Indian Journal of Science and Technology, *Vol 8(15), DOI: 10.17485/ijst/2015/v8i15/54635, July 2015*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

A Novel Approach to Enhance DNS Cache Performance in Web Browser using SPV Algorithm

K. Sampath Kumar[1*] and G. K. D. Prasanna Venkatesan[2]

[1]Department of Computer Science and Engineering, PGP College of Engineering and Technology, Namakkal - 637 207, Tamil Nadu, India; ksampathkumara@yahoo.com
[2]Department of Electronics and Communication Engineering, SNS College of Engineering, Coimbatore – 641035, Tamil Nadu, India

Abstract

The Domain Name Server (DNS) and web browser are playing major role in the global connection to share the information. Internet browsers are available more in number and most of the browsers generate the results based on the searched content within their own browser methodology. But for the user it can be a difficult task to identify the best results in this output list. In this scenario the proposed method, Shortest Parsing Verb algorithm which is used to identify the best top three browsers search content that can be inserted in this browsers individually and generate the result to store in separate location. All these three results are correlated with the help of association rules, the results present in all three browsers grouped in first category, the results present in any two browsers grouped in second category and the remaining will be placed in third category. The results domain name and Internet Protocol address also stored to increase the DNS cache performance so that frequently used domains can be fetched from the DNS cache memory. The results of a particular search exhibits easily based on the users need.

Keywords: Domain Name System, Shortest Parsing Verb, Time-to-Live Mechanism, Uniform Resource Locator, Web Browser

1. Introduction

Domain Name System (DNS) server provides the right domain name for IP address mapping, so that the user can access the web for sending mails, online transactions etc. In Internet, IP addresses are used to route packets from the user to server. It is easy to remember few IP addresses but difficult to remember thousands of these addresses and finding which server is associated with each address is also a threatening task. The purpose of domain names entered in web browser is to communicate the user need to the server. For example in google.com, whenever enter the Google domain name into the address bar of a browser the particular page appears and the system carry out a process to resolve google.com to an IP address[1]. The Domain name system is a basic building block of the Internet. In day-to-day life, the performance of various applications depends on the responsiveness of DNS and accuracy[2]. The Domain Name System was originally planned to provide a naming service by one-to-one mapping between a domain name and an IP address. But, the popular applications enhanced from static content hosting to distribute and dynamic hosting[3].

The local DNS resolvers against Google DNS and OpenDNS are applicable for a huge set of vantage points. The host measurement inside the different commercial

** Author for correspondence*

ISPs reveals that they have two aspects to explore the impact of responsiveness: latency and content of the DNS cache whenever the query is issued[4]. A complete analysis of traces of Domain Name System (DNS) connected with TCP traffic will be collected on the Internet links. The first element of the analysis is about clients to act with the wide area DNS, focusing on performance and the occurrence of failures and errors. The second element evaluates the efficiency and the performance of DNS caching. The results of trace-driven simulations explore uneven TTLs and degrees of cache sharing on DNS cache hit rates. Due to the heavy-tailed nature of name access, reducing the TTLs of address records as low as few hundred milliseconds has little adverse effect on hit rates and additional gain is obtained by sharing DNS cache between more than 10 to 20 clients[5,6.] The wide area DNS performance at client's location is based on the user data obtained by grouping of different network environments such as connection technology, Internet topology and client ISP.

The explosion of WWW traffic has triggered huge interest in distributed web server systems. Among the different distributed web server architectures, the DNS-based distributed system is a promising solution in terms of scalability, availability and performance. DNS name caching has significant effects on the load balancing of distributed web server systems due to the traffic skewness. The page caching schemes also can affect the system load balancing a lot[7.] Effective caching in the Domain Name System is significant to its performance and scalability. Existing DNS supports weak cache consistency using the Time-To-Live (TTL) mechanism, which functions reasonably well in normal situations[2.] The maintenance of strong cache consistency in DNS as an essential handling mechanism has become demanding one for three objectives which are: 1) To quick respond and handle exceptions in Internet failures caused by natural and human disasters, 2) To grasp frequent changes of Internet Protocol addresses and 3) To provide fine grain controls for content delivery services to balance server load distributions[2,8]. The web search normally used for searching contents like lexical, structural, instance, inference similarity and ontology method also the best way for representing the retrieval of useful information[9].

