CiteSearcher:
A Google Scholar frontend for Mobile Devices

Kevin Brandstatter
Illinois Institute of Technology
Computer Science Department
kbrandst@hawk.iit.edu

Ioan Raicu
Illinois Institute of Technology
Computer Science Department
iraicu@cs.iit.edu

ABSTRACT
Google Scholar is an invaluable resource in a quickly expanding digital world with an estimated 50,000,000 scholarly articles in existence. It lets one search for scholarly articles by title, author names, publishing venues, and keywords. Their web-based interfaces work great on desktop/laptop screens with 1024x768 resolution or better. However, many mobile devices such as phones and tablets have lower resolutions, with some being as low as 320x480. In order to support such low resolution devices, we have developed CiteSearcher, a front-end application for iOS and Android mobile devices, which communicates with Google Scholar to perform users' queries. CiteSearcher allow users to easily search Google Scholar for an author's work, and computes the publications impact through the H-index and G-Index. We deployed CiteSearcher on both iOS and Android-based devices.

Categories and Subject Descriptors
H.3.7 [**Digital Libraries**]: Collection, Dissemination, Systems issues, User issues; H.5.2 [**User Interfaces**]: Ergonomics, Graphical user interfaces (GUI), Screen design (e.g., text, graphics, color), Training, help, and documentation; K.3.2 [**Computer and Information Science Education**]: Computer science education, Information systems education.

General Terms
Management, Design.

Keywords
Citations, h-index, g-index, Google Scholar, iOS, Android

1. INTRODUCTION
With an estimated 50,000,000 scholarly articles in existence [1], it is really difficult to find relevant articles, as well as to keep track of our own articles and their impact. Luckily, tools such as Google Scholar [2] and Microsoft Academic Search [3] have both been available for some time to help the research community organize these millions of articles. Both tools let you search for scholarly articles by title, author names, publishing venues, keywords, and offer quite a rich experience.

Their web-based interfaces work great on desktop/laptop screens with XGA+ (1024x768) resolution (see Figure 1). However, many mobile devices such as iPods, smart phones, and tablets have lower resolutions, with some being as low as 320x480, making the results quite difficult to understand (see Figure 2). Furthermore, there is functionality that is not provided through such web interfaces, such as summarizing the total number of citations for an author, as well as computing an author's various impact metrics. These impact metrics include the Hirsch index (H-index) [4] and G-Index [5], amongst others. The index h is "defined as the number of papers with citation number ≥h", and is often used "as a useful index to characterize the scientific output of a researcher". For the G-Index, given a set of articles ranked in decreasing order of the number of citations that they received, the g-index is the (unique) largest number such that the top g articles received (together) at least g^2 citations.

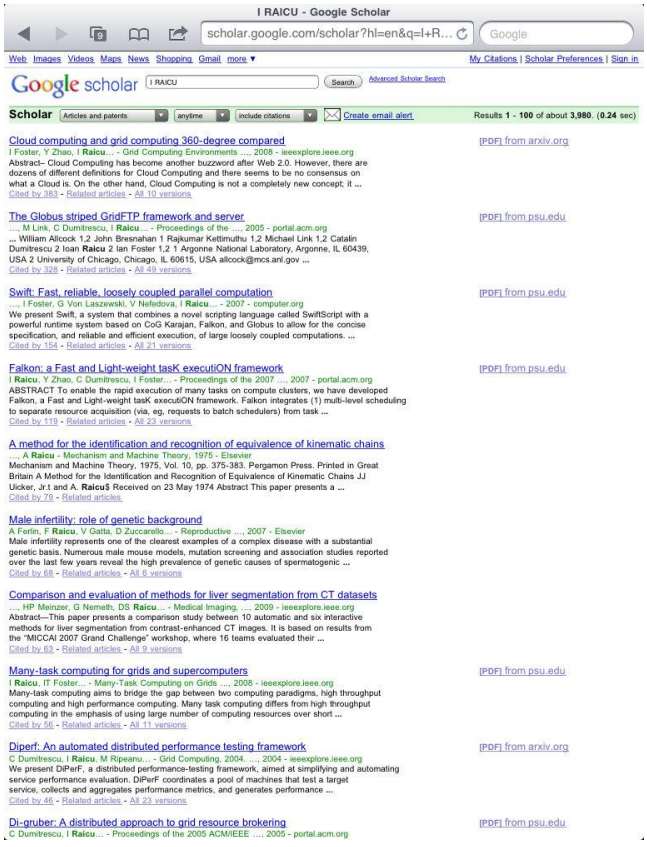

Figure 1: iPad2 Safari browser screen shot of Google Scholar results for query "I Raicu"; resolution of 1024x768

