Exposing Click-Fraud Using a Burst Detection Algorithm

D. Antoniou, M. Paschou, E. Sakkopoulos, E. Sourla, G. Tzimas, A. Tsakalidis, E. Viennas
Computer Engineering & Informatics Department
University of Patras
Patras, Greece
antonid, paschou, sourla, tsak, viennas@ceid.upatras.gr
sakkopul, tzimas@cti.gr

Abstract— The explosive growth in the size and use of the World Wide Web continuously creates new great challenges and needs. One such need is dealing with click fraud, which aims at increasing clicks on certain ads and thus the profit of the websites which display them. In this work, we extend the concept of click fraud and redefine it as any pattern of clicks whose goal is to alternate the normal operation of a website in order to produce specific results. An indication of a click fraud may be a burst of clicks that can be simulated by an automated program or script. We deal with the problem of efficient real-time Click Fraud detection utilizing advanced data structures and exploiting their advantages concerning space and time required.

Keywords-burst web visits; advertising; click fraud; splay trees;

I. INTRODUCTION

The explosive growth in the size and use of the World Wide Web continuously creates new great needs and challenges. Focusing on challenges, a growing phenomenon that deeply affects a Website's reported traffic and functionality is Click-Frauds, which is highly connected to the burst of visits on a specific Webpage or Web Ad.

A. Bursts and Click-Fraud

A pattern of visits or accesses is 'bursty' when they occur with high intensity over a limited period of time. In particular, in bursty cases, a few Web Pages become very popular for short periods of time and are accessed very frequently in a limited temporal space. Such patterns have been also observed in various Internet applications in a number of studies [1], [16].

A short description of a burst scenario is described hereupon. We have a set of categories of WebPages and a number of random visits are executed to all WebPages by users. We define a set of WebPages to be preferred by the user when these WebPages are the most highly visited by him/her based on the visits recorded for a certain time interval. In particular, we count for each WebPage, how many accesses have taken place since the last time it was visited. If this number is sufficient to denote this WebPage as preferred and the time interval during which the accesses are performed is satisfying, then this access pattern is considered as a burst of visits.

Bursts occur in a Website's traffic as well and affect its functionality in many aspects as the following one. As more and more commercial enterprises go online, it is vital to make their Websites attractive to customers. One way to attract Website traffic is online advertising on search engines. In this case, besides the search results, an ad is placed in the search engine's WebPage. If the visitor clicks the ad, the advertiser has to pay a fee to the search engine. A problem that has arisen with pay-per-click is Click-Fraud [2]. "Click-Fraud" is the practice of deceptively clicking on search ads with the intention of either increasing third-party website revenues or exhausting an advertiser's budget [3]. The term "fraud" is used because in either case, the advertiser is paying for a click without receiving any true value [4]. Someone can use an automated script or program to simulate multiple clicks by a browser on an ad. Of course, the number of clicks has to be large enough in order to gain a considerable amount of money [2]. Therefore, a burst of clicks may be an indication of a Click-Fraud.

Click-Fraud is sometimes called "invalid clicks" or "unwanted clicks". This is partly because the word "fraud" has legal implications that may be difficult to prove or contrary to the interests of some of the parties involved. Google calls click fraud "invalid clicks" and says it is "clicks generated through prohibited methods. These prohibited methods include but are not limited to: repeated manual clicks, or the use of robots, automated clicking tools, or other deceptive software". Google acknowledges and describes the risks posed by click fraud in its annual reports [3].

B. Types of Click-Fraud

There are two main types of click fraud [3]:

- Inflationary click fraud: Search advertisements often appear on third-party websites and compensate those website owners on a per-click basis or with a share of advertising revenues. These third parties may click the ads to inflate their revenues.
- Competitive click fraud: Advertisers may click rivals' ads with the purpose of driving up their costs or exhausting their ad budgets. When an advertiser's budget is exhausted, it exits the ad auction. A common explanation for competitive click fraud is that firms have the goal of driving up rivals' advertising costs, but such an explanation may not be sub game perfect. If committing competitive click fraud is costly, then driving up competitors' costs comes at the expense of driving down one's own

978-1-4577-0681-3/11/$26.00 ©2011 IEEE

profits. A more convincing explanation may be found in the structure of the ad auction. When a higher-bidding advertiser exits the ad auction, its rival may claim a better ad position without paying a higher price per click.