2. Proposed Work

2.1 Web Browser

A web browser is a software application for presenting, traversing and retrieving information resources on the World Wide Web. An information resource is identified by a Uniform Resource Locator or Identifier (URL or URI) and may be a web image, video, page or other part of content hyperlinks present in the resources permit users to find the way to their related resources[10.] This process begins when the user enters an input as a Uniform Resource Locator (URL), for e.g., www.google.com into the browser. The prefix of the URL or URI starts with Hyper Text Transfer Protocol (HTTP) and identifies a resource with help of HTTP. Number of browsers supports a mixture of prefixes, such as HTTPS, File Transfer Protocol (FTP) and local files to access internet. Web browsers cannot handle the prefixes directly and often handed over to another application completely[11.] Even though browsers are primarily planned to use the World Wide Web, they can be used to access information provided by web servers in private networks or file systems. As a client/server model, the browser is the client run on a computer that links to the web server and request information[12.]

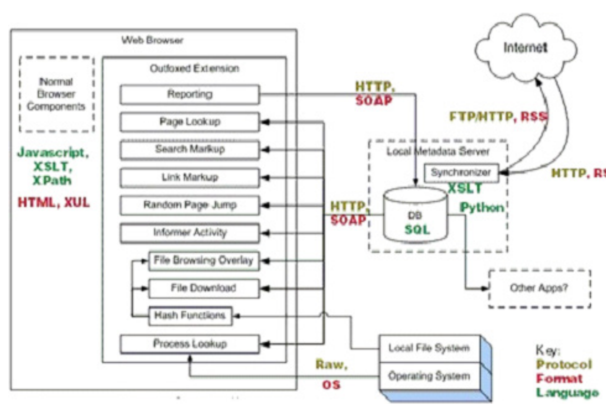

Figure 1. Basic Architecture of Web Browser.

The web server sends the requested information back to the web browser which displays the results on the computer or other Internet allowed device that supports a browser. Internet Protocol version 6 (IPv6) is the latest version of the Internet Protocol (IP) which provides an identification and location system for computers on networks and routes traffic globally[13]. IPv6 was developed by the Internet Engineering Task Force (IETF) to deal with the long anticipated problem of IPv4 address exhaustion[14,15]. Figure 1 represents the basic architecture of web browser get in touch with Internet through HTTP, FTP, SOAP, RSS etc., Figure 2 shows the basics architecture of web server along with Internet and database.

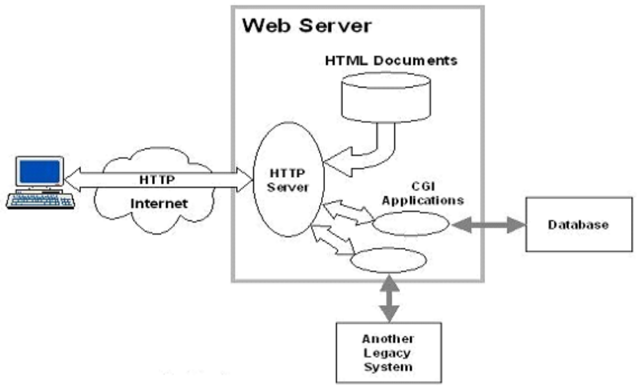

Figure 2. Basics Architecture of Web Server.

