DEVELOPMENT OF SEARCHBADGER, A FRAMEWORK FOR EVALUATION OF SEARCH RESULTS

Sebastian Merli[1,2] and Hao Shi[1]

[1]School of Engineering and Science, Victoria University, Melbourne, Australia
hao.shi@vu.edu.au
[2]Technische Universität München,Germany
sebastian.merli@mytum.de

ABSTRACT

Google represents a significant share in the search market and dominates the world of search engines. Although Google has constantly made effort to improve its search results by adding a result page preview and offering a block URL option, individual user feedback to the real time search operation is still very limited. In this paper, a new framework called SearchBadger is proposed to optimise search results in real time by interacting with user and taking into account immediate user feedback. The proposed framework SearchBadger is developed using C#, db4o and Visual Studio. SearchBadger architecture contains the major component BadgerWorkspace and is followed by graphical user interface and search behaviour. Keyterm significance and stemming are discussed and then page re-ranking is achieved by instant user feedback. The framework can be integrated to the existing search engines or served as a test bed for future development of semantic web.

KEYWORDS

SearchBadger, BadgerWorkspace, Framework, Web Search, Search Engine, Google, db4o(database for objects), WPF (Windows Presentation Foundation) and Search Optimisation.

1. INTRODUCTION

Who dominates Search Engine market share? The answer undoubtedly is Google. Google, Bing and Yahoo are the front runners followed by Baidu, a Chinese search engine with support of a largest population in the world. The first four search engines have almost 99% market with Google's share over 85% [1]. Google holds the leading edge which is reflected by many new products introduced in recent years. For example, Google Search added a top menu bar to provide a direct link to its Image Search, Google Maps, YouTube, Gmail and Translation. Google released its own web browser, Google Chrome in 2008. Google Chrome is catching up very quickly with current share over 20% [2] even though Internet Explorer is till dominating in Web Browser market. The report from Wikimedia indicates Google Chrome exceeding other web browser on Wikimedia with over 27.25% share [3]. The growth of Google Chrome from 1% in February 2009 to 10% in April 2011, and then 27% in August 2012 is unprecedented [4].Android, Google's open source Operating System for touch screen mobile devices has helped Samsung overtake Apple to claim smartphone market share lead [5]. Google phenomenon heavily replies on the success of Google Search which is regarded as the most-used search engine on the World Wide Web, receiving hundred million hits each day [6]. Although Google Search is constantly improving its search results, for example, a result page preview and a URL block option, fundamentally Google replies on 'page rank' from collected user actions in a period of time, not individual user action in real time. In this paper, the proposed SearchBadger aims to optimise search results by individual user feedback in real time.

2. The SearchBadger Framework

The SearchBadger framework is developed using C#, db4o(database for objects) and Visual Studio. A graphical user interface developed using WPF is set on top of it.WPF is framework for developing graphical interfaces by Microsoft [7]. WPF provides a tier approach which completely separates the user interface from the business logic:"WPF applications can be either deployed as standalone desktop programs or hosted as an embedded object in a website" [7]. db4o is one of the open source object databases. It supports both Java and .NET platforms with their respective APIs [8]. Visual Studio is an integrated software development environment created by Microsoft [9]. It supports multiple programming languages such as C++, C#, and VB .NET and can be deployed as desktop or web-based application. The overall development environment for SearchBadger is illustrated in Figure 1.

