# BotSeer: An automated information system for analyzing Web robots

Yang Sun, Isaac G. Councill, C. Lee Giles
College of Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802, USA
{ysun, icouncill, giles}@ist.psu.edu

## Abstract

*Robots.txt files are vital to the web since they are supposed to regulate what search engines can and cannot crawl. We present BotSeer, a Web-based information system and search tool that provides resources and services for researching Web robots and trends in Robot Exclusion Protocol deployment and adherence. BotSeer currently indexes and analyzes 2.2 million robots.txt files obtained from 13.2 million websites, as well as a large Web server log of real-world robot behavior and related analyses. BotSeer provides three major services including robots.txt searching, robot bias analysis, and robot-generated log analysis. BotSeer serves as a resource for studying the regulation and behavior of Web robots as well as a tool to inform the creation of effective robots.txt files and crawler implementations.*

## 1 INTRODUCTION

Web search engines, digital libraries, and many other web applications depend on robots to acquire documents. Web robots, also called "spiders", "crawlers", "bots" or "harvesters", are self-acting agents that continuously navigate through the hyperlinks of the Web, harvesting topical resources without significant human management cost [3, 4, 14].

Web robots are highly automated and seldom regulated manually. With the increasing importance of information access on the Web, online marketing, and social networking, the functions and activities of Web robots have become extremely diverse. These functions and activities include not only regular crawls of web pages for general-purpose indexing, but also different types of specialized activity such as extraction of email and personal identity information and service attacks. Even general-purpose web page crawls can lead to unexpected results for Web servers. For example, robots may overload the bandwidth of a small website such that normal user access is impeded. Robot-generated visits can also affect log statistics significantly so that real user traffic is overestimated.

Robot activities can be regulated from the server side by deploying the Robots Exclusion Protocol in a file called robots.txt in the root directory of a web site. The Robots Exclusion Protocol[1] is a convention that allows website administrators to indicate to visiting robots which parts of their site should not be visited. If there is no robots.txt file on a website, robots are free to crawl all content. The format of Robots Exclusion Protocol is described in [11]. A file named *"robots.txt"* with internet media type "text/plain" is placed under the root directory of a Web server. Each line in the robots.txt file has the format: $< field >:< optionalspace >< value >< optionalspace >$. A correctly formatted robots.txt file typically includes the "user-agent" field, the "disallow" field and the "allow" field. An unofficial field *Crawl-Delay* is also recognized by many crawlers to limit the frequency of robot visits.

Currently around 30% active websites deploy the Robots Exclusion Protocol [7, 10, 19]. Although it is not an enforced standard, ethical robots (including many commercial bots) will follow the rules specified in robots.txt. These rules may disallow access to certain locations within the website, and can be used to explicitly specify access preferences for specific robots by name, thereby biasing the availability of a particular site in favor of or against certain robots[18]. On the other hand, since a primary resource for webmasters of Web servers in designing effective robots.txt files is the Web access log, robots.txt files also reflect at a certain level the behavior of Web robots. The study of biases toward each Web robot can benefit the webmasters in designing an effective regulation policy for their own websites. Robot administrators can also benefit from

[1] http://www.robotstxt.org/wc/norobots.html

978-0-7695-3261-5/08 $25.00 © 2008 IEEE
DOI 10.1109/ICWE.2008.27

the bias study of robots in robots.txt files in designing their own robots in order to avoid potential privacy and security violations.

It is important to note that the Robots Exclusion Protocol is not an enforcement standard. Although access policies are explicitly specified in the robots.txt files, technically, Web robots are still able to access and download the contents even though they are forbidden for robots. Robot-generated Web server logs are an important resource for webmasters in monitoring web traffic and designing their websites. Since search engines have become an indispensable tool for information access, the design of websites must involve consideration of robot behaviors in order to ensure that websites are well indexed as well as to prevent undesirabled visits.

In order to help address the above problems, we propose BotSeer[2] as an informational service for analyzing Web robot behavior and trends in the Robots Exclusion Protocol deployment. BotSeer provides full-text search service of robot-related documents, bias analysis from robots.txt files collected from the Web, and a log analysis service. The prototype of BotSeer contains a full-text index of 2.2 million robots.txt files from 13.2 million websites. Users can employ BotSeer as a resource for studying the regulation and behavior of Web robots, a tool to write robots.txt files, and a resource for information on the development of crawlers.