2.2 Methodology

Each browser has a function separately for searching the content in the web browser, and individual results are displayed in the screen. Normally various browsers are available, so that the user get confused to select the relevant results for particular content. In this case, the results are based on the priority which one is rated as first rank, but it is difficult for the user to find the content in dropdown list, because the contents are different at all the times. We proposed a novel approach by taking the three different web engines or web browsers like www.google.com, www.in.search.yahoo.com and www.bing.com. While searching for the content in these three browsers, sometimes the content will be similar in all three web browsers. In that case the similar contents have to be rated first, different contents are rated second rank, third rank and so on. In this process to improve the DNS performance, Shortest Parsing Verb (SPV) algorithm has been used. While implementing the idea of Shortest Parsing Verb in each search engine, the data from the particular request are fetched from the server database. The fetching methodology followed by each browser is different from each other. Hence we implement Shortest Parsing Verb (SPV) on top 3 search engines for our case study and commonly used keywords entered in each search engine to study the result. According to the results, the most repeated content has high rank. Figure 3 shows the basic structure of SPV. Figure 4 and Figure 5 represents the architecture of SPV to put together with browser, association rule tool and rank Calculator.

The SPV searching method implemented by choosing any of the following:

- Giving the key word directly to the search engine.
- Giving the key word with some wild card character (for ex. '... ? etc).
- Giving the key word with define in the search engine.

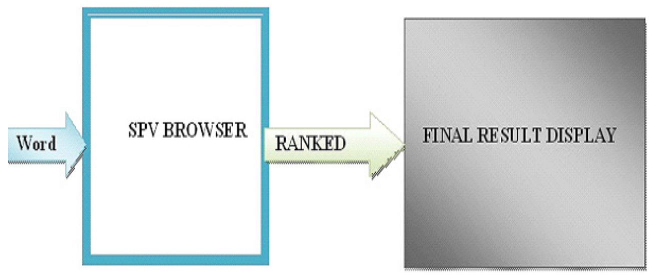

Figure 3. Block Diagram of SPV Browser.

2.3 Shortest Parsing Verb Modules

In the proposed system, we have three module functions. The modules are: Mining Module, Filtering Module and Reporting module.

2.3.1 Mining Module

In mining module gets attribute from the user and search from the top three browsers with three different methods. In this module, once the user gives the search attribute on the search space, immediately the word get searched in three different browsers and displays the top search results. Based on the SPV criteria, it takes three roles: wild card character, define keyword and direct key word. These results are used in the filter module and stored on the cache memory. All the stored results are entered in the filtering module.

2.3.2 Filtering Module

In this filtering module displays the list of the top searches that are made on the mining module. Now, each filter results are used to search with the search results of the other two browser search results. When the first and next browser results are matched, the data has been removed and stored in another location on the server. The remaining data is being mapped to the next browser by this methodology. After this analysis, the same attribute is being checked with keyword define and with wild card character. At the final stage, results are made available for the final filtering.

2.3.3 Reporting Module

In this reporting module final search results of the search attributes with the rank value representation. The rank value is given by comparing three browsers redundant values with three, two and one rank values. In the reporting module, the top most search result wise report can be generated and the graph can be verified with the variation of the result is shown in that chart.

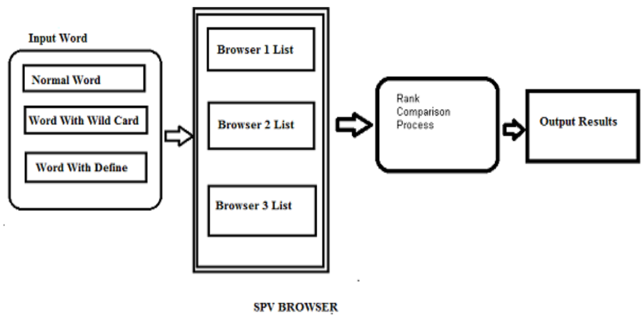

Figure 4. Basic Architecture of SPV Browser.

2.4 SPV Algorithm

The Shortest Parsing Verb (SPV) algorithm is as follows:

Step 1: Enter the word in the SPV browser and the same word is placed on the top three browsers.

Step 2: The results are displayed on different pages.

Step 3: The associative algorithm uses the top results from all the browsers.

Step 4: If the result is found on the multiple browsers, then that result is given ranking and saved in the separate page and the remaining on another page.

Step 5: This method is repeated using wild card character and define Keywords, and this results are replaced over the existing results.

Step 6: go to step 4

Step 7: Three different categories of results are shown and the best output is listed out.