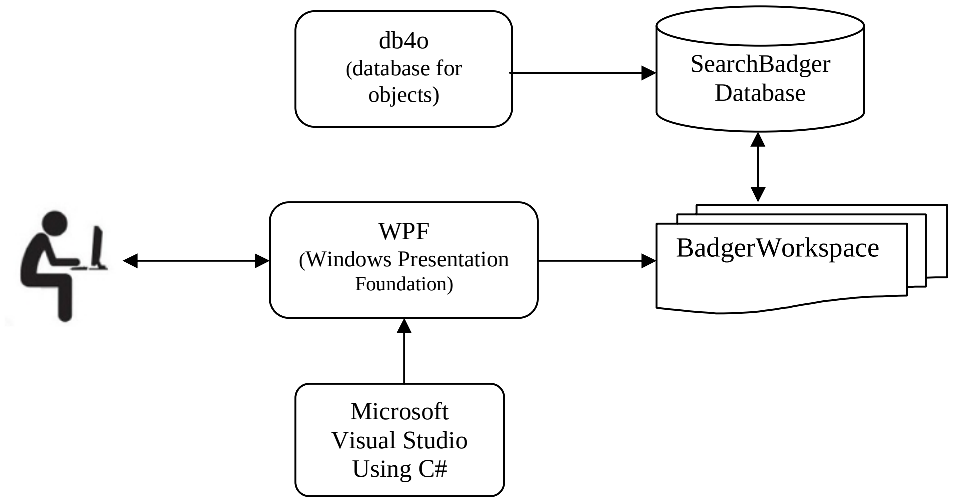

db4o
(database for objects)
SearchBadger Database
WPF
(Windows Presentation Foundation)
BadgerWorkspace
Microsoft Visual Studio Using C#

Figure 1. SearchBadger development environment
(user icon is sourced from images.google.com)

The SearchBadger framework aims:

- To provide a UI which allows a user to conduct web search more efficiently
- To allow the user to influence search criteria and change weighting
- To capture the user's search strategies
- To collect the user's feedback for statistical analysis in term of successful searches
- To use the user's just-in-time feedback to optimise immediate search results

2.1. System Architecture

The main part of SearchBadger's framework is the BadgerWorkspace as shown in Figure 2. It provides a facade to control search and page ranking. Basically the BadgerWorkspace allows a user to add and update criteria and start search operations on a chosen search engine. The management of the results and tracking of the user's actions are carried out in background automatically and asynchronously. That means, the workspace is accessible during operations and offers a consistent view all the time.

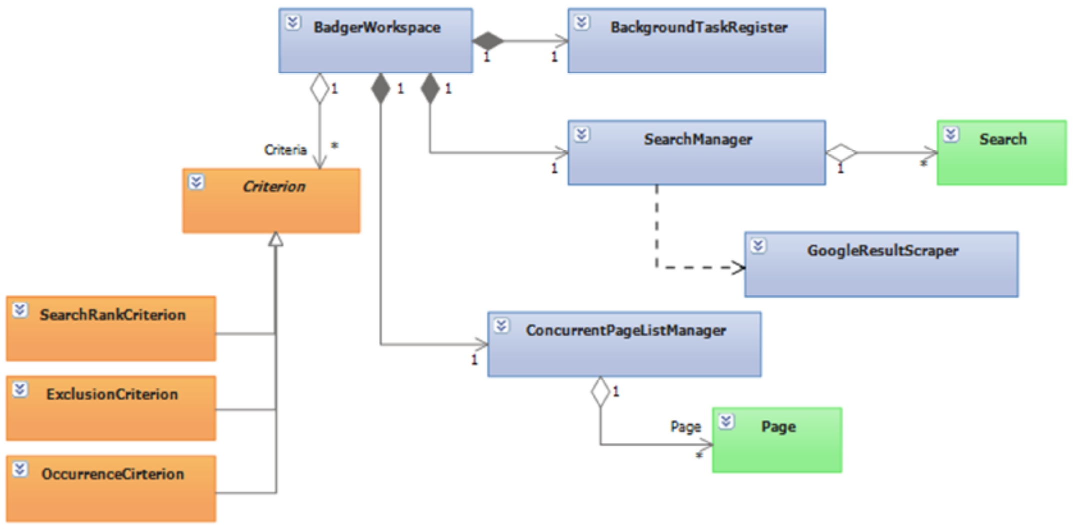


Figure 2. SearchBadger architecture

The search request presented to the user is formulated in criteria, where one Criterion stands for one feature of a result page. Every implemented Criterion derives from the abstract base class Criterion, which implements communication with the workspace for evaluating every result page. The specific Criterion implements the code that actually evaluates a page. The BadgerWorkspace consists of three implemented criteria:

