Real-time Web Crawler Detection

Andoena Balla
Department of Computer Science
University of Cyprus
1678 Nicosia, Cyprus

Athena Stassopoulou
Department of Computer Science
University of Nicosia
1700 Nicosia, Cyprus

Marios D. Dikaiakos
Department of Computer Science
University of Cyprus
1678 Nicosia, Cyprus

***Abstract*—In this paper we present a methodology for detecting web crawlers in real time. We use decision trees to classify requests in real time, as originating from a crawler or human, while their session is ongoing. For this purpose we used machine learning techniques to identify the most important features that differentiate humans from crawlers. The method was tested in real time with the help of an emulator, using only a small number of requests. Our results demonstrate the effectiveness and applicability of our approach.**

I. Introduction

A web robot (crawler) is a program that traverse the Web autonomously. The crawler starts with a list of web addresses (URLs) called "seeds" and recursively visits hyperlinks accessible from that list. The purpose of crawlers is to discover and retrieve content and knowledge from the Web on behalf of various Web-based systems and services. For example: *search-engine crawlers* seek to harvest as much Web content as possible on a regular basis, in order to build and maintain large search indexes [3], [2]; *shopping bots* crawl the Web to compare prices and products sold by different e-Commerce sites; *focused crawlers* seek and acquire Web-pages belonging to pre-specified thematic areas [4]; *email harvesters* collect email addresses on behalf of email marketing companies or spammers, and *site-specific crawlers* perform various Web-site maintenance chores, such as mirroring Web sites or discovering their broken links.

The need to differentiate web crawlers from human users arises from the following:

1) The amount of traffic caused by crawlers may result to performance degradation of busy web servers.
2) E-commerce web sites may not wish to serve incoming HTTP requests from unauthorized web crawlers.
3) In human-user profiling using data mining of log files, requests originating from crawlers must be excluded otherwise misleading results regarding the navigational patterns of real users can be derived.
4) Pay-per-click advertising can be seriously harmed by click fraud [5], [15], [1], which involves among other things the unwilling or malicious repetitive "clicking" on advertisement links by Web robots.

Crawler detection is an important and challenging problem [17], and several published works have focused on crawler characterization [6], [7] and the off-line discovery of crawlers [13], [11], [12], [10], [9]. In this paper we present the design and implementation of a system that detects robots *on-the-fly in real-time* as opposed to off-line. Real-time detection offers the benefit of timely identifying robots and evaluating their behavior, so that unwanted crawlers can be contained as early as possible during their visit. We build on the crawler detection work of Stassopoulou and Dikaiakos [12], and use their Bayesian network to distinguish between crawlers and humans and then use statistical methods to extract the properties and features of crawler-induced web sessions. These features are then combined with weights to develop a system capable of identifying crawlers in real time.

The rest of the paper is structured as follows: Section II presents an overview of our approach which includes a description of the pre-processing steps and feature identification phase using statistical techniques. Section III describes the architecture of the system developed for real-time crawler detection. Experimental results are presented in Section IV, and we conclude in Section V.

II. Overview

Before implementing the real-time detection system, certain pre-processing steps needed to be accomplished, namely:

1) Characterization of crawler sessions using the off-line detection system developed by Stassopoulou and Dikaiakos in [12]:
   - Access-log analysis and session identification.
   - Extraction of session features to be used in the Bayesian network.
   - Learning of the Bayesian network parameters.
   - Classification of sessions into crawler or human.
2) New features extracted from the now classified sessions.
3) Statistical analysis of these features.
4) Selection of the most discriminant features to be used by the real-time detection system.

Feature Extraction: Following the off-line identification of crawler and human sessions using the system presented in [12], we extracted some new features for each session for the purpose of characterizing crawlers through the properties of their sessions. The extracted features from each session were as follows:

- *Percentage of HEAD requests*: HTTP GET requests are used to retrieve web-page content whereas HTTP HEAD requests retrieve web-page metadata. It is expected that "polite" crawlers would use the HEAD method, when

possible, in order to detect and download only recently updated pages, so as to minimize the consumption of Web-server resources.

- *Percentage of 2xx responses*: Prior studies show that human-induced sessions tend to have a higher percentage of responses with 2xx code [7].
- *Percentage of 3xx responses*: Prior studies show that human-induced HTTP requests receive more frequently 304 response codes, indicating that browsers dispatch `if-modified-since` requests more often that crawlers.
- *Percentage of page requests*: This denotes the percentage of htm, html, php and jsp requests.
- *Percentage of night requests*: This feature takes into account sessions that started between 2am and 8am local time.
- *AvgTime:* This feature denotes the average time between two consecutive htm, html, php and asp requests.
- *StdDevTime:* Considering that the average can be a misleading feature (due to outliers), we also calculate the standard deviation for the above feature.
- Percentage of requests for zip, $Multimedia$, $ascii$ and $Binary$ files.