The rest of the paper is organized as the following. Section 2 discusses the use cases and architecture of BotSeer system. Section 3 describes how the data for each component is gathered and processed. Section 4 introduces each component of BotSeer with sample use cases in detail. And finally section 5 concludes the paper and discusses the future work of the system.

## 2 BotSeer

BotSeer is designed to provide an efficient tool for researchers, webmasters and developers to study web robot related issues and design websites. The purpose of BotSeer is to build an information system that assists the regulation and development of web robots. The Robots Exclusion Protocol is a de facto standard on the Web to regulate the behavior of web robots. Websites can explicitly specify an access preference for each robot by name in their robots.txt files. Such biases data can be a valuable resource to the research of Web robot regulation and more. For example, a new type of network can be constructed based on the similarity of two websites in terms of bias toward web robots. Such network may present new perspectives of the Web graph based on similar privacy concerns and hidden community of webmasters or web designers. Thus, robots.txt files are an important data source for BotSeer.

We design three major components in BotSeer to assist these tasks. We implement the favorability measure [18] on all the robots.txt files we collected. The measure for each website and the statistics over all the websites provide researchers a more efficient tool for related research. A fielded full-text index of robots.txt files provides the necessary functionalities for studying the bias over different robots. A large scale log analysis of web access logs provides a valuable resource for identifying robots and characterizing them, hence developing more efficient regulations. The search components are based on the open source indexer, Lucene [6].

Architecture for BotSeer system is illustrated in Figure 1.

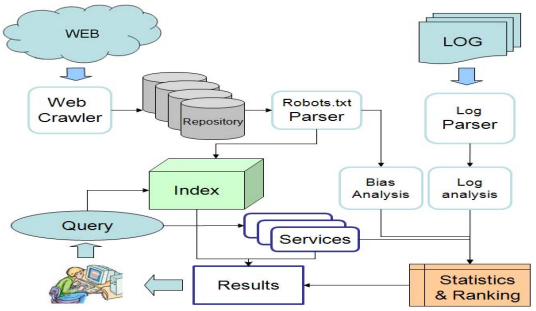



**Figure 1. The architecture of BotSeer system.**

All the services provided by BotSeer are based on the data obtained from the Web and Web server access logs. A Web crawler handles the data acquisition tasks including getting robots.txt files. The data crawled from the Web is saved in a local repository for further data processing. A data select module handles the data organization and feed the applications that need data inputs. The file parser implements the Robots Exclusion Protocol and parses each rule in robots.txt files to provide fielded searching and filtering for research. The parsed robots.txt files are then indexed using Lucene to provide efficient access to the documents. A fielded index is designed to store the fields extracted according to the Robots Exclusion Protocol as well as the bias analysis result for each robots.txt file. Robot generated Web access logs are analyzed by the log analysis module and the statistical results are stored in MySQL database. A Web based user interface provides the access to each services in BotSeer for

[2]Currently, the BotSeer search tool is available in Beta at http://botseer.ist.psu.edu

users. User queries are handled by a query processor with the support of multiple query modifiers. Three major services, robots.txt search, bias analysis and log analysis, delivers the data and analysis to users.

BotSeer system is planned to crawl 80 million top level domains in the first phase and their subdomains in the second phase. BotSeer updates the robots.txt files in a monthly basis to capture the temporal properties. With the growing size of robots.txt file index, prioritized updating policy is to be applied to BotSeer index. The system is built on many open source development tools including JDK, Apache Tomcat, MySQL, and Lucene which allow a highly scalable system. Current BotSeer system handles complicated search queries and returns results from 2.2 million documents within 1 second. The system is expected to be able to index robots.txt files for 200 million websites.

## 3 DATA

BotSeer provided services are based on robots.txt files and Web server logs. The acquisition, storage and access of each data source is discussed below.