- *OccurrenceCriterion*
  Count the occurrences of a given keyword and compare it to a target value.
- *ExclusionCriterion*
  Look for a given keyword and rate it good if it isn't in the page
- *RankCriterion*
  Check how the search engine ranks the page in a given search

Whereas criteria are listed and managed by *BadgerWorkspace* itself, for searches it uses the class *SearchManager*, which keeps track of the list of performed searches and starts the search asynchronously, e.g. *Google Result Scraper*, which is a placeholder for any possible search operation using Google search engine. This approach makes the SearchBadger framework portable, scalable and extensible. Later a class like *BingResultScraper* can be easily added to perform searches on Bing. The detail is presented in the class diagram as shown in Figure 3.

Every new result generated by SearchManager, creates a *Page*, which is responsible for extracting its content from the search engine and reporting results so that evaluations can be carried locally. Once the *Page* is loaded and evaluated by all criteria, the *Page* calculates its overall rating.

Like search, pages are also managed by a special class, *ConcurrentPageListManager* which has an immensely more difficult task. While a search is running, new pages are continuously added and evaluated by every Criterion. In the meantime, the user can add or change criteria, which make it necessary to re-evaluate every existent page. *ConcurrentPageListManager* keeps track of these operations and ensures that the lists of pages are sorted by rating and kept consistent at all times, while informing the UI of every change that has occurred.