In the following section we present the statistical analysis of these features in order to identify the most discriminant ones to be used by our real-time crawler detection system.

Feature Analysis: The statistical analysis of features relied on the classification of session using the Bayesian network of [12]. We have used only sessions that were classified with probability of belonging to a class (human or crawler) higher than 80 %. Out of the 270,000 sessions, 70,000 were classified as crawlers and 180,000 classified as humans. We used the SPSS (Statistical Packages for the Social Sciences) package to analyze the features utilizing various statistics such as the QQPlot, means and crosstabs.

The results of the analysis show that in the majority of the features, the behavior of crawlers is similar to that of human users. The most discriminant features, whose distributions vary significantly in the two classes, crawlers and humans, are the following:

- Percentage of Image requests. As it can be seen from figure 1, almost all human induced sessions have at least 10% requests for images, whereas only 1% of crawler sessions contain image requests.
- Percentage of page requests (% of html, htm, php, jsp requests). In figure 2, we can see that the majority of crawler sessions have a percentage of page requests higher that 60%, compared to that of human sessions which is lower (only 6%).
- Percentage of 4xx response code. Our results showed that 35% of crawlers have a percentage of 4xx response code higher than 20% compared with humans which is approximately 4%.
- Session time-out threshold. Our analysis shows that in 75% of the crawler sessions, the time between two consecutive page requests (html, htm, php, asp) is less than 50 seconds.
- Maximum click-rate. The maximum click rate in 99% of human user sessions does not exceed 10 clicks per minute whereas 60% crawler sessions have a click rate higher than 10.

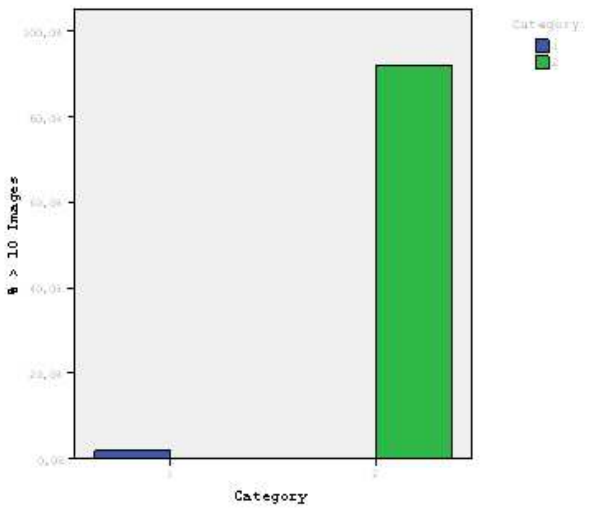

Fig. 1. Percentage of Image requests higher than 10%

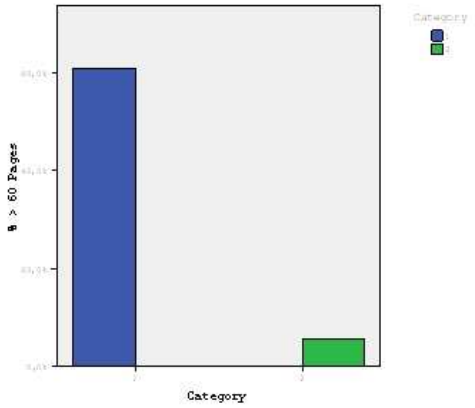

Fig. 2. Percentage of page requests higher than 60%

These features will be used in the implementation of our real-time detection system, which is described in the following sections.

III. Real-time Crawler detection

In this section we present the system developed for detecting crawlers in real-time. The system acts as a filter between client and server. It receives incoming http requests and processes them based on the features identified in the the pre-processing stage described earlier. The system then classifies the incoming requests as crawler or human induced, and in the former case it registers the IP address in a list of discovered robots. A detailed description of the system architecture follows.