### 3.1 Robots.txt Files

For the use cases discussed previously, robots.txt files are a primary data source for BotSeer to provide the services. It has been reported on the Netcraft web server survey that there are more than 142 million active websites in October 2007. The crawler for BotSeer is required to be able to check a significant portion of all the active websites and find robots.txt files in a short period of time in order to provide up-to-date analyses and information regarding robot regulation. Traditional Web crawlers rely on parsing Web pages and following hyperlinks to traverse the Web graph. This method usually requires a great deal of resources in order to parse the content of Web pages. This is inefficient and unnecessary for acquiring and parsing robots.txt files since valid robots.txt files must always be placed in a standard location relative to the root domain of a website. In addition, relying entirely on links in the Web graph may fail to identify sites that fall in disconnected components within the web graph.

Instead, we use a java based automated agent to collect robots.txt files without explicitly traversing the link structure of the Web. We first collect the active top level domain names from the .com, .net, .org, and .biz domains from their respective authoritative root providers, yielding over 80 million distinct hostnames. The robots.txt files are checked under the root directory of these websites and then downloaded directly from each domain where they exist. Our assumption is that most websites from the above domains will be acquired via this method; however, our robot will not locate and index robots.txt files outside of the root providers. It is an open question whether bots will ever see these files in any case.

A crucial issue for crawling robots.txt files in this manner is that the requests are always pointing to unique domain names. Thus, a single DNS server will not be able to handle the heavy request load (>100 name resolving requests per second beyond a day). We tried several DNS servers within the network service range. All of them stop responding after a few minutes of heavy requests. A DNS request is a major cost in the web crawling process. DNS requests are typically determined by the connection bandwidth between the client and the server. Although a name resolving request over a socket connection can be very fast (several milliseconds), the major cost will be spent in constructing the connection. Multiple DNS servers can be used to feed multithreaded name resolving requests. But these DNS servers will quickly stop responding because of the requirements for crawling speed. Another problem is that some of the available DNS servers do not correctly resolve hosts (e.g. pointing to a specific website or localhost). It is therefore necessary to check the correctness of each DNS server before using it. If all the checks are implemented together in one crawling task, the crawling performance can be significantly slowed. An experiment shows that only 10-20 websites can be checked in one second for this crawling method on a 3.0GHz Xeon PC. Since millions of websites need to be checked, it will take months to check most of the active websites.

In order to address the issues of mass DNS requests, an asynchronous DNS resolving method is implemented into the BotSeer robot which separates the DNS server checking, the name resolving and the document downloading tasks. A DNS resolving module tests available DNS servers by triangulating multiple known resolving results and simultaneously recording the response time. The correctly available DNS servers are stored in a pool sorted by the response time. A name resolving module reads all the domain names that need to be resolved and sends them to the DNS pool. The best available DNS server will be used to resolve a domain name and then be put into a rest pool until it can be reused. For all the resolved domain names, the robots.txt files can be checked and downloaded by constructing IP-based socket connections to the websites. Such a method provides a lightweight task for each crawling thread and a fast socket connection over IP addresses. We successfully obtain a process-

ing speed result (checking and downloading robots.txt files if available) of 100 websites in one second with this crawling method. We are able to check 13,257,110 websites and download 2,264,820 within a few days on a single Intel 2.8GHz Xeon PC with 1GB RAM (disk access often delays the process in practice). BotSeer also provides a submission system for users to submit robots.txt files directly to the system. The user submission system is integrated with the bias analysis module so that when a user submits a website, it is indexed with the bias analysis result is provided. Since each robots.txt file relates to a unique domain name, the information regarding each domain (whether the domain has a robots.txt file or not) is also an important resource for the study of robot regulation. The domain information is stored in MySQL database tables for further analysis.

We also propose to provide access to past robots.txt for longitudinal research. The robots.txt files are stored based on the date that they were downloaded. Since robots.txt files tend to be modified for privacy and security purposes, it is of value to track the changes in robots.txt files to investigate temporal trends. However, storing robots.txt files from each crawl will be redundant since a significant percentage of websites will not show frequent changes in the file. This issue is handled by storing only unique files for each site. A data selection middleware can select a set of robots.txt files according to temporal criteria.