In this work, the goal is to deal with more types of click fraud. A burst of visits may affect the functioning of a website in many ways. For example, a burst of visits to a poll may indicate a voting click fraud. In addition, bursts of clicks may intend to alternate and increase the popularity of posts in blogs, where the popular posts are promoted. There are myriad other types of click fraud, such as fraud designed to boost click-through rates, to invite retaliation by search engines against rival websites, or to do malicious harm based on philosophical or economic grounds.

The motive behind dealing with all kinds of click fraud is to protect website owners and increase the websites' security. The application of a technique against the click fraud phenomenon can prove to be crucial in making a Website more trustworthy and increase its credibility. In particular, the goal of the algorithm described in the next paragraphs is to detect click frauds real-time and ban the IPs responsible.

The necessity for real-time dealing with click frauds is more intense in cases of click frauds not related with advertizing. In advertising, periodical offline analyzing of the log-files can provide us with information that indicate click frauds and hence prevent the owners of the websites from being defrauded. This is not the case in other types of click fraud, such as click fraud in polls, where detecting click frauds afterwards in an offline manner is not effective, since the outcome of the poll is already affected. Internet polls are nowadays being used on a large scale and have become quite popular. However, there are security issues to be considered and solved when it comes to utilize web surveys [15]. We present an innovative solution which considerably increases the security of Internet polls by detecting bursts of votes. In general, a real-time detection of the click-fraud would protect websites from there kind of click behaviors that intend to affect certain outcomes of the website's operations.

C. Click-Fraud Detection

Search engines implicitly acknowledge they cannot fully detect Click-Fraud. Most search engines claim to offer advertisers some basic protections against Click-Fraud, though they do not explain specifically how they identify fraudulent clicks [3].

Detecting Click-Fraud is a relatively new area of research. On the simplest form, the broker performs threshold-based detection. If a web page is receiving a high number of clicks from the same IP address in a short interval, these clicks can be flagged as fraud. The detection gets complicated if the clickers are behind proxies or globally distributed. In such cases the broker has to use global IP blacklists and continually update its list as the botnets evolve, at the expense of losing income from authentic customer clicks [6].

In our work, a novel approach for Click-Fraud detection is proposed, exploiting bursts of visits. Our algorithm is based on analyzing the number of visits in a certain time interval. Depending on the type of click fraud under investigation, the algorithm may be set to count visits to WebPages or data units found on a Webpage, such as polls or posts. The proposed algorithm uses:

- Splay tree to store WebPages/Data units and detect the 'bursty' ones using visit frequency.
- Splay tree to store visiting IPs and detect suspicious ones.

The novelty of our approach is based on the fact that it is the first time, to the best authors' knowledge, that data structures are used for Click-Fraud detection, thus leading to less space and time needed. Moreover, data structures allow the Click-Fraud problem to be dealt real-time.

The rest of the paper is organized as follows. In section II details concerning Click Fraud and related work are presented. In the next section a burst detection algorithm is described, based on the self-organizing data structure called splay tree [13]. In section IV, a solution is presented in order to identify which bursts are suspects for Click Fraud. Finally, conclusions and future work are discussed.

II. Details on Click-Fraud and Related Work

Click fraud is an issue of particular concern for advertisers and service providers as it can potentially threaten the business model of the pay-per-click (PPC) advertising market. Therefore, a variety of proposals for reducing this particular phenomenon have emerged during the last few years.

Most PPC networks attempt to detect click fraud using various methods, such as stopping multiple clicks from the same IP address. To avoid detection, attackers have become more sophisticated, using a variety of techniques, including proxy servers, DNS hijacks, and multiple ISPs to generate fraudulent clicks from different IP addresses. [4]

Another approach to the problem of click fraud is an attempt to automatically recognize fraudulent clicks using machine learning algorithms to recognize them. These algorithms use information regarding the navigational behavior of users to try and distinguish between human clicks and those generated by robots. Some drawbacks of these techniques are that they require large datasets to train the learning methods, have high classification error, and are at the mercy of the efficiency of the scammers. [8]

Some common methods may help advertisers detect patterns that indicate possible click fraud. Most online advertisers begin to suspect click fraud when they notice one of the following cases:

- Increase in click rates with few conversions – One sure sign of click fraud is a drastic increase in the number of clicks without any increase in conversions.
- Large growth in advertising impressions on Content networks – Another sure sign of click fraud is a drastic increase in the number of impressions for ads within the content network. This happens when your ad is suddenly being triggered by keywords from a click farming site.