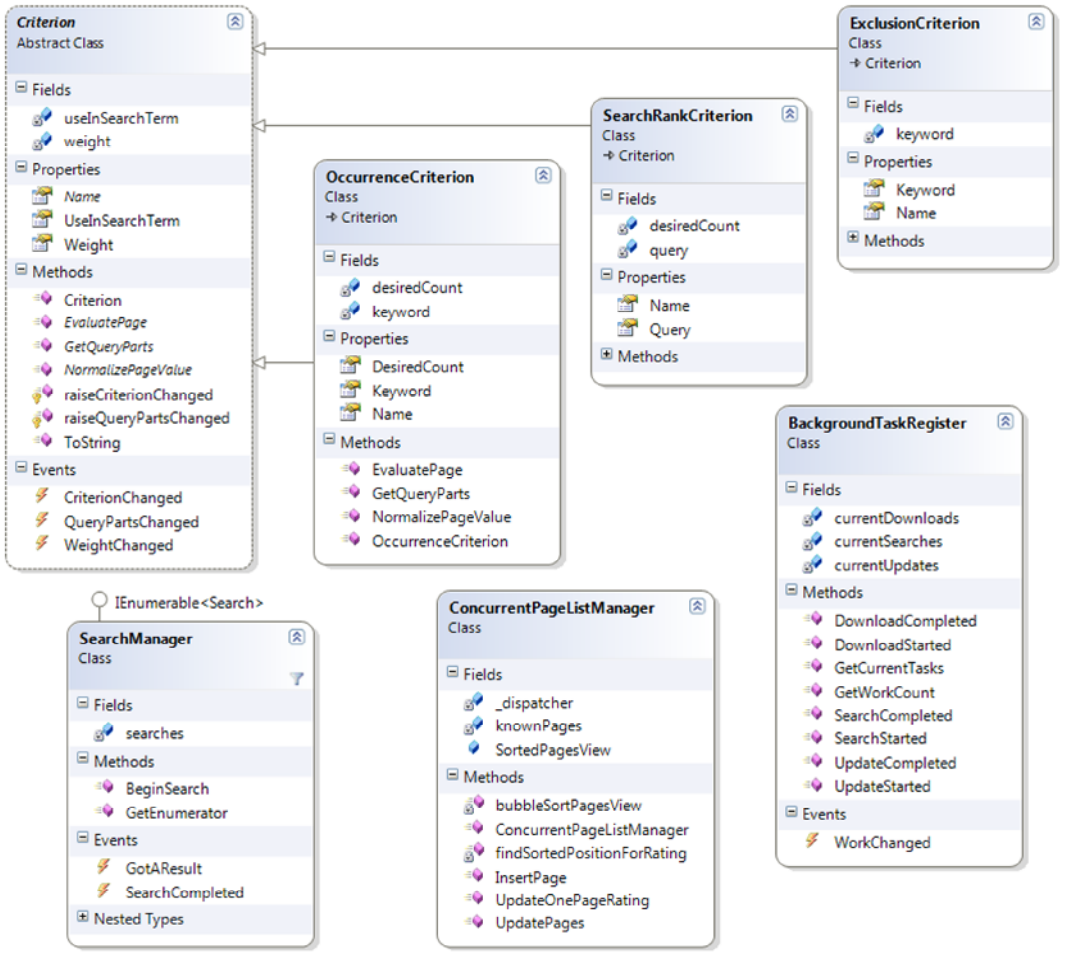
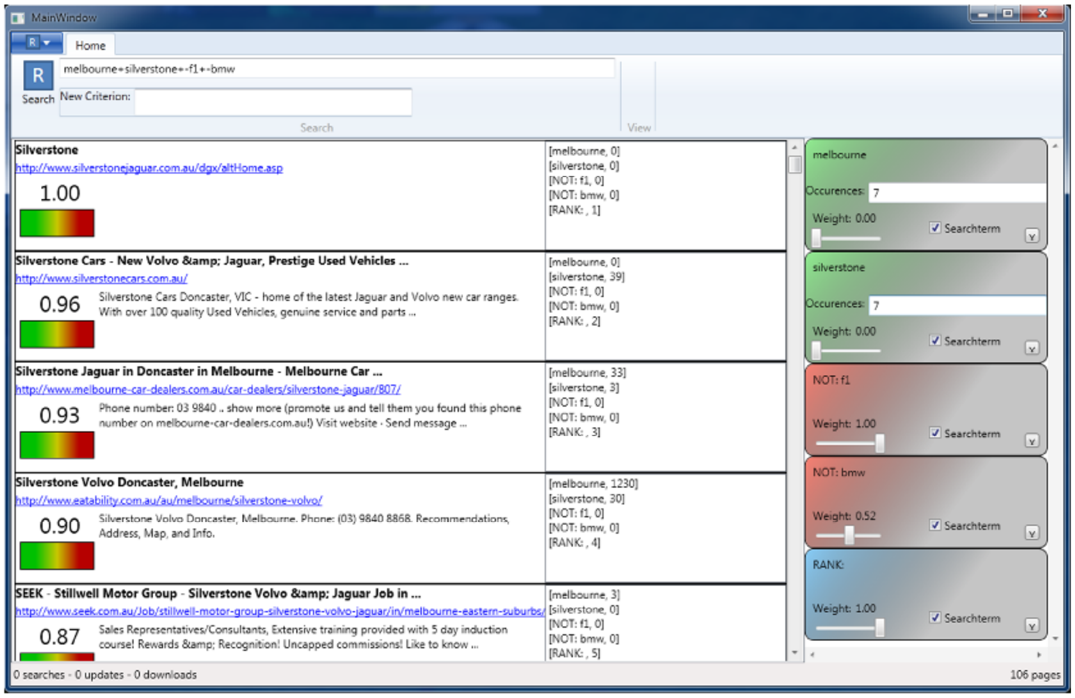