### 3.2 Web Server Logs

Web server access logs are one of the most important resources used to study Web robot behavior since logs have all the access and activity information of generated by robots. A typical Web server access log record includes the connection IP address, the requested files, the identity of a robot, and the time of access. Extracting the robot-generated log records from Web server logs is a non-trivial problem since not all robots identify themselves as such, instead pretending to be normal user-driven browsers. Much research has been done in the area of log mining[1, 5, 12, 16, 17] that mentions the robot identification in the log data preprocessing step. Since most of the experiments was conducted before 2000, the spambot and bot masquerading issues are not as serious as they are today. None of the work deals with the robot identification carefully in terms of accuracy.

In BotSeer we implement a multi-step log filter in order to identify web robots. The first step checks the User-Agent string field recorded in the logs. Basically, we used a robot agent string list[3] to identify obvious robots. The second step involves a sophisticated log mining method that extracts a set of features to characterize the browsing behavior of an IP address based on sessions. The sessions are identified by correlating the request and reference fields with page structures in CiteSeer.

For the study of robot behavior, the log analysis service provides statistical data including the total visits, number of requests that disobey robots.txt rules, unique IP addresses with the same User-Agent name, total bandwidth used by robots and the overall favorability of a robot.

## 4 WEB APPLICATION

### 4.1 Robots.txt File Search

The goal of BotSeer is to provide information to anyone interested in studying robot behavior and regulation. The design of the BotSeer web application considers several possible use cases for these users. The most anticipated use case is the search for named robots in robots.txt files, and biases associated with specific robots. The BotSeer response to the query "botname:msnbot" is shown in Figure 2. The search results indicate how sites generally treat specific robots. Different ranking methods are required for each purpose. For the use case of analyzing the robots.txt files, users are more interested in the robots.txt files from well used website. On the other hand, for the use case of studying how the robot is received, users are more interested in the websites that favor or disfavor the robot. The PageRank based ranking method ranks the results by combining the PageRank score [13] of websites and the tf-idf scores [15] between the query term and the robots.txt files. The bias based ranking method ranks the results by the bias of the named robot presented in robots.txt files. Users are able to switch between ranking methods when they are searching for the User-Agent field for different use cases.

For each returned robots.txt file, BotSeer presents the link to the URL of the robots.txt file, the abstract and cache of the file, and the link to the bias analysis of the file. Users may check the cached robots.txt file as well as link to the original source directly. By clicking the link to bias analysis, users are able to obtain the biases presented in the robots.txt file (the bias analysis is discussed in [18]). The search results give users references on how webmasters regulate the crawler msnbot. The operator of msnbot is also able to see how their robot is received.

[3] http://www.pgts.com.au/pgtsj/pgtsj0208d.html


**Figure 2. BotSeer robots.txt search component response to query "botname:msnbot".**

The fielded index provides a wide range of opportunities for BotSeer use cases. Users may specify which field of the robots.txt they are searching for by adding search modifiers including "botname", "site", "disallow", "allow", "favor", "disfavor", "nobias" and "delay". Each modifier refers to a specific field in the full-text index. "botname" refers to the "User-Agent" field which identifies the name of a robot. For example the query "botname:msnbot" searches "msnbot" in the User-Agent fields of all robots.txt files. "site" refers to the website names from which the robots.txt files are collected. "disallow", "allow" and "delay" refer to the corresponding directives in robots.txt files. We also support the modifier "favor", "disfavor" and "nobias" to enable the search of favorability (bias) presented by robots.txt files. With the fielded index, it is possible to study some of the topics we mentioned in section 2 by using BotSeer. For example, a user can search for "favor:googlebot site:gov". BotSeer will return all the government websites that favor googlebot. For the second use case in section 2, user can search for "contain:universal site:gov disallow:/" to return all the government websites that have the rule of "Disallow:/" and "User-Agent:*" and then find the websites needed with simple further processing. It is also easy to compare the favorability of two or more robots. User can search for "favor:googlebot disfavor:msnbot disfavor:slurp" which refers to the websites that favor "Google" but disfavor "MSN" and "Yahoo".