There is a well-known application, Harzing's Publish or Perish [6], which offers exactly this functionality (see Figure 3). Although this application has been run on Windows, Mac OS X (through WINE or virtual machine running Windows), and Linux

(through WINE), the application has not been ported to mobile devices. If someone needs access to this information from a phone and/or tablet, they would be limited to the web interface.

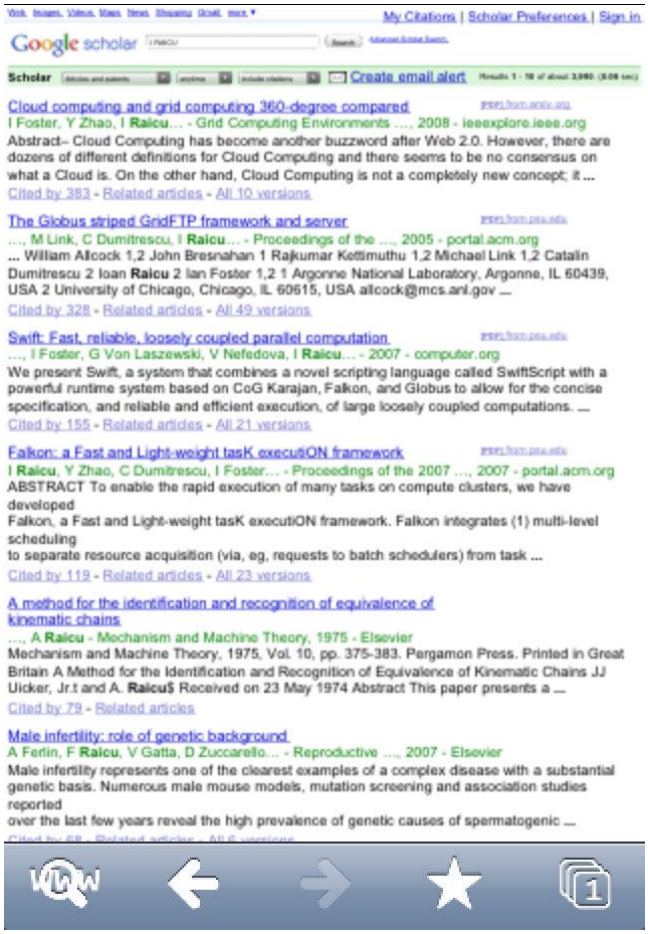

Figure 2: iPhone 3G browser screen shot of Google Scholar results for query "I Raicu"; note the fuzziness in the image due to the low resolution of 320x480

In order to support such low resolution devices, as well as offer a summary of the author's impact in terms of total citation counts, H-Index, and G-Index, we have developed CiteSearcher [7] (see Figure 5, Figure 6, Figure 7, Figure 8, and Figure 9), a front-end application for iOS (e.g. iPod Touch, iPhone 3G, 3GS, 4G, iPad, and iPad2) [8] and Android (e.g. Android v2.2 and later) [9] mobile devices, which communicates with Google Scholar [2] to perform users' queries. CiteSearcher allow users to easily search Google Scholar for an author's work, and computes their Hirsch index (H-index) and G-Index.

We have deployed this application on both iOS (on August 9th, 2011) and Android-based devices (on August 20th, 2011), and have received positive feedback from the community. The CiteSearcher website went public on August 22nd, 2011. On August 31st, 2011, we sent out the first announcements to the community at large about CiteSearcher, and we received positive feedback. On the day of the announcement and the following day, we received 315 web page views, 100 iOS downloads, and 50 Android downloads. In total, we have received 260 downloads. See Figure 10 for more details on the number of application downloads and web site activity.