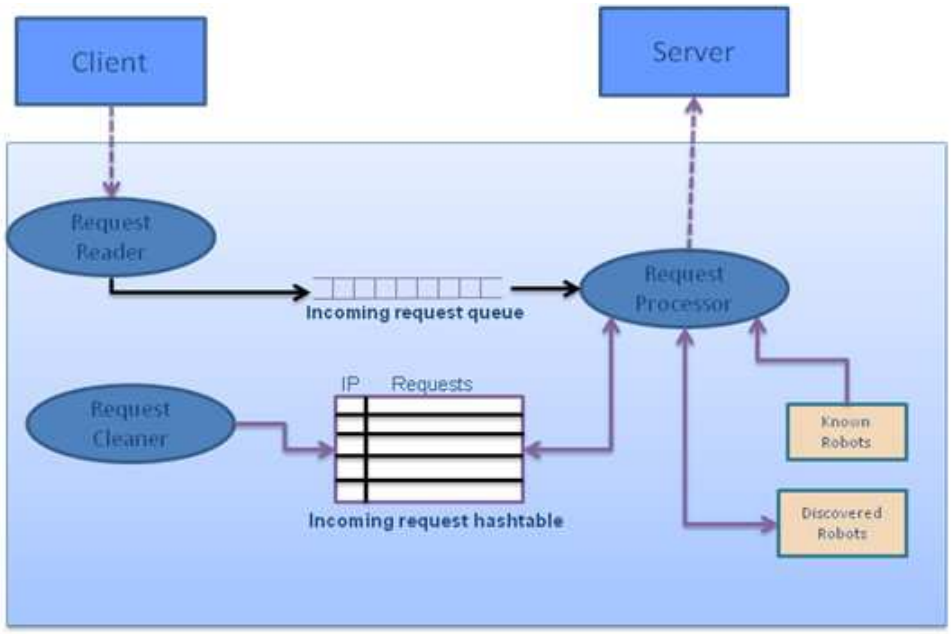

Fig. 3. The architecture of the real-time crawler detection system

System Architecture: The system consists of three threads shown in Figure 3. Firstly, the *Request Reader* receives the HTTP requests arriving at a web server and inserts them in a queue (FIFO structure), the *Incoming Request Queue*. The *Request Processor* is responsible for the classification of the incoming requests as crawler or human induced. In order to enable a classification of an IP address, there must be a minimum number of requests from that IP address. Until this is achieved, the incoming requests are inserted into a hash table (the *Incoming Request Hashtable*). The requests are indexed in the hash table based on their IP address. *Known Robots* is a file containing the list of IP addresses of known robots. *Discovered Robots* contains all IP addresses that have so far been classified as crawlers by the Request Processor. Finally, the *Request Cleaner* deletes all non-active requests. This ensures that the memory requirements of the crawler detection system are kept low. In the following subsection we describe in detail the most important module of the system, the *Request Processor*.

The Request Processor: As mentioned above, the Request Processor is the subsystem responsible for classifying incoming requests into crawler or human induced. As seen from figure 4, it reads a request from the Incoming Request Queue and then checks whether the IP address is from a known robot. If this is the case, then the IP address is added to the list of discovered robots and moves on to the next request from the queue. If the IP address is not in the list of known robots, it checks whether it is in the list of discovered robots, in which case the request is classified as robot and the processor moves on to the next request. However, if the IP address is not in the Discovered Robots list, then the IP address is added to the hash table. If this is the first request from this IP address, the system simply moves to the next request from the queue, otherwise it gets the latest active session from this IP and performs an analysis of the session using a decision tree. The last active session (Figure 5) is estimated as follows: We assume $n+1$ requests from the IP address (r_0 to r_n) sorted in terms of their arrival time. Starting from the latest request r_n, we check whether the elapsed time between this request and r_{n-1} is less than 2 minutes, in which case the requests belong to the same active session. This continues until the time difference between the current request and the previous is more than 2 minutes. It should be noted that the threshold of 2 minutes, was extracted from the statistical analysis described in section II.

IV. Experimental Results

In this section we present the experiments performed in order to apply our methodology and evaluate the performance of our real-time crawler detection system.

Training Data Sets: For the purpose of training the system we have used logs obtained from the Web server of the Cyprus Hotline dedicated to the fight against illegal Internet content (`http:safeweb.org.cy`) and span a period of 4 months from Aug. 2007 to Dec. 2007. We created 1000 training sessions out of the Safeweb log, which were classified into crawler and human based in the Bayesian network model of Stassopoulou and Dikaiakos [12]. Out of the 1000 session, 300 were classified as crawler and 700 as human. For training the decision tree (using the ID3 algorithm, [8]) we used the 300 crawler sessions and then randomly selected 300 human sessions out of the 700, in order to maintain a balanced data set [12]. All sessions underwent a manual inspection by a human expert to ensure the correctness of the classification before training commences.