Google.com also indexes robots.txt files. Google users are able to search in the robots.txt files by issuing the query with modifier "inurl:robots.txt filetype:txt". Unlike the BotSeer search component which parses robots.txt files according to the Robots Exclusion Protocol, Google search indexes robots.txt files as normal plain text files. A comparison of Google search results for "msnbot" and BotSeer results shows that BotSeer provides more information on the robots from a regulation perspective. (On BotSeer homepage, users can click on the Google button to perform a robots.txt file search in Google.)

## 4.2 Crawler Search

The robot behavior search component searches the bias analysis results and robot generated logs from a large Web server. The log is generated by 8,932 robots from 61,204 IP addresses. This component is designed for users to check the real-world behavior of a named robot. Users may submit a robot name as the query to the system. The system matches the query to the bias analysis records and log records generated by corresponding robots and returns statistical data for the robots.

The result page lists the records in the log with bias toward a crawler, User-Agent strings contain the name of the crawler, and the geographical distribution of the crawler's IP address (see Figure 3). When users click on the User-Agent string, BotSeer will present the detailed information about this User-Agent string including the type of the crawler, the actual IP of the crawler, and the registered name of the IP addresses. If the registered name of IP addresses does not match the name of the crawler, user will know this is a potential crawler using fake names.

Since the User-Agent field in the log is specified by the robots in the HTTP request header, it is possible for any robot to "pretend" to be and use the name of another robot. But the IP addresses where the robots are actually from are harder to spoof. It is necessary to check the IP addresses as well as the name specified in the User-Agent string to identify a robot. From the example shown in Figure 3, part of the log records are generated by the robots who named themselves as Googlebot while we believe they are not. Users can click on the IP addresses listed on the result page and BotSeer will show users the WHOIS information[4] about that IP address. For the example of "Googlebot" records, the IP addresses that obeyed the Robots Exclusion Protocol in fact those allocated to Google Inc. We also plot the geographical locations of each visiting robot. As an example, figure 4 shows the geographical distribution of robots that visit CiteSeer. Each gray point represents a unique robot IP address. We use the blue color to circle out "Googlebots" that obey the robots.txt of CiteSeer and red color to circle out fake "Googlebots" that disobey the robots.txt.

We also design a honeypot, a set of sites where each site is configured with a distinct regulation specifica-

[4]Whois information is provided by GeekTool.com.

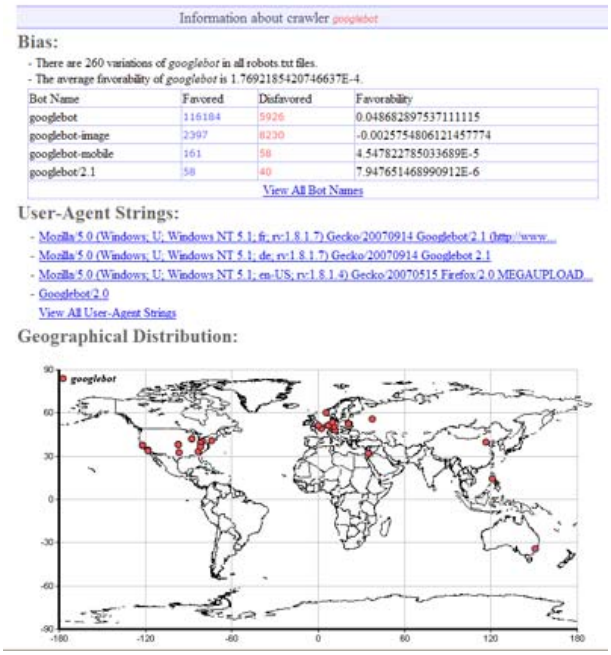



**Figure 3. The crawler search result page for query "googlebot".**

tion using the robots exclusion protocol in order to capture specific behaviors of web crawlers. A detailed log analysis on the test websites will reveal how well a web robot behaves. The analysis of honeypot logs will provide rules to measure the ethicality of web crawlers. The analysis of web access logs is a very important part of BotSeer not only for searching robot behavior but also for possible new regulation standards. Much data could be generated by combining the bias analysis results and the log analysis results to study the true behavior of web robots. With the log analysis results, BotSeer can also be used as a reference to identify poorly behaved robots and robots pretending to be browsers and other well known robots. Therefore, BotSeer can be a service to the webmaster community.