2. RELATED WORK

The Harzing's Publish or Perish [6] (see Figure 3) is probably the most complete citation impact analysis software, and was the initial inspiration for CiteSearcher. It queries Google Scholar and returns the results and metrics. However, this application was limited by the fact that it had to be run from Windows, and there were no portable versions of it for mobile devices.

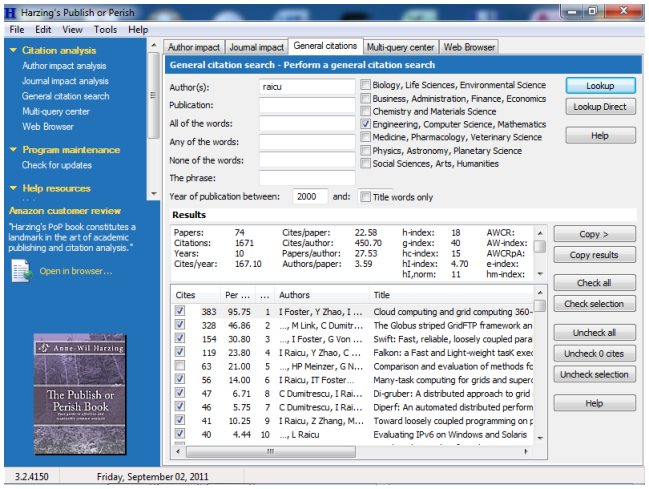

Figure 3: Harzing's Publish or Perish application [6]

scholarIndex [10] is a web script written in python. It has similar functionality to Publish or Perish, added portability of being web-based, which means any web enabled phone could use it as well. However with this approach the user is required to use a browser in order to use the application which makes for a decreased user experience, especially on a small resolution screen. Figure 4 shows a screen shot of scholarIndex on an iPhone 3G, and how tiny and fuzzy the web content is. Furthermore, the results page links articles to their respective publisher pages (e.g. IEEE or ACM Digital Library), which on mobile devices working over a 3G/4G wireless connection will almost always prompt users for a user id and password to gain access to the PDF electronic articles (note that many publishers use IP filtering to control access to their content, which means that a mobile device must be connected to a Wi-Fi connection at an institution that has paid access to the respective content). CiteSearcher links to other available sources (e.g. Citeseer, arxiv.com) which allows many times for the articles to be opened on the mobile device, regardless of the network the device is connected to.

iScholarReader [11] is an iOS application. Its main function is to aid in finding, saving, and reading articles in PDF format. It returns no information about the citation counts of individual papers or the impact of a particular author.

ScientistS [12] is another iOS application with a few interesting features. It allows the user to create a profile for an author, which in turn allows it to narrow the search queries than just a name. However, the results are returned in HTML format, which in turn are just as bad as those found in Figure 2 when viewed from a iPhone 3G phone.

Figure 4: iPhone 3G browser screen shot of scholarIndex results for query "I Raicu"; note the fuzziness in the image due to the low resolution of 320x480

3. CiteSearcher: a Google Scholar frontend for iOS and Android mobile devices

CiteSearcher is a Google Scholar frontend for iOS and Android mobile devices. With it, you can easily search Google Scholar for an author's work and their impact (H-index and G-Index). We have shown how poor the interface to Google Scholar on a mobile device such as an iPhone (see Figure 2 and Figure 4). Figure 5 shows the exact same results for query "I Raicu", but as displayed by the CiteSearcher user interface. Note the clean interface, including large font size and citation information. More details can be found on each article by clicking on the title of a particular article.

3.1 Google Scholar

The biggest challenge CiteSearcher had was how to communicate with the Google Scholar servers. Normally, when using Google in an application, one gets data by structuring a query with Google's web search API. Unfortunately, Google does not, as of yet, have a published API for Google Scholar. In order for CiteScholar to fulfill its intended functionality, it was imperative that we were able to access the articles cite counts, which are only returned with results from Google Scholar. This means that we had to search for an alternative method of acquiring the lists of results, and putting them in a useful format. In order to do this, we examined other applications (e.g. Publish and Perish [6] and scholarIndex [10]) that performed similar tasks.