Step 8: The result is displayed along with the graph and the exact result.

Step 9: Selects the best results from SPV browser.

This novel approach of SPV algorithm uses the following rules:

- Insert(X), Where X will be the input word.
- List of (A) = Search of (X, B1), List of (B) = Search of (X, B2), List of (C) = Search of (X, B3), Where A, B, C are list of X (word content) related search result and output received from Browser B1, Browser B2, Browser B3.
- If (List of (A) = List of (B) = List of(C)) add related result to RANK (TOP1) and store.
- If (List of (A) = List of (B)) or (List of (A) = List of (C)) or (List of (B) = List of (C)) add related result to Rank (TOP2) and store.
- If (List of (A) ≠ List of (B) ≠ List of (C)) then add the related result to Rank (TOP3) and store.

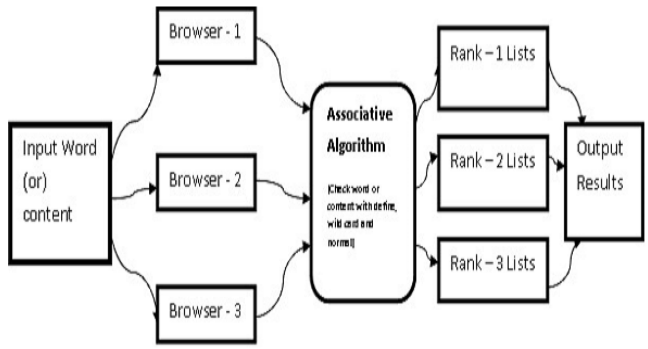

Figure 5. Basic Architecture of SPV Algorithm.

3. Results and Discussion

In this proposed work, we find a huge difference in the three search engine result contents and also find variation while giving the words with definition and without definition in the each of the search engine. Figures 6 to 11 show the variation of the top three web browser search results. Figure 12 and 13 represents the variation of the search result of three different browsers with specification, without specification and with wild card character for each searching text.

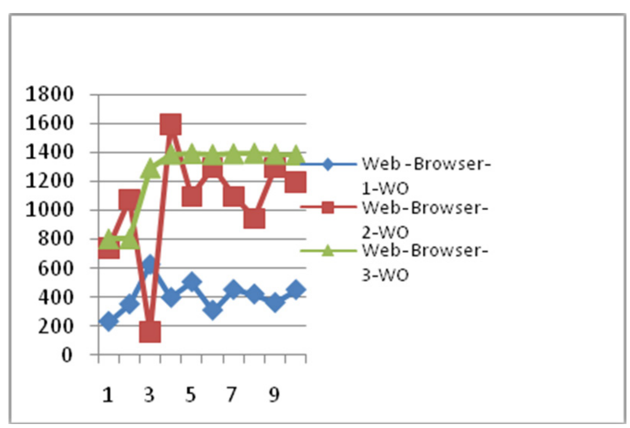

Figure 6. Without using define and wild card.

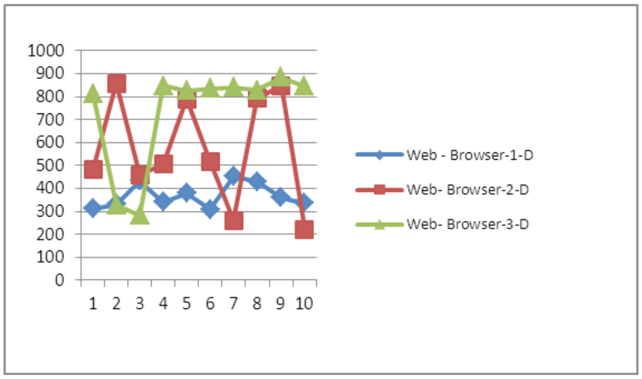

Figure 7. Using define key word.