- Early depletion of daily budgets – As click fraud increased, advertisers will see their daily budgets become depleted quickly. This is especially true when advertisers enable the content network within their PPC campaigns. In these cases, clicks from both sources will hit the budget.

There are businesses devoted to detecting and auditing click-fraud by providing detailed evidence of possible click fraud patterns. This data can be used by advertisers to obtain a refund when possible. Moreover, businesses that advertise on the Internet may control click fraud by manually entering lists of known domains into their blocked site list, thus preventing their ads from appearing on certain sites. [4]

A real time click fraud detection and prevention system based on multi-model and multi-level data fusion is proposed in [7]. Each independent component can be considered as an invisible data mining module, in which "smart" software incorporates data mining into its functional components, often unbeknownst to the user. Evidence for click fraud from multiple models is "fused" in this system, so that it achieves improved accuracy for detecting fraudulent traffic. Conversely it increases the quality of clicks reaching advertisers' websites.

Anupam et al. [5] present a hit inflation attack on pay-per-click advertising schemes, which demonstrates the difficulty of detecting click frauds. A deceitful publisher puts a script on his website that is automatically downloaded onto a visitor's computer when said visitor goes to the publisher's website. The script then imitates a click onto the advertisement leaving the visitor none the wiser. The log files of the advertiser (and, if applicable, of the publisher) will show the visitor's IP address.

Bluff Ads, which are presented in [6], are a set of ads designed to be detected and clicked only by machines, or poorly trained click-fraud work force. These ads are targeted at the same audience profile as the other ad groups, but their displayed text is totally unrelated to the user profile. Hence they should not be clicked on by the benign user, acting as a litmus test for the legitimacy of the individual clicking on the ads.

According to [7] prevention mechanisms are based on blocking suspicious traffic by IP, referrer, city, country, ISP, etc. The system used in this case maintains an online database of these suspicious parameters. Real world data from an actual ad campaign are used to test the system. The results show that use of multi-level data fusion improves the quality of click fraud analysis.

Our solution is based on detecting bursts. As long as bursts as an algorithmic issue are concerned, several papers have been presented as well. A new algorithmic framework for elastic burst detection has been recently introduced: a family of data structures that generalizes the Shifted Binary Tree, and a heuristic search algorithm to find an efficient structure given the input [16].

III. DETECTING BURSTS OF VISITS

A. The Burst Detection Algorithm

We propose an algorithm for extracting the bursty web pages of a Website based on the number of visits of each WebPage in a certain time interval. The data structure that is used for the purposes of our algorithm is the Splay tree. In the following paragraphs we give a short description of the data structure for the non-specialist reader in order to make the algorithm as plain as possible. The senior reader may omit the next paragraph.

B. Splay Tree

The splay tree [13] is a self-adjusting form of binary search tree. The binary search tree is a data structure for representing tables and lists so that accessing, inserting, and deleting items is easy. On an n-node splay tree, all the standard search tree operations have an amortized time bound of O (log n) per operation, where by "amortized time" is meant the time per operation averaged over a worst-case sequence of operations. Thus splay trees are as efficient as balanced trees when total running time is the measure of interest. Good performance for a splay tree depends on the fact that it is self-balancing, and indeed self-optimizing, in that frequently accessed nodes will move nearer to the root where they can be accessed more quickly. This is an advantage for nearly all practical applications, and is particularly useful for implementing caches and garbage collection algorithms. In fact, for sufficiently long access sequences, splay trees are as efficient, to within a constant factor, as static optimum search trees. The efficiency of splay trees comes not from an explicit structural constraint, as with balanced trees, but from applying a simple restructuring heuristic, called splaying, whenever the tree is accessed. In our case, when a node is accessed more than a number of times, then it is splayed to the root of the tree.