Figure 3. The class diagram

2.2. Graphical User Interface

The developed GUIbased on Google search engine consists of two columns as shown in Figure 4. The criteria list on the right is to control the search. A new criterion can be edited and added to the list, which means the user can change a search strategy while running over the search results, and receive a page ranking in real-time. The left column shows the actual contents. The search results are listed under their title, URL and short description which are gathered from the search engine. In addition every search result has its own rating, calculated from the criteria defined on the right and the result for every single criterion is shown in a list. Once a result page is shown to a user, the user can give the feedback, telling if the page is helpful, by using the gradient slider under the rating.



melbourne+silverstone+-f1+-bmw

Silverstone
http://www.silverstonejaguar.com.au/dgx/altHome.asp
1.00

[melbourne, 0]
[silverstone, 0]
[NOT: f1, 0]
[NOT: bmw, 0]
[RANK: , 1]

Silverstone Cars - New Volvo &amp; Jaguar, Prestige Used Vehicles ...
http://www.silverstonecars.com.au/
0.96 Silverstone Cars Doncaster, VIC - home of the latest Jaguar and Volvo new car ranges. With over 100 quality Used Vehicles, genuine service and parts ...

[melbourne, 0]
[silverstone, 39]
[NOT: f1, 0]
[NOT: bmw, 0]
[RANK: , 2]

Silverstone Jaguar in Doncaster in Melbourne - Melbourne Car ...
http://www.melbourne-car-dealers.com.au/car-dealers/silverstone-jaguar/807/
0.93 Phone number: 03 9840 .. show more (promote us and tell them you found this phone number on melbourne-car-dealers.com.au!) Visit website · Send message ...

[melbourne, 33]
[silverstone, 3]
[NOT: f1, 0]
[NOT: bmw, 0]
[RANK: , 3]

Silverstone Volvo Doncaster, Melbourne
http://www.eatability.com.au/au/melbourne/silverstone-volvo/
0.90 Silverstone Volvo Doncaster, Melbourne. Phone: (03) 9840 8868. Recommendations, Address, Map, and Info.

[melbourne, 1230]
[silverstone, 30]
[NOT: f1, 0]
[NOT: bmw, 0]
[RANK: , 4]

SEEK - Stillwell Motor Group - Silverstone Volvo &amp; Jaguar Job in ...
http://www.seek.com.au/Job/stillwell-motor-group-silverstone-volvo-jaguar/in/melbourne-eastern-suburbs/
0.87 Sales Representatives/Consultants, Extensive training provided with 5 day induction course! Rewards &amp; Recognition! Uncapped commissions! Like to know ...

[melbourne, 3]
[silverstone, 0]
[NOT: f1, 0]
[NOT: bmw, 0]
[RANK: , 5]

melbourne
Occurences: 7
Weight: 0.00 ✓ Searchterm

silverstone
Occurences: 7
Weight: 0.00 ✓ Searchterm

NOT: f1
Weight: 1.00 ✓ Searchterm

NOT: bmw
Weight: 0.52 ✓ Searchterm

RANK:
Weight: 1.00 ✓ Searchterm

0 searches - 0 updates - 0 downloads
106 pages

Figure 4. Graphical User Interface

2.3.Analysis of Search Behaviour

When a user searches using SearchBadger several ‘behaviours’ can be traced:

- the way that the user searches, e.g. the keywords or criteria that the user chooses,
- the order of pages viewed during search,
- the success of the search through the user feedback to the results,
- the fine-tune to the query while the user is searching and gathering more information about the search domain.

Likewise, SearchBadger can easily be used for studying search behaviour. However, keep in mind that even the usage of another frontend, i.e. SearchBadger’s UI instead of web browser window with Google, could change the user’s actions.

2.4.User Influence and Statistical Analysis

One important part of SearchBadger is to give a user wider influence in the way the search is conducted and how the ranking of the result pages is presented. This is achieved by splitting every search request in a number of criteria, where each stands for one ‘*feature*’ of the result page, e.g. like “has at least 7 occurrences of the word ‘Melbourne’ on it”, “is highly ranked in Google when searching”, “has the words ‘Melbourne’, ‘liveable’ and ‘city’ in one sentence”. Every criterion can be given a weight by the user, defining how important this particular ‘*feature’* is.