## 4.3 Data Analysis

BotSeer provides two types of statistical data analysis. The bias analysis module analyzes the bias presented in the robots.txt files, indicating how specific robots are regulated on the Web. Bias metrics can also be used to evaluate the reputation of specific robots. The robot-generated log analysis provides the statistics of the robots' actual behavior on a Web server.

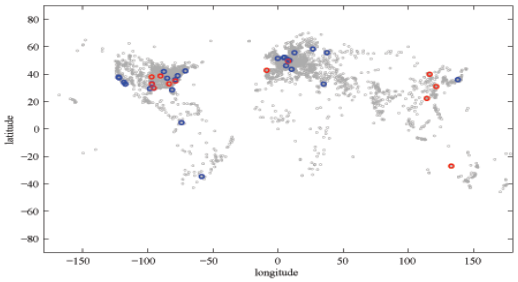



**Figure 4. The geographical distribution of web robots that visit CiteSeer. Gray points are the location of robots that visit CiteSeer. The blue and red circles point out the well behaved and bad behaved "Googlebot" respectively.**

### 4.3.1 Bias Analysis

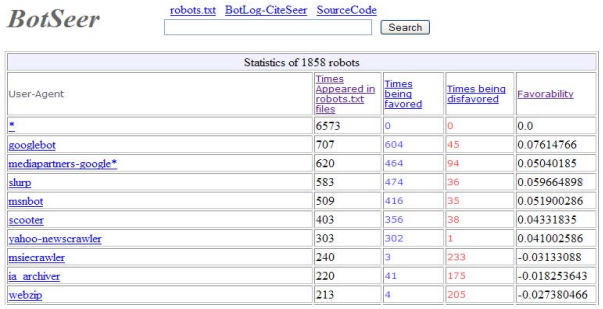
Websites can explicitly specify an access preference for each robot by name. Such biases can be used as a resource to study the regulation of Web robots. The bias analysis module analyzes the biases in robots.txt files based on the bias measure [18]. A favored robot is defined as a robot allowed to access more directories than the universal robot according to the robots.txt file in the website. The universal robot "*" is any robot that has not matched any of the specific user-agent names in the robots.txt file. In other words, the universal robot represents all the robots that do not appear by name in the robots.txt file. The detailed discussion about the bias analysis algorithms and results can be found in [18]. BotSeer implements the bias definition and provides the bias analysis for each robots.txt files. Users are able to get the bias analysis result by following the hyperlink of "Analyze it!" on the search result page of robots.txt files. The hyperlink will lead users to the robots.txt analysis result page.

BotSeer system also analyzes the overall bias on a set of robots.txt files based on the favorability defined in [18]. An example of a favorability analysis on 1,858 web robots is shown in Figure 5.

Users can click on each column title, and the result can be ranked by the times the robot's name appeared, the times it was favored, times disfavored, and favorability respectively. The hyperlink of the robot name will direct users to the log search page.

### 4.3.2 Robot Generated Log Analysis

The robot-generated log analysis component parses the Web log file from BotSeer and indexes the user-agent



BotSeer robots.txt BotLog-CiteSeer SourceCode Search

Statistics of 1858 robots

| User-Agent | Times Appeared in robots.txt files | Times being favored | Times being disfavored | Favorability |
|---|---|---|---|---|
| * | 6573 | 0 | 0 | 0.0 |
| googlebot | 707 | 604 | 45 | 0.07614766 |
| mediapartners-google* | 620 | 464 | 94 | 0.05040185 |
| slurp | 583 | 474 | 36 | 0.059664898 |
| msnbot | 509 | 416 | 35 | 0.051900286 |
| scooter | 403 | 356 | 38 | 0.04331835 |
| yahoo-newscrawler | 303 | 302 | 1 | 0.041002586 |
| msiecrawler | 240 | 3 | 233 | -0.03133088 |
| ia_archiver | 220 | 41 | 175 | -0.018253643 |
| webzip | 213 | 4 | 205 | -0.027380466 |

**Figure 5. Bias analysis result page of 1,858 named robots.**

strings and corresponding IP addresses and behaviors. The visits distribution and geographical distribution of robot-generated logs can be monitored through the log analysis system [5].