The accessed node is brought to the root of the tree by performing splay operations which consist of rotations bottom up on the access path (Figure 4). These operations are:

- Zig Step: This step is performed when the accessed node A has a parent but no grandparent. In other words, its parent is the root. A single rotation of the edge connecting A and its parent is executed in this case.

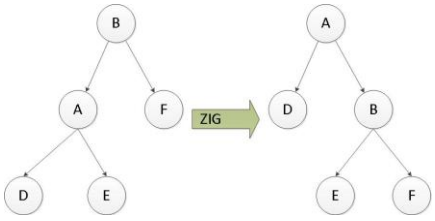

Figure 1. Zig Step

- Zig-zig Step: This step is executed when the accessed node and its parent are either both right children or are both left children. The edge connecting the parent of A and its grandparent is

rotated and then the one connecting A with its parent is rotated.

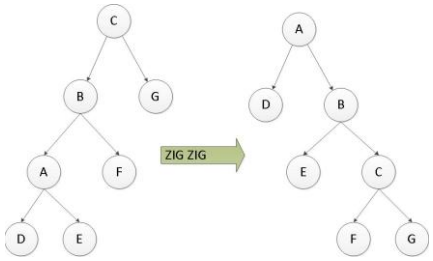

Figure 2. Zig-zig Step

- Zig-zag Step: This step is done when the accessed node is a right child and its parent is a left child or vice versa. The edge between A and its parent is rotate and then we rotate the edge between A and its new parent.

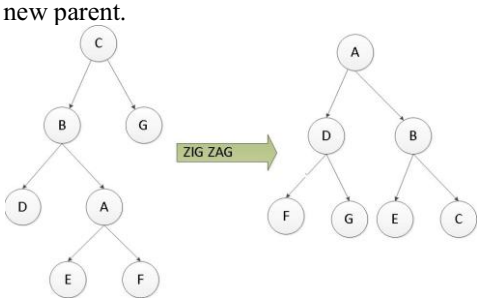

Figure 3. Zig-zag step

Splay trees perform many tree rotations after every access, which makes them less practically efficient in some applications. These rotations can be particularly deleterious when nodes are augmented with auxiliary structures. Nevertheless, we decided to use splay trees instead of biased skip lists so that we can take advantage of the conditional splaying variants' possibilities and enhance them into our algorithm. In general, splay trees have unique properties that make them more than appropriate for solving the problem addressed in this work, since they exploit time locality and space locality as well.

Data locality is much easier to maintain in problems having static data. But for dynamic data, such as occurring in splay trees, maintaining locality requires significant effort. In dynamic problems, such the one of proposing certain web pages to users, the design of the data structure is crucial in the problem of moving data efficiently. In splay trees the most recently accessed elements are brought closer to the root and the tree is being self-reorganized dynamically according to the elements accessed, so that the least frequently used elements will be those furthest from the root.

A good example of an application where there is the case of looking for the same element multiple times is a network router [14]. Routers must decide on which outgoing wire to route the incoming packets, based on the IP address in the packets. The router needs a big table (a map) that can be used to look up an IP address and find out which outgoing connection to use. If an IP address has been used once, it is likely to be used again, perhaps many times. Splay trees are designed to provide good performance in this situation.

As far as implementation details are concerned, aside from being easy to program and efficient in practice, splay trees also use less space because no balance information is stored. However, its disadvantages are its requiring adjustments at every tree search, resulting in potentially expensive individual operations. Of course, in the case of a highly visited web site, individual operations are not common and hence this field is ideal for applying splay trees.

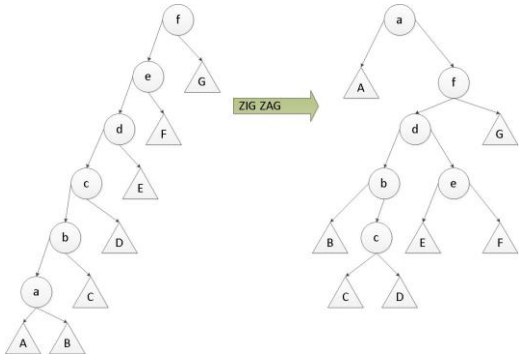

Figure 4. Accessing node 'a' and splaying it to the root of the tree

C. *The Burst Detection Algorithm*

In order to extract the bursty webpages we exploit the time and space locality of the splay trees. As we mentioned before, a sequence of accesses to a webpage is considered a burst depending on the number of accesses (A) and the time period (T) in which they have been performed. We modify the splay tree in order to splay a node to the root under two conditions:

- The node is accessed A times or more
- And these A accesses have taken place in a time period equal or less than T.