As the user can see for every page how well it fulfils each criterion, the user can get an idea the influence. By changing the weight or even the configuration of one criterion, the user can watch the rating of the results being updated instantly, which allow the user to analyse his current request in real time. That means, the moment a user rates a page helpful, the result ranking can be changed immediately, so that pages that are similar to the rated page are shown on top and vice versa if the page has been rated not helpful.

When the user provides feedback about the results on screen, it can be used to do a statistical analysis on the effectiveness of different criteria depending on the type of search request. Most search engines are able to use feedback just by recognizing if a user returns from a result page as well as how much time the user spends before returning to the page. Having an environment where the user rates every visited page with a percentage value and where every of these pages is downloaded and stored locally, statistical analysis can be produced very efficiently. It leads to find correlations between the (user stated) helpfulness of a page and different properties of the page (measured as the evaluation of different criteria). It results in the integration of a new criterion not only offers more specific ways of defining a search to the user, but also gives more opportunity to analyse search success.

3. KEY WORD CONSIDERATIONS

As described in Sections 2.4, the SearchBadger can propose more specific keywords to the user such as a rate with "good" which should be included in a result page, "bad" which should not be included in a result page, "ignore" which isn't relevant at all or doesn't matter if it is on a result page. So the result can be re-ranked immediately according to the user's feedback.

3.1 Keyterm significance

In order to evaluate the significance of a word in search, the frequently used words like "the" should NOT be taken into consideration or at least with very little influence. Usually this kind of words is made of corpora with millions of words. Here another attempt is tried: having downloaded many pages of search results to a specified topic, it is important to determine whether the amount of data is enough to get a good rating of word significance. If so, the word is rated highly significant to the related topic.

For every keyterm (1-4 consecutive words) its occurrences are counted per page. The idea is: a word that appears on many topic related pages is topic related itself, whereas a word that appears often on a page is a frequent word itself and rather insignificant. The attempt to get a value for significance is to divide the number of pages a word is on by the number of overall occurrences, where the number of pages is powered by a number between 1 and 2 to increase its influence. A sample calculation of the keyterm significance is listed below:

$$keyterm\ significance = \frac{(number\ of\ pages)^{\sqrt{2}}}{overall\ occurrences}$$

In order to determine the index value, 1, $\sqrt{2}$ and $\sqrt{3}$ are tested. It is found that $\sqrt{2}$ to be reasonable while 1 is too small and $\sqrt{3}$ is too big.