## 5 CONCLUSION

We present the design and data analysis process of the BotSeer system. The Web-based information system provides resources and services for researching Web robots and trends in Robot Exclusion Protocol deployment and adherence. BotSeer currently indexes and analyzes 2.2 million robots.txt files obtained from 13.5 million websites, as well as a large Web server log of real-world robot behavior and related analyses. BotSeer provides robots.txt search, robot bias analysis, and robot-generated log analysis services to aid the Web robots related studies and developments. With the large amount of data and fielded index, BotSeer can be used to search the favorability of web robots, compare web robots in terms of how well they are received, study web robot regulations in a specific group of websites, study the relationships between the robots regulation and the market, and find new communities and networks based on the preferences for certain web robots. BotSeer serves as a resource for studying the regulation and behavior of Web robots as well as a tool to inform the creation of effective robots.txt files and crawler implementations. Future work will focus on more advanced data analysis based on BotSeer. Other Web robot-related applications using BotSeer data will also be a direction.

## References


[1] B. Berendt and M. Spiliopoulou. Analysis of navigation behaviour in web sites integrating multiple information systems. *The VLDB Journal*, 9(1):56–75, 2000.

[2] M. L. Boonk, D. R. A. d. Groot, F. M. T. Brazier, and A. Oskamp. Agent exclusion on websites. In *Proc. of The 4th Workshop on the Law and Electronic Agents*, 2005.

[3] S. Chakrabarti, M. Van den Berg, , and B. Dom. Focused crawling: A new approach to topic-specific web resource discovery. In *Proc. of the 8th WWW Conference*, pages 545–562, 1999.

[4] J. Cho, H. Garcia-Molina, and L. Page. Efficient crawling through url ordering. In *Proceedings of the 7th International World Wide Web Conference*, 1998.

[5] R. Cooley, B. Mobasher, and J. Srivastava. Data preparation for mining world wide web browsing patterns. *Knowledge and Information Systems*, 1(1):5–32, 1999.

[6] D. Cutting. The lucene search engine. http://lucene.apache.org/.

[7] M. Drott. Indexing aids at corporate websites: The use of robots.txt and meta tags. *Information Processing and Management*, 38(2):209–219, 2002.

[8] D. Eichmann. Ethical web agents. *Computer Networks and ISDN Systems*, 28(1-2):127–136, 1995.

[9] C. L. Giles, K. Bollacker, and S. Lawrence. CiteSeer: An automatic citation indexing system. In I. Witten, R. Akscyn, and F. M. Shipman III, editors, *The Third ACM Conference on Digital Libraries*, pages 89–98, Pittsburgh, PA, 1998. ACM Press.

[10] B. Kelly and I. Peacock. Webwatching uk web communities: Final report for the webwatch project. *British Library Research and Innovation Report*, 1999.

[11] M. Koster. A method for web robots control. In *the Internet Draft, The Internet Engineering Task Force (IETF)*, 1996.

[12] B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization based on web usage mining. *Commun. ACM*, 43(8):142–151, 2000.

[13] L. Page and S. Brin. The pagerank citation ranking: bring-ing order to the web. In *Tech. report SIDL-WP-1999-0120, Stanford University*, 1999.

[14] G. Pant, P. Srinivasan, and F. Menczer. *Crawling the Web*, chapter Web Dynamics. Springer-Verlag, 2004.

[15] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.

[16] M. Spiliopoulou. Web usage mining for web site evaluation. *Commun. ACM*, 43(8):127–134, 2000.

[17] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: discovery and applications of usage patterns from web data. *SIGKDD Explor. Newsl.*, 1(2):12–23, 2000.

[18] Y. Sun, Z. Zhuang, I. G. Councill, and C. L. Giles. Determining bias to search engines from robots.txt. In *Proceedings of the 2007 IEEE/WIC International Conference on Web Intelligence*, San Jose, CA, USA, 2007. IEEE Computer Society.

[19] Y. Sun, Z. Zhuang, and C. L. Giles. A large-scale study of robots.txt. In *Proc. of the 16th international conference on World Wide Web*, 2007.


[5] http://botseer.ist.psu.edu/stats/logmonitor.jsp