Testing the system: For the purpose of testing the system we used a different log file obtained form

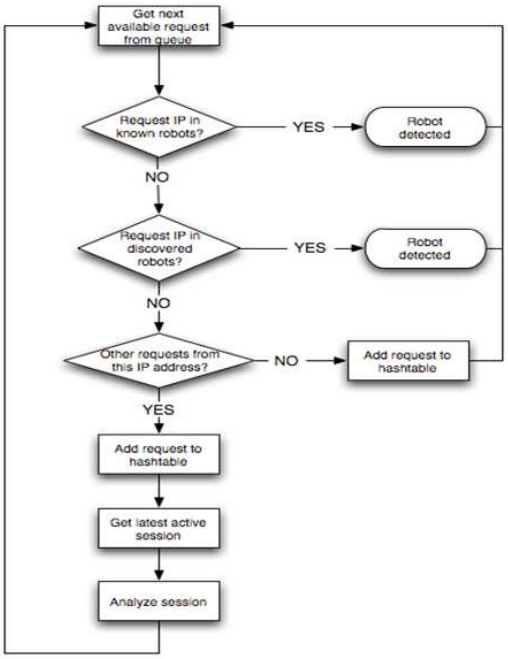

Fig. 4. The Request Processor flow chart

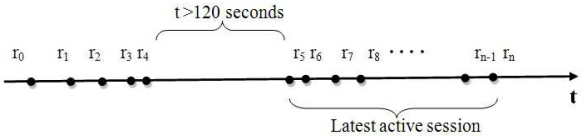

Fig. 5. Last active session

the Web server of a European research-project portal (`http://www.geclipse.eu`. The log file span a period of 4 months from Aug. 2007 to Dec. 2007. In order to send requests to the crawler detection system in real-time, we designed and implemented an emulator. The emulator reads each request from the log file and sends to the system as it would send them in real-time from client to server.

We performed 5 experiments, each time changing the minimum number of requests that must be collected from the same IP address, before it can be classified into crawler or human induced. (It should be noted that by the term $requests$ here, we mean http, php, htm, html, etc). In experiment 1, the classification of the IP address is performed after 5 requests from that address. We then increase the requests to 10, 15, 20 and 50 for experiments 2, 3, 4 and 5 respectively.

For evaluating the system, we performed the 5 experiments and calculated 3 metrics [14]:

i) $Recall$ (True Positive):Shows the ratio of correctly classified IP addresses as crawlers over the total number of crawler IP addresses:

$$Recall = \frac{\text{No. of Crawler IP addresses correctly classified}}{\text{Total no. of actual crawler IP addresses}} \quad (1)$$

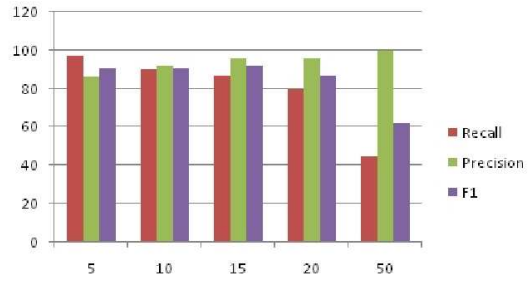

Fig. 6. Results of the experimental evaluation: the three metrics shown for each of the five experiments.

ii) *False Positive*: Shows the ration of IP addresses wrongly classified as crawlers over the total number of IP addresses detected as crawlers by our system.

$$FalsePos = \frac{\text{No. of IP addresses wrongly classified as crawlers}}{\text{Total no. IP addresses classified as crawlers}}$$

iii) *Precision*: Shows the number of IP addresses correctly classified as crawlers over the total number of IP addresses detected as crawlers.

$$Precision = \frac{\text{No. of crawler IP addresses correctly classified}}{\text{Total no. of IP addresses classified as crawlers}} \quad (2)$$

The two metrics recall(R) and precision (P) can be combined into one metric, the $F1$-measure [14]:

$$F_1 = \frac{2RP}{R+P}$$

The F_1-measure, summarizes both recall and precision by taking their harmonic mean. F_1 summarizes the two metrics into a single value, in a way that both metrics are given equal importance. The F_1-measure penalizes a classifier that gives high recall but sacrifices precision and vice versa. For example, a classifier that classifies all examples as positive has perfect recall but very poor precision. Recall and precision should therefore be close to each other, otherwise the F_1-measure yields a value closer to the smaller of the two. The results of the 5 experiments using the aforementioned metrics are shown in Table I.