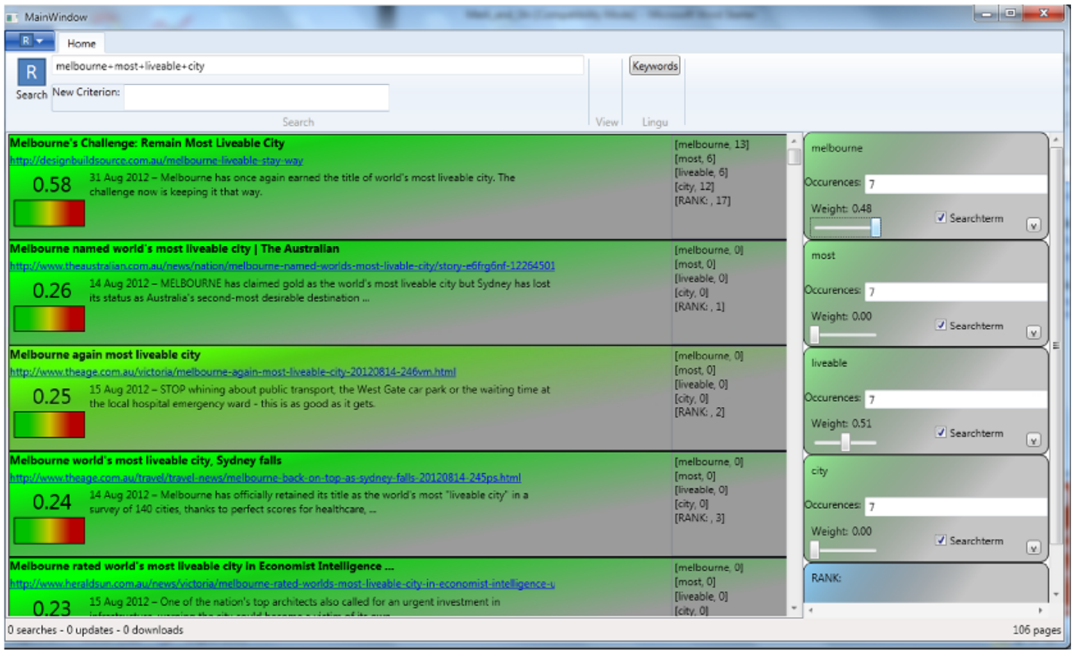
3.2 Stemming

To be able to properly count word occurrences, it needs to recognize a word in its different forms (e.g. house, houses, housing, …). Stemming finds the stem of a word, so that an occurrence for the stem can be counted. Doing stemming is not necessary to find the actual word stem, but to find the same stem for every form of a word (and for nothing else). Much research has been done in this field[10, 11, 12]. For illustration purposes, a very simple stemming algorithm is developed for the SearchBadger. The algorithm can be summarised as follows:

- Remove any special chars from the word
- Replace the ending “ier” by “y” (e.g. heavier -> heavy)
- Cut off the ending “er” (e.g. harder -> hard)
- Replace “gg” by “g” (e.g. bigger -> bigg -> big)
- Cut off any of this endings: “s”,”ly”,”ing”,”n” (e.g. houses -> house, friendly -> friend, growing -> grow)
- Replace ending “tion” by “te” (e.g. ignition -> ignite, distribution -> distribute)

3.3 Page re-ranking

In order to use the user’s feedback as one way to re-rank the result pages, a new Criterion called *FeedbackTopicCriterion* has been created. Once all pages are downloaded and the keyterm significance is calculated, it is ready to use. For every significant keyterm the average value of the user’s feedbacks of the pages on which the term is on is then saved. To re-rank the pages, the most significant keyterms are listed with their values. The page evaluation value used for ranking is the average of these keyterm values weighted by their significance. Figure 5 illustrates a typical example of a re-ranked page.



melbourne+most+liveable+city

New Criterion:

Melbourne's Challenge: Remain Most Liveable City
http://designbuildsource.com.au/melbourne-liveable-stay-way
0.58
31 Aug 2012 – Melbourne has once again earned the title of world's most liveable city. The challenge now is keeping it that way.
[melbourne, 13]
[most, 6]
[liveable, 6]
[city, 12]
[RANK: , 17]

Melbourne named world's most liveable city | The Australian
http://www.theaustralian.com.au/news/nation/melbourne-named-worlds-most-livable-city/story-e6frg6nf-12264501
0.26
14 Aug 2012 – MELBOURNE has claimed gold as the world's most liveable city but Sydney has lost its status as Australia's second-most desirable destination ...
[melbourne, 0]
[most, 0]
[liveable, 0]
[city, 0]
[RANK: , 1]

Melbourne again most liveable city
http://www.theage.com.au/victoria/melbourne-again-most-liveable-city-20120814-246vm.html
0.25
15 Aug 2012 – STOP whining about public transport, the West Gate car park or the waiting time at the local hospital emergency ward - this is as good as it gets.
[melbourne, 0]
[most, 0]
[liveable, 0]
[city, 0]
[RANK: , 2]