Figure 5: CiteSearcher version 1.2 for iOS 3.0+ on iPhone 3G; results for query "I Raicu"

The starting point was Harzing's Publish or Perish application [6]. However, this proved to be of little use since we had no access to the source code. We tried sniffing network packets to get a clue about the query composition, but soon gave up after coming across a web based application called scholarIndex [10]. ScholarIndex is an application written in python and hosted on the Old Dominion University's Computer Science Departments web server that has similar functionality to Harzing's application. It queries Google Scholar and uses the results to calculate and authors h-Index. The notable difference with this application was that it was open source. After inspecting the source code, we found the request that was being sent to Google Scholar. From this, we were able confirm that the results were being generated from parsed HTML. Upon this discovery, we used scholarIndex's URL string as a basis for or own HTTP requests.

This led to a new problem surfacing. At first, the connection seemed fine and queries were being responded to, however we were getting indeterminate errors. After analyzing the network traffic, we determined that the possible cause was the user agent field of the HTTP request. We assumed that Google may be blocking requests from non-browsers, as we were getting HTTP request errors when we tried the same URL request from a simple WGET client. Upon this discovery, we found a way to alter the user-agent field of the HTTP request to imitate a browser, which resolved this set of issues.

Our last issue was the limited results per page that Google imposes (e.g. there is a maximum of 100 entries returned per

query). This caused all sort of problems, such as infinite loops in the index calculators if they were higher than the number of results returned, as well as incomplete results for authors with high number of articles and citations (e.g. I Foster). In order to prevent this, we added the ability for CiteSearcher to fetch additional results if needed. We also limit the results to 1000 entries, in case a query is made that is so generic (e.g. "a") that it returns too many results. Before implementing this feature, on several occasions, we were able to get Google to block the IP of the mobile device for sending automated queries, which would render the application useless (until a CAPTCHA [13] would be completed).

3.2 CiteSearcher Application

3.2.1 iOS version 1.2

The iOS version of CiteSearcher is simply a structured iOS applications written in Objective C, broken up into various views, and controlled by view controllers. CiteSearcher starts with a root view, which is the search bar and search results view. The view controller for this view handles most of the tasks of the application. CiteSearcher sends the specified queries to Google Scholar. The view controller also handles the parsing of the HTML results, using an Objective C HTML parser class [14]. The HTML parser was simply a set of classes that made use of the built in xml parsing libraries of the iOS SDK. We created an article class so that we could store the data in a uniform format. This allowed us to easily sort and store the article data for use with the rest of the program.

Once we had the data, we displayed the names of the articles in the table view that was a subview of the search view. We also set it so that when one of the results was selected the application would launch the article subview. This view displays the articles title, authors and has the links to the webpage result and the PDF version if available (see Figure 6).

Figure 6: CiteSearcher article view

Since the root window is managed by a navigation controller, the back button is displayed in the upper left corner in the toolbar, and that brings the main view back to the foreground. On the main screen has a menu button that allows the options to enter delete mode, as well as brings up the author view (see Figure 7). The author view is where the statistics about the author are presented. This is where the user will find the H-Index and G-Index calculations. Finally, an article can be opened directly in CiteSearcher if the PDF file is freely available online (see Figure 8), a handy feature that should save much time and resources, when compared to the alternatives.

Figure 7: CiteSearcher author information showing the total citation numbers, H-Index and G-Index for query "I Raicu"