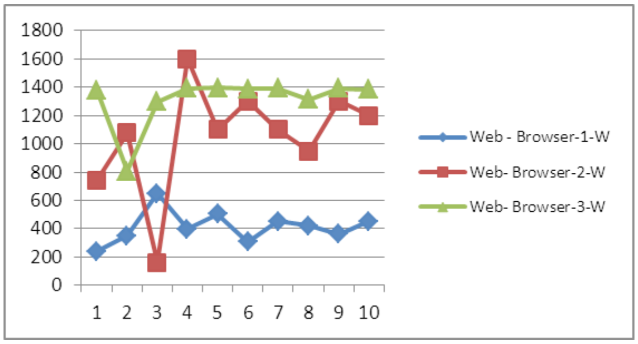

Figure 8. Using wild card.

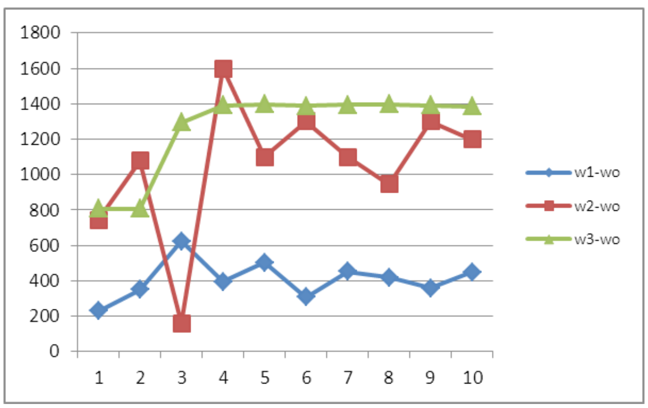

Figure 9. Without define key word and Wild card result for three browser graph.

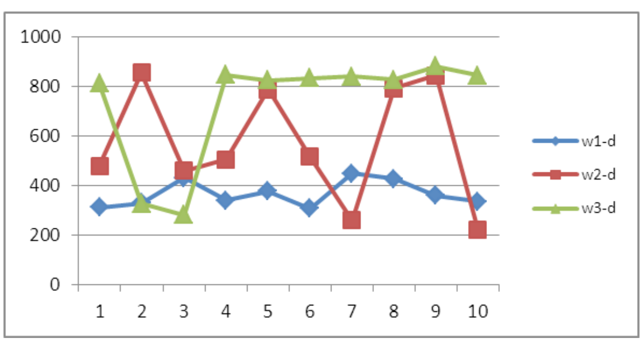

Figure 10. Using define key word for three browser comparison graph.

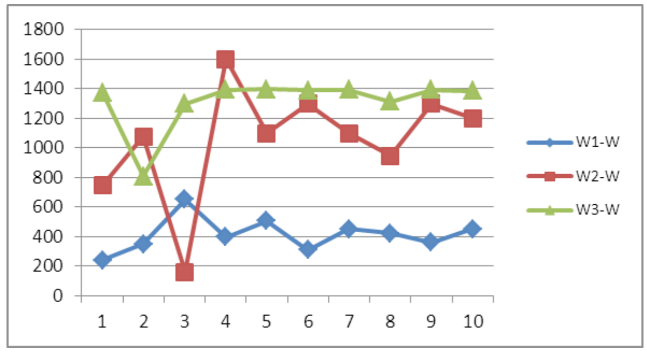

Figure 11. Using wild card character for three browser comparison graph.

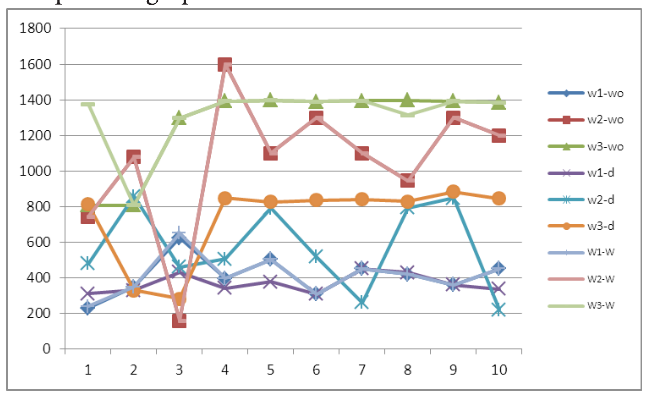

Figure 12. Overall results.