Melbourne world's most liveable city, Sydney falls
http://www.theage.com.au/travel/travel-news/melbourne-back-on-top-as-sydney-falls-20120814-245ps.html
0.24
14 Aug 2012 – Melbourne has officially retained its title as the world's most "liveable city" in a survey of 140 cities, thanks to perfect scores for healthcare, ...
[melbourne, 0]
[most, 0]
[liveable, 0]
[city, 0]
[RANK: , 3]

Melbourne rated world's most liveable city in Economist Intelligence ...
http://www.heraldsun.com.au/news/victoria/melbourne-rated-worlds-most-liveable-city-in-economist-intelligence-u
0.23
15 Aug 2012 – One of the nation's top architects also called for an urgent investment in
[melbourne, 0]
[most, 0]
[liveable, 0]
[city, 0]

melbourne
Occurences: 7
Weight: 0.48
[✓] Searchterm

most
Occurences: 7
Weight: 0.00
[✓] Searchterm

liveable
Occurences: 7
Weight: 0.51
[✓] Searchterm

city
Occurences: 7
Weight: 0.00
[✓] Searchterm

RANK:

0 searches - 0 updates - 0 downloads
106 pages

Figure 5. Re-ranked page by SearchBadger

4. Conclusion & Future Work

It is important to understand that, in terms of search optimization, SearchBadger is still in its infancy and may not be beneficial to inexperienced searchers. However, SearchBadger framework presents an opportunity for further development to optimise the existing search engines. New criteria can be developed quickly and integrated into the SearchBadger and evaluation of its efficiency can be carried out easily. As SearchBadger loads all result pages and stores it in a local database, it can also be used for lasting analyses on the page content and the subjective helpfulness to the user. SearchBadger could be an ideal platform for developing future semantic web.

REFERENCES

[1] Search Engine Market Share, http://www.netmarketshare.com/search-engine-market-share.aspx?qprid=4

[2] Web Browser Market Share, http://www.statowl.com/web_browser_market_share.php

[3] Goole Chrome,http://en.wikipedia.org/wiki/Google_chrome

[4] Usage Share of Web browsers,http://en.wikipedia.org/wiki/Usage_share_of_web_browsers

[5] Samsung Overtakes Apple, http://appleinsider.com/articles/12/05/01/samsung_overtakes_apple_to_claim_smartphone_market_share_lead.html

[6] Google Search, http://en.wikipedia.org/wiki/Google_search_engine

[7] WPF, http://en.wikipedia.org/wiki/Windows_Presentation_Foundation

[8] db4o, http://en.wikipedia.org/wiki/Db4o

[9] Microsoft Visual Studio, http://www.microsoft.com/visualstudio

[10] Lovins, Julie Beth (1968). "*Development of a Stemming Algorithm*". Mechanical Translation and Computational Linguistics 11: 22–31.

[11] Savoy, Jacques; *Light Stemming Approaches for the French, Portuguese, German and Hungarian Languages*, ACM Symposium on Applied Computing, SAC 2006,ISBN 1-59593-108-2

[12] Jongejan, B.; and Dalianis, H.; *Automatic Training of Lemmatization Rules that Handle Morphological Changes in pre-, in- and Suffixes Alike*, in the Proceeding of the ACL-2009, Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Singapore, August 2-7, 2009, pp. 145-153 [1]

Authors

Mr. Sebastian Merli is a final year Computer Science student at Technische Universität München in Germany. He was supervised by Associate Professor Hao Shi as a research intern at Victoria University from May 2012 to September 2012 and successfully completed the project with topic “Development of SEARCHBADGER, a framework for evaluation of search results”.

Dr. Hao Shi is an Associate Professor in School of Engineering and at Victoria University, Melbourne, Australia. She completed her PhD in the area of Computer Engineering at University of Wollongong and obtained her Bachelor of Engineering degree at Shanghai Jiao Tong University, China. She has been actively engaged in R&D and external consultancy activities. Her research interests include p2p Network, Location-Based Services, Web Services, Computer/Robotics Vision, Visual Communications, Internet and Multimedia Technologies.