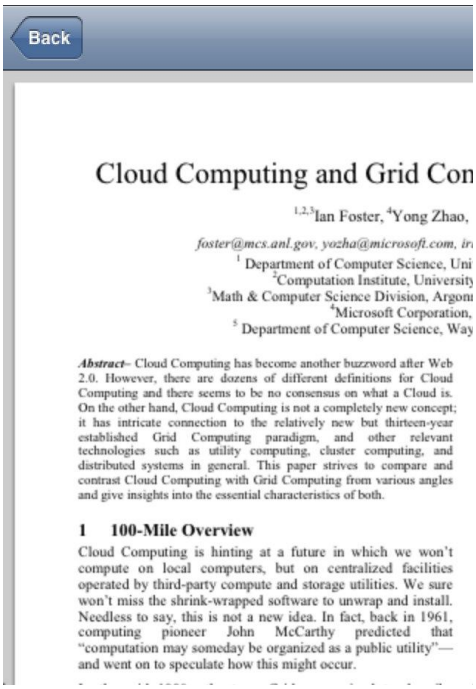

Figure 8: CiteSearcher vie of article in PDF format

3.2.2 Android version 1.0

The Android version of the application was created with the same basic structure as the iOS application. We have adopted the Java programming language as well as the new Eclipse IDE. For HTML parsing, we used the jericho library [15], which is not part of the Android SDK, but we found an instable JAR file. We used a different color scheme in the Android version from the iOS version, to easily differentiate between them (see Figure 9).

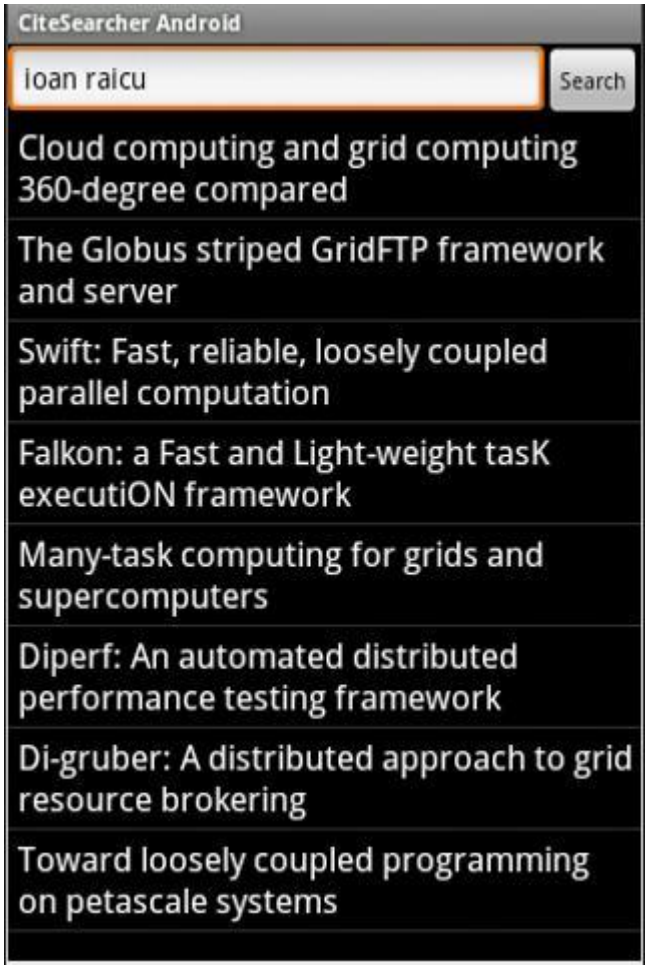

Figure 9: CiteSearcher version 1.0 for Android 2.2+; results for query "I Raicu"

3.3 Activity Monitoring

Although both Apple and Android allow us to keep track of downloads, we thought it would be interesting to keep track of the unique users that are repeatedly using CiteSearcher, as well the kinds of searches they are making. In order to record usage statistics about the application, we used a background HTTP POST request to send the data to a web server at Illinois Institute of Technology (specifically datasys.cs.iit.edu). On the web server, we run a php script that handles the incoming data. Once the data is received, it simply appends the data in XML format to a persistent file. Part of the information it records is the IP address of the mobile device where the query originated, in a hashed format; the IP address is hashed to preserve anonymity of the users, but yet still allow it to provide a unique user identification. The php script is simple, when it's requested, it binds the data in the message body and then writes it to the xml file. The posted data is sent in XML format. There is also a bug reporting form for people to submit problems with the application.

4. CONCLUSIONS AND FUTURE WORK

This paper described the development of a new tool CiteSearcher, a front-end application for Google Scholar, aimed at iOS and Android mobile devices. CiteSearcher allow users to easily search Google Scholar for an author's work, and computes their H-index and G-Index. We have deployed this application on both iOS and Android based devices with positive feedback from the community. Students, staff members, faculty members, and industry professionals who are involved with research find this kind of mobile application excellent in their arsenal of tools to interpret and find useful information in today's exponentially increasing world of online articles.