We use timestamps in order to ensure both conditions. A timestamp is a sequence of characters, denoting the time at which a certain event occurred. A timestamp is the time at which an event is recorded by a computer, not the time of the event itself. This data is usually presented in a consistent format, allowing for easy comparison of two different records and tracking progress over time. The practice of recording timestamps in a consistent manner along with the actual data is called timestamping.

Timestamps are typically used for logging events, in which case each event in a log is marked with a timestamp. In filesystems, timestamp may mean the stored date/time of creation or modification of a file.

In our case, we maintain for each node a queue of size A. In this queue we store timestamps of the accesses performed to the node according to the FIFO logic. The queue is of size A in order to record the time period in which A accesses are performed to a node and therefore the first condition of a burst event is satisfied (Figure 5). Every time a node is accessed, its current timestamp is pushed into the queue and the oldest timestamp is popped out of the queue. In order to check if the second condition is satisfied, we subtract the oldest timestamp from the current timestamp. If the result is

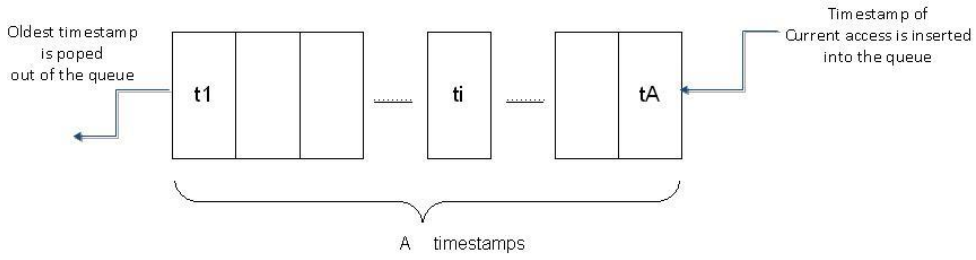

Figure 5. Timestamps are used to ensure that the number of hits and the time interval in which they take place suggest a burst of visits

equal or smaller than T, then a burst has occurred and the node is splayed to the root of the tree (Figure 6).

IV. WHICH BURSTS ARE CLICK FRAUDS

In order to identify the IPs which are suspect for Click Fraud, we utilize splay trees as well. For every visit that is recorded in the Splay tree in which WebPages or data units are stored, the IP which is responsible for the visit is inserted into a global splay tree. The same technique as described in the previous section for detecting bursts is followed. More specifically, every IP that visits the Website is stored in the splay tree. Therefore, each node in the splay tree represents an IP and consecutive visits in a short time interval from the same IP would result to splaying the IP to the root of the tree. Hence we would have access to the bursty IP in O (1) time. Conclusively, whenever a burst to a unit occurs, if simultaneously a bursty IP is detected, then this IP is responsible for the burst of clicks/visits and a click fraud has been exposed. Of course, the bursts in the two splay trees do not have to happen absolutely simultaneously. Depending on the website's popularity, the suspect IP may be the most recent bursty IP or the one that may result after certain time periods. In all cases this IP can be banned in order to prevent future click frauds and secure the Website's services.
The disadvantage of this solution is that it is not able to detect visits that come from the same network address but not the same IP.

V. THE SPACE AND TIME ANALYSIS

A. Space requirements

As long as the space complexity is concerned, the space, as expected, is mainly consumed by the splay trees and the timestamps' queues.

- Splay tree: We need 1 splay tree for the website. In the splay tree we store all the WebPages of the site. Therefore, the space required is N, where N is the number of web pages of the website.
- Timestamps' queues: Each queue has in worst case size A, where A is the number of hits that suggest a burst. We need 1 queue per node of the splay tree.

So, in total, the space required is N*A. Since we need one more splay tree for storing the IPs, the space required in this case is proportional to the number of IPs that the visits are coming from.

