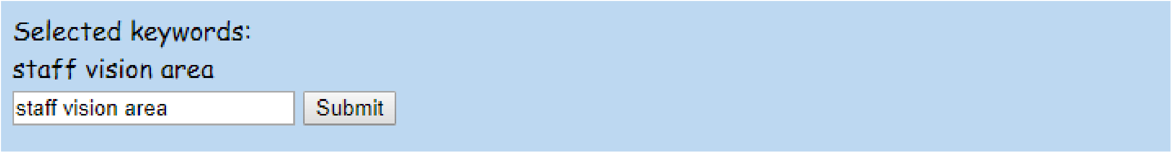
(award, 29) (student, 26) (cse, 13) (innov, 13) (mphil, 11)

----

5) Click “Keyword Search” to get keywords from pages returned by search engine

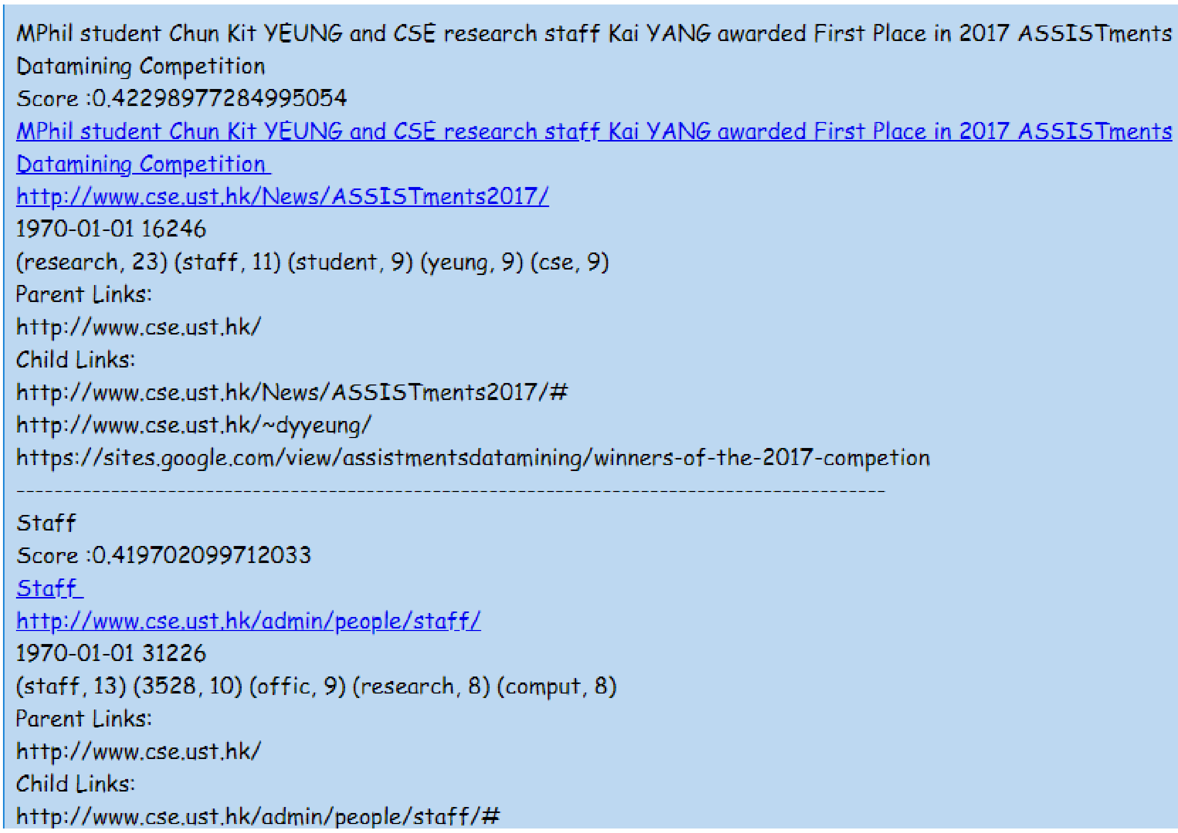


staff ☑
alumni ☐
job ☐
seeker ☐
employ ☐
about ☐
welcom ☐
mission ☐
vision ☑
cse ☐
fast ☐
fact ☐
new ☐
event ☐
open ☐
contact ☐
us ☐
peopl ☐
research ☐
area ☑



Selected keywords:
staff vision area

staff vision area



MPhil student Chun Kit YEUNG and CSE research staff Kai YANG awarded First Place in 2017 ASSISTments Datamining Competition
Score :0.42298977284995054
MPhil student Chun Kit YEUNG and CSE research staff Kai YANG awarded First Place in 2017 ASSISTments Datamining Competition
http://www.cse.ust.hk/News/ASSISTments2017/
1970-01-01 16246
(research, 23) (staff, 11) (student, 9) (yeung, 9) (cse, 9)
Parent Links:
http://www.cse.ust.hk/
Child Links:
http://www.cse.ust.hk/News/ASSISTments2017/#
http://www.cse.ust.hk/~dyyeung/
https://sites.google.com/view/assistmentsdatamining/winners-of-the-2017-competion

---

Staff
Score :0.419702099712033
Staff
http://www.cse.ust.hk/admin/people/staff/
1970-01-01 31226
(staff, 13) (3528, 10) (offic, 9) (research, 8) (comput, 8)
Parent Links:
http://www.cse.ust.hk/
Child Links:
http://www.cse.ust.hk/admin/people/staff/#

8. Conclusion

1) Strength

   I. Integrated handling of database

      We integrate all the accessing and modifying function of database in one class Indexer so that we don’t need to directly handle database relationships. This feature provides an overall control of database and reduces the probability of wrong operation on database

   II. Multiple searching options

      We implemented several options of searching. Our search engine supports topic search and similar page search. Topic search allows users to submit a certain topic and return pages that related to the topic. Similar page search allows users to search for pages that are similar but do not exactly match the keywords.

2) Weakness

   I. Relatively slow speed of crawler

      Because of repeatedly and redundantly use of url connection and functions to extract text information from webpage, our crawler currently performs at a relatively low efficiency.

   II. Complicated database design

      We implement many seldom used functions to access and modify database. Because of the lack of structural design of our database schema, some redundant information was stored in database. Also, the retrieval functions need to be improved to achieve higher efficiency.

3) Improvement

   I. Better database structure

      We need to analyze database relationship and design a more structural database schema to implement a less redundant data structure, which is expected to enhance the total efficiency of our web search engine.

   II. Better retrieval algorithm to achieve higher recall and precision

      We can apply SVM to our retrieval algorithm which has vector space model now.

   III. Use better api and recursive algorithm to handle crawling procedure.

9. Contribution

MU, Yifan: 1/3, WANG, Yili: 1/3, WUU, Cheng-hsin: 1/3