We have had hundreds of downloads already with only a few days since we officially started our advertising campaign (see Figure 10). We have high hopes for CiteSearcher, as long as the research community seems to think that this is a useful tool. We plan to add advanced search options, where one could specify advanced search options in order to reduce the number irrelevant search results. We also plan to implement some advanced filtering techniques, to help disambiguate common names. Finally, we plan on implementing an iPad native application, which makes use of the four times higher screen resolution. Note that the current CiteSearcher version 1.2 works on the iPad in compatibility mode, but the resolution is still the smaller iPhone resolution of 320x480.

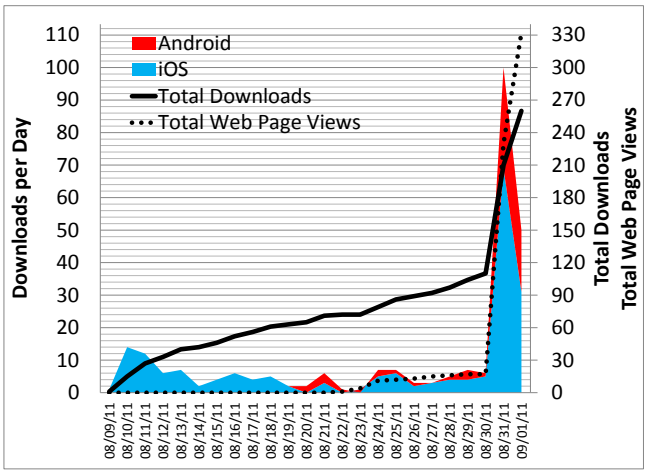

Figure 10: CiteSearcher application downloads (iOS and Android) and web site activity

5. ACKNOWLEDGMENTS

This work was performed in the DataSys Laboratory at Illinois Institute of Technology through a high-school research experience program, sponsored by the National Science Foundation grant NSF-1054974.

6. REFERENCES

[1] Arif Jinha (2010). Article 50 million: an estimate of the number of scholarly articles in existence Learned Publishing, 23 (3), 258-263 DOI: 10.1087/20100308

[2] Google Scholar; http://scholar.google.com/, 2011

[3] Microsoft Academic Search; http://academic.research.microsoft.com/, 2011

[4] J.E. Hirsch. "An index to quantify an individual's scientific research output". PNAS 102 (46): 16569–16572, 2005

[5] Leo Egghe. “Theory and practise of the g-index”, Scientometrics, vol. 69, No 1, pp. 131–152, 2006; doi:10.1007/s11192-006-0144-7

[6] Publish or Perish, http://www.harzing.com/pop.htm, 2011

[7] CiteSearcher, DataSys Laboratory, Illinois Institute of Technology; http://datasys.cs.iit.edu/projects/CiteSearcher/, 2011

[8] CiteSearcher, Apple App Store; http://itunes.apple.com/us/app/citesearcher/id453186643?mt=8, 2011

[9] CiteSearcher, Android Marketplace; https://market.android.com/details?id=datasys.iit, 2011

[10] scholarIndex; http://www.cs.odu.edu/~mln/pubs/2007-10-09-h-index.html, 2011

[11] iScholarReader; http://itunes.apple.com/us/app/ischolarreader/id380597575?mt=8, 2011

[12] ScientistS; http://itunes.apple.com/us/app/scientists/id397322548?mt=8, 2011

[13] L. Ahn, M. Blum, N.J. Hopper, J. Langford. "CAPTCHA: Using Hard AI Problems for Security". Advances in Cryptology — EUROCRYPT 2003. Lecture Notes in Computer Science. 2656. pp. 294–311, 2003

[14] Objective-C-HTML-Parser, An objective c wrapper around libxml for parsing HTML; https://github.com/zootreeves/Objective-C-HMTL-Parser, 2011

[15] Jericho HTML Parser, a java library allowing analysis and manipulation of parts of an HTML document; http://jericho.htmlparser.net/docs/index.html, 2011