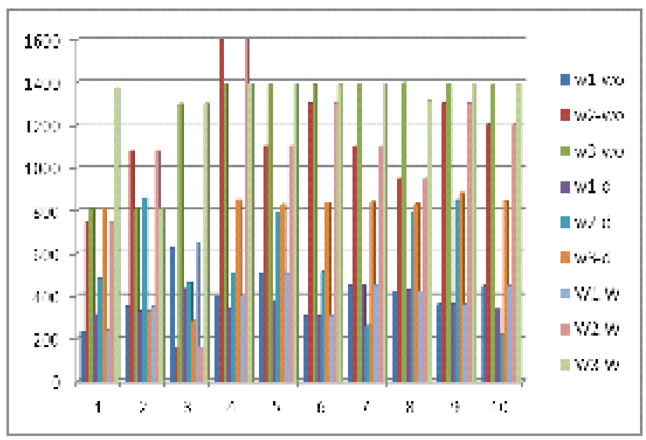

Figure 13. All Three browser comparison graph.

4. Conclusion

The proposed work distinguishes and assesses the capability of the different search engine. The SPV algorithm increased the optimum search results based on the given input. An individual browser generates different output based on the rank calculation with respect to key word and frequently accessed web sites are displayed. The SPV algorithm connected the three different browsers to compare the given key word content, and these three browser results are summarized based on the comparison with association method. Finally, the best or top most results are stored and displayed. This method reduces the DNS cache complication of caching the IP address and increases the caching performance of frequently used IP address and stored in order. So that the user is able to classify the top most web sites IP address easily and thus it minimizes the search time.

5. References

1. Kumar KS, Venkatesan GKDP. Effective method of prevention of cache poisoning for wild card secure DNS-A novel approach. Engineering Science and Technology: An International Journal. 2013; 3(2):256–9.
2. Chen X, Wang H. Maintaining strong cache consistency for the domain name system. IEEE Transactions on Knowledge and Data Engineering. 2008; 19(8):1057–71.
3. Ager B, Muhlbauer W. Comparing DNS resolvers in the wild. IEEE Proceedings of Internet Measurement Conference (ACM); 2010. p. 15–21.
4. Poese I, Frank B. Improving content delivery with PaDIS. IEEE Proceedings of International Conference on Internet Computing; 2011. p. 1–7.
5. Jung J, Sit E, Balakrishnan H. DNS Performance and the effectiveness of Caching. IEEE/ACM Transactions on Networking. 2002; 10:589–603.
6. Available from: http://academicresearchmicrosoft.com/paper/113513.aspx
7. Xu Z, Huang R. Load balancing of DNS-based distributed web server systems with page caching. IEEE Proceedings of Parallel and Distributed Systems; 2004. p. 587–94.
8. Ren S, Tan E. TopBT: A topology-aware and infrastructure-independent bit torrent client. IEEE Proceedings of INFOCOM; 2010 Mar. p. 1–9.
9. Vigneshwari S, Aramudhan M. Social information retrieval based on semantic annotation and hashing upon the multiple ontologies. Indian Journal of Science and Technology. 2015 Jan; 8(2):103–6.
10. Available from: http:/en.wikipedia.org/wiki/web browser
11. Yu Y, Wessels D, Larson M. Authority server selection of DNS caching resolvers. ACM-SIGCOMM Computer Communication Review. 2012 Apr; 42(2):80–6.
12. Wang L, Guo Y-P. Web Pages Search engine based on DNS. Journal of Network and Computer Applications. 2007 Apr; 30:466–78.
13. 13.Krishnan S, Monrose F. DNS Prefetching and its Privacy implications: When good things go bad. Proceedings of the 3rd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more; 2010. p. 10–20.
14. Deccio C, Kant K. Measuring availability in the domain name system. 2010. p. 5. Available from: spirit.cs.ucdavis.edu/pubs/March
15. Available from: http://en.wikipedia.org/wiki/IPv6