Prefetching of Web Objects For Effective Retrieval Process Through Data Mining Techniques

T S Bhagavath Singh (rnsit.tpo@gmail.com)
Anna University https://orcid.org/0000-0001-6830-5490
S Chitra
Er. PM College of Engineering

Research Article

Keywords: caching, clustering, prefetching, proxy, prediction

Posted Date: April 30th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-266666/v1

Abstract

With the exponential increase of the internet's user base, performance enhancing network architectures and algorithms has manifested themselves as a requisite. Algorithms for Prefetching and Caching of Web Objects have been observed to effectively minimize user perceived latency. These algorithms are made use of in architectures limited to a particular user. We can further improve the performance of these algorithms by making use of techniques like data mining. We propose an innovative idea of implementing Prefetching and Caching algorithms in a Clustered Network. This will enable all users in a particular cluster to make use of pre-fetched and cached web objects from all other users. The result of simulations indicates a reduction in web latency, internet traffic, and bandwidth consumed.

Introduction

With the web offering a surfeit of services and contents, the increase in its complexity is unavoidable. One of which is the escalation of user perceived web latency. This is made worse due to the upsurge in internet traffic. In order to curtail this latency, innovations like caching and prefetching were introduced.

A. Caching

In its simplest form, cache is a temporary storage for data that is used frequently. Web caching can be implemented either in the client side or in the server side. In the client side, Proxy Server Applications are made to act as an intermediate, between client and server, where web objects are cached. Content Delivery Systems (CDNs) are used in place of Proxy Server Applications when caching is implemented in the server side. In both cases, the intermediate source is requested for the required data first, and only when it fails does the retrieval take place from the original server. When a new object is received, a copy of it is cached for future retrieval.

When implemented with respect to the World Wide Web, Caching curtails network traffic, bandwidth consumption and network congestion. Moreover, web caching ameliorates reliability, as a cached copy will be available at all times, regardless of the server's functional capabilities.

B. Prefetching

The implementation of web caching resulted in reduced latencies, yet at least for the first access the network had to be traversed. This can be avoided with the help of Web Prefetching. It is the technique of fetching, followed by caching, of web objects before user requests for them. This involves deducing the user requests which is based on referring the spatial locality of Web Objects [17]. It utilizes the client's unused time, which is the time gap between the requests, in order fetch Web Objects. Prefetching algorithms are based on clustering of users, probability of access and popularity of web objects. With the help of prefetching, latency is reduced, and bandwidth under-consumption is avoided.

C. Clustering

Clustering is the process of organization of the data into clusters, to maintain closeness among the objects of the same cluster but difference with the objects of the other clusters. It can also be considered as a form of data compression, as a cluster of data objects can be treated as one group.

The analysis of clusters provides information about the nature of the data in each cluster and the placement of data. Efficient bandwidth usage decreased latency and lower network congestion is made possible when cluster analysis, along with web caching and web prefetching, is applied to the users of the web.

In this paper, we propose a scheme which utilizes the techniques of Web Caching and Web Prefetching, along with the help of Clustering, to enable users in the same cluster to access cached Web Objects.

Literature Survey

In [17], the author presents a prediction design based on analytical interrelation among web objects and proposes away to integrate caching and prefetching algorithms by including the prefetching technique within a caching algorithm. Author proposes to perform this combination in the intermediate server applications. Author proposes a unified algorithm that performs both prefetching and caching, called Pre-GDSF. Which looks at prefetching aggressiveness, increased network traffic and replacement policy. From simulations, it is shown that the algorithm increases the web caching performance with increase in traffic. Also, the author advises about the choice of selecting a fitting prefetching range as it directly affects the conduct of the system. Finally, the author concludes that small-scale caches perform better with prefetching.

In [2], the author presents an innovative prefetching technique based on semantics. Prediction of subsequent requests is based on the priorities set because of previously obtained records, instead of temporal rapport among Uniform Resource Locators. Because the client navigates based on keywords in Uniform Resource Locators. Author refers to the event as “semantic locality”. Search engines accept keywords and provides a series relevant documents, on the contrary this type of prefetching systems obtains priorities based on the user’s previous usage patterns and fetches new URLs while the user is browsing the current site. Additionally, new URLs that were never visited can be prefetched because of semantics. Neural network is used to build a collection of URLs and to prefetch user’s requests. It also includes a good adaptability to the varying nature of the user’s browsing pattern. The technique was tested by simulating it in the daily browsing of various news sites.

In [3], the author noted the outcomes of condensation of the cache performance by measuring a squid cache for various compression ratios, where the cache condensed the data, and the decompression takes place in the client-side. The performance was determined by benchmarking it in the Best- and Worst-case scenarios. Also, the author demonstrated a point of crossover where expense of cache compensated the performance gains. Advantages of caching condensed content are greater compression ratios and performance of the cache.

Compressed content requires less space, therefore caching them will improve the hit ratio as more objects can be stored in the system's main memory. Disk Input / Output is also sped up, which will increase the throughput and decrease the delay. Content compression is useful as an improvement means as its features can be combined with transmission latency reduction.

In [4], the author proposes a prediction model based on clustering of web user sessions. Author makes use of various data mining techniques on the logs in the server to predict the next page. The browser can predict and prefetch the next page which will decrease the user perceived latency. In this way the analysis of predicting the consequent pages for user is performed. The author claims that this model provides better accuracy than the present prediction models. It is essential to maintain the order of the accesses made by the user to predict the next page queried by the user. Additionally, sequence alignment was used to measure the integrated distance among 2 terms.