As it can been seen from the results shown in Table I and Figure 6, our real-time crawler detection system yields promising results with both recall and precision being above 80% and 86% respectively in all experiments performed, with the exception of experiment 5. The reason for this difference in recall results in experiment 5 is due to the large number of minimum requests. In this case, the system did not classify any IP with less than 50 requests, which was our set minimum for this experiment. For example, if a crawler IP address had 49 or less requests in its active session then this session was simply not analyzed and its IP address not classified at all. However, this crawler IP was included in the total number of actual crawler IP addresses, i.e. the denominator of the recall definition 1, and hence the low recall value. If, on the other hand, we observe the precision for experiment 5, where the denominator of equation 2, does not include the "skipped" IP addresses due to minimum requests threshold, but only

Minimum number of requests	Recall	False Positive	Precision	$F_1 - measure$
5	0.97	0.12	0.86	0.91
10	0.90	0.07	0.92	0.91
15	0.87	0.04	0.96	0.912
20	0.80	0.036	0.96	0.87
50	0.45	0.00	1.00	0.62

TABLE I
EVALUATION METRICS OF EACH OF THE 5 EXPERIMENTS. THE EXPERIMENTS DIFFER IN THE MINIMUM NUMBER OF REQUESTS USED BY THE SYSTEM BEFORE MAKING A CLASSIFICATION DECISION FOR AN IP ADDRESS.

considers the total addresses classified as crawlers, then we have a perfect precision of 1. The best results, as indicated by the F_1 measure, are obtained when the minimum number of requests uses is 15, in which case our system correctly classifies 87% of the crawlers and wrongly classifies 5 IP addresses out of the 453.

V. CONCLUSIONS

In this paper we have presented a system for detecting web crawlers in real-time using machine learning techniques. To this end we have utilized an off-line, probabilistic web crawler detection system, in order to characterize crawlers and extract their most discriminating features based on statistical analysis. We tested our system in real-time with very promising results. The high accuracy with which our system detects crawler IP addresses while a session is ongoing, proves the effectiveness of our proposed methodology. These results provide a promising direction for future work.

REFERENCES

[1] Google and Yahoo! accused of click fraud collusion. ElectricNews.Net, April 5, 2005. http://www.theregister.co.uk.

[2] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan. Searching the Web. *ACM Transactions on Internet Technology*, 1(1):2–43, 2001.

[3] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual (Web) Search Engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[4] S. Chakrabarti, M. van den Berg, and B. Dom. Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery. In *Proceedings of the 8th World Wide Web Conference*, pages 545–562, Toronto, May 1999. Elsevier Science.

[5] K. Crawford. Google CFO: Fraud a big threat. Google exec calls click fraud the "biggest threat" to the Internet economy, urges quick action. CNN Money, December 2, 2004. http://www.cnn.com.

[6] M. D. Dikaiakos, A. Stassopoulou, and L. Papageorgiou. Characterizing Crawler Behavior from Web Server Access Logs. In A. M. T. K. Bauknecht and G. Quirchmayr, editors, *Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies (EC-Web 2003)*, volume 2738 of *Lecture Notes in Computer Science*, pages 369–378. Springer, September 2003.

[7] M. D. Dikaiakos, A. Stassopoulou, and L. Papageorgiou. An Investigation of WWW Crawler behavior: Characterization and Metrics. *Computer Communications*, 28(8):880–897, May 2005.

[8] J. R. Quinlan. Induction in decision trees. *Machine Learning*, 1(1):81–106, 1986.

[9] P. Huntington, D. Nicholas, and H.R. Jamali. Web robot detection in the scholarly information environment. *Journal Information Science*, 34(5):726-741, 2008.

[10] J. Lee, S. Cha, D. Lee, and H. Lee. Classification of web robots: An empirical study based on over one billion requests. *Computers and Security*, 28(8):795-802, 2009.

[11] A. Stassopoulou and M.D. Dikaiakos. A Probabilistic Reasoning Approach for Discovering Web Crawler Sessions. *Advances in Data and Web Management, Proceedings of the Joint 9th Asia-Pacific Web Conference, APWeb 2007, and 8th International Conference, on Web-Age Information Management, WAIM 2007. Lecture Notes in Computer Science*, vol. 4505:265-272, 2007.

[12] A. Stassopoulou and M.D. Dikaiakos. Web Robot Detection: A Probabilistic Reasoning Approach. *Computer Networks*, 53(3):265-278, 2009.

[13] P.-N. Tan and V. Kumar. Discovery of Web Robot Sessions Based on their Navigational Patterns. *Data Mining and Knowledge Discovery*, 6(1):9–35, January 2002.

[14] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[15] D. A. Vise. Clicking to Steal. When Advertisers Pay by the Lok, Fraud Artists See Their Chance. Washington Post, April 17, 2005. http://washingtonpost.com.

[16] G. M. Weiss. Mining with rarity: a unifying framework. *SIGKDD Explorations*, 6(1):7–19, 2004.

[17] S. Yang, I.G. Councill and C.L. Giles. The Ethicality of Web Crawlers. *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 668–675, 2010.