B. Time requirements

As long as time complexity is concerned, per access we need:

- log (Ñ/ñ), in order to splay a node. That is at most log(#nodes)
- We need O (1) time to update the queue of the node accessed and perform the necessary subtraction. We also need O (1) time to return the root from the splay tree in case of a bursty pattern. Hence, log (#nodes)*O (1) time is required to return a bursty node.

VI. DESCRIPTION OF THE APPLICATION OF THE ALGORITHM ON A CLICK FRAUD SCENARIO

In order to present a validation and verification scheme for the proposed concepts we present in this section an approach that functions as a proof of concept for our prototype.

A simple bot is used to vote automatically by repeatedly selecting the same voting option of a poll on a website. Depending on the Website's popularity, the size of a burst in terms of time period and number of clicks has been already defined. Let's suppose that if there are more than 100 clicks in a time period shorter than 10 seconds, then a burst has occurred. In addition, the goal of the bot is to increase the votes of a certain option by more than a thousand. Therefore, if in the first 10 seconds, more than 100 clicks have taken place, and if according to the global splay tree, most of the clicks are coming from the same IP, then this IP is banned, the click fraud is detected and the negative effects of its action have been minimized.

VII. CONCLUSIONS AND FUTURE WORK

In this paper we presented an algorithm for real-time detection of click frauds. Click fraud is defined as a burst of visits to a webpage or a data unit coming from the same IP. The proposed solution focuses on efficient real-time Click Fraud detection utilizing advanced data structures and exploiting their advantages concerning space and time required. In the future, the issue of effective click-fraud detection and auditing needs to be addressed more

aggressively. Detecting fraudulent websites could be relatively simple once detailed log information was obtained.

REFERENCES

[1] B. Zhou, S. C. Hui, K. Chang, "An intelligent recommender system using sequential web access patterns", Proc: IEEE Conference on Cybernetics and Intelligent Systems, IEEE, Singapore, pp.393–398, December 2004.

[2] E. Sakkopoulos, et al., "A web personalizing technique using adaptive data structures: The case of bursts in webvisits", J. Systems and Software, vol. 83, issue 11, pp. 2200-2210, November 2010, doi: 10.1016/j.jss.2010.06.026.

[3] K. C. Wilbur, Y. Zhu, "Click Fraud", J. Marketing Science, vol. 28, issue 2, March 2009, doi:10.1287/mksc.1080.0397

[4] Anonymous, "Identifying and Stopping Click Fraud", March 2007.

[5] V. Anupam, A. Mayer, K. Nissim, B. Pinkas, and M. K. Reiter, "On the security of pay-per-click and other web advertising schemes" Comput. Netw., 31(11-16):1091–1100, 1999.

[6] H. Haddadi, "Fighting Online Click-Fraud Using Bluff Ads", ACM SIGCOMM, vol. 40, n. 2, pp. 22-25, April 2010.

[7] C. Walgampaya, M. Kantardzic, R. Yampolskiy, "Real Time Click Fraud Prevention using multi-level Data Fusion", WCECS 2010, October 20-22, 2010, San Francisco, USA

[8] A. Tuzhilin, "The Lane's Gifts v. Google Report", July, 2006

[9] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[10] D. Kornack and P. Rakic, "Cell Proliferation without Neurogenesis in Adult Primate Neocortex," Science, vol. 294, Dec. 2001, pp. 2127-2130, doi:10.1126/science.1065467.

[11] H. Goto, Y. Hasegawa, and M. Tanaka, "Efficient Scheduling Focusing on the Duality of MPL Representatives," Proc. IEEE Symp. Computational Intelligence in Scheduling (SCIS 07), IEEE Press, Dec. 2007, pp. 57-64, doi:10.1109/SCIS.2007.357670.

[12] X. Zhang, D. Shasha, Better burst detection, Proceedings of the 22nd International Conference on Data Engineering April 03–07 (2006), p. 146.

[13] Sleator, D.D., Tarjan, R.E., 1985. Self adjusting binary search trees.Journal of the ACM 32, 652–686.

[14] Srinivasan, T., Nivedita, M., Mahadevan, V., 2005. Efficient packet classification using splay tree models.IJCSNS6 (May(5B)).