In [5], the author explains the group of prefetching algorithms which are aimed at improving the efficiency of the system. The terms used for comparison are obtained by combining the hit rate and the bandwidth. The author also proposed a set of algorithms that included the ratio. The proposed algorithms are called "Objective-Greedy prefetching". The algorithm fetches objects in a greedy fashion, that is, more than those that are finally required. The result provided by the paper are obtained using the comparison between the "Objective-Greedy Algorithms" and the current algorithms based on the various metrics specified above.

The highest hit rate is obtained by the "Hit Rate-Greedy" and the lowest bandwidth is obtained by the "Bandwidth-Greedy" techniques. The simulations performed resulted in the conclusion that the proposed techniques provided better results than the previously proposed techniques.

In [6], the author analyzes the effect of the architecture of web prefetching in the reduction of the latency perceived by the users. The author also examines prefetching architectures with respect to the aspects that restrict the predictive power and evaluate these hypothetical limits in terms of number of requests and the latency identified by the user. The main goal of the study is to analyze the architecture to decrease the latency and get analysis on the cap of performance.

The location was analyzed based on the number of requests and the latency observed by the user. Predictors based on clients/servers were found to be favorable up to a certain extent. Analysis was performed on "Collaborative predictors" too.

In [7], the author proposes a prefetching technique wherein the operation is performed on the client side and the user chooses the objects to fetch. The simulations provided the result that this technique can reduce latency "by up to 81% in a homogeneous environment" and "63% in a heterogeneous environment". This technique does not increment traffic, but on the contrary, it reduces latency. It can be used on the side of the clients with any help from the other parts of the network infrastructure.

In [8], the author proposes an analysis of performance of web servers based on “distribution-aware caching”. Static Web document caching is performed using “request distribution”. The author uses “cooperative caching” and “exclusive caching” to avoid producing object copies. Author explores the interests of “cooperative caching algorithms” to drive them towards “general purpose cooperative caching algorithms”. The kernel of Linux was exercised for the caching technique’s implementations.

In [9], the author documents a technique which integrates caching and prefetching. The prefetched URLs are dependent on the “documents”, “cache contents” and the space in the network. To manage user inclinations, “Preference lists” are used. Cryptographic methods are used for keeping track of the account of URLs. The prediction engine is assisted by a monitor to determine the objects to be obtained and the bandwidth. The weight information is also stored by making use of cryptographic techniques.

In [10], the author focuses on predictive prefetching. Author identifies two factors, “page accesses” and “noise”. The noise influences the technique for processing the accesses made by the users which is then used for measuring the efficiency of the algorithms. Using the above-mentioned factors, the author presented a framework describing the “markov predictors”. The author also described a new way to interpret the prefetching algorithms, called “markov predictors”. This way displayed the important aspects that alter the performance of the algorithms. Additionally, the author develops an innovative method and proves formally that it is indeed a “generalization of the current methods”. A broad investigative and exploratory examination of all calculations shows that the proposed calculation beats existing ones by consolidating their preferences without exhibiting their inadequacies.

Existing System

A. Caching Architecture

With the speedy development of the web in a recent years, the principle objective was to give speedier access to information and administration of cached information. A question that arose was, “how to build lots of caches?”. A few models were proposed for satisfying the above target and they are:

1) Single and Multiple Caching: A relatively simple algorithm was introduced by Shim et al. [12] consisting of only a “single proxy cache”. They thought about consistency, upkeep and substitution approaches in these web proxies. Soon after, Fan et al. [13] portrayed a convention for multiple caching called “Summary Cache”. Here, every proxy keeps up a rundown of the cache registry of each partaking reserve. Queries that result in a hit, the “summary cache” is looked into before sending data.

2) Hierarchichal Caching: The idea of “Hierarchical caching for web”, is the arrangement which includes caches that are situated at various diverse levels of network. At the bottom there are “client caches”. If there is a cache miss in the client cache, then the request is forwarded to the next level called Institutional cache. When a cache miss is encountered in the Institutional cache, the request travels to the Regional cache. The “National cache” handles the request when a cache miss occurs in the Regional cache. If the request is not not responded at any cache level, then the “National cache” is responsible for gratifying the

request by directly querying the “original server”. Data that is fetched from the “origin server” travels down all the levels till the Client cache leaving a copy in each intermediate level.

3) Distributed Caching: In the distributed caching design, there are just “institutional caches” at the end of the system. These caches co-work with each other to resolve the misses.Due to the absence of intermediate caches, other sharing mechanisms are used by institutional caches in order to consolidate the queries presented by the below level caches:

- “Institutional caches” work together with other co-operating “institutional caches” to handle all local misses.
- “Institutional caches” can look after the meta data of the contents of other co-operating caches. They are periodically exchanged.

4) Hybrid Caching: Hybrid caching is mix of “hierarchical” and “distributed” caching methods. Caches at every stage of progressive system co-work with each other at a similar level or with larger amount caches utilizing dispersed caching. Rajeev Tiwari et al. [15], this provides a resolution for “robustness and scalability problem” in internet caching. The idea of clustering is utilized alongside the element of dynamic designation of requests by keeping up summary of surrounding systems. They give the idea of dealing with the heap of over-burden server by exchanging solicitations to less loaded intermediary servers.

B. Web Prefetching

It is a technique which reduces the web latency by predicting the web objects that are to be accessed by the user in future and prefetching those objects before user actually demands for them. Some of the techniques are:

1) Markov Model Based Approach: In [16] Markov chain model is used to implement an access time and frequency based page rank like algorithm for conducting web page predictions. Time length and frequency are used as primary factors in the page rank algorithm to give a higher ranking to the pages that queried more often