[15] Alessandro Basso, Michele Miraglia: Avoiding Massive Automated Voting in Internet Polls. Electr. Notes Theor. Comput. Sci. 197(2): 149-157 (2008)

[16] Evangelos Sakkopoulos: Semantic technologies for mobile Web and personalized ranking of mobile Web search results. Metadata and Semantics, Post-proceedings of the 2nd International Conference on Metadata and Semantics Research, MTSR 2007, Corfu Island in Greece, 1-2 October 2007.Spring, pp. 299-308

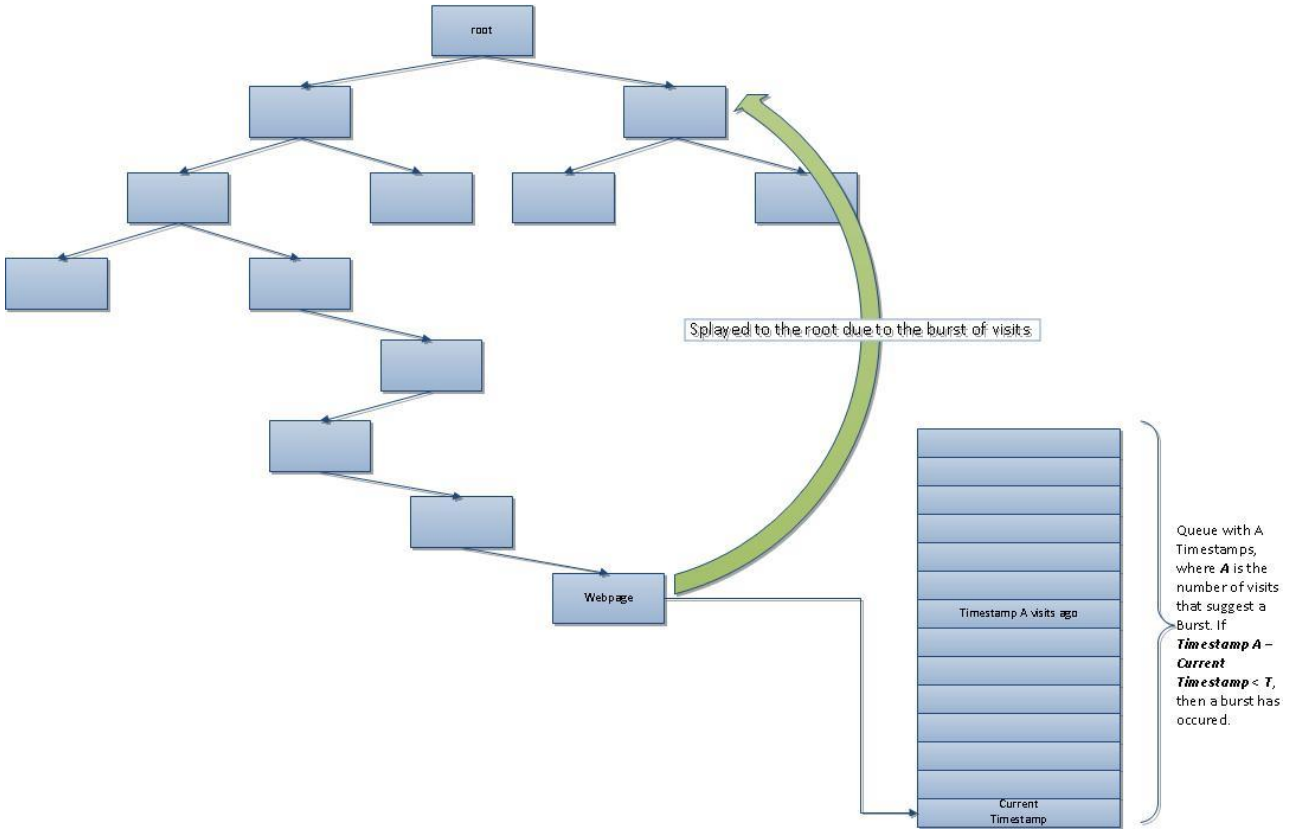

Figure 6. Splaying the most recent bursty web page to the root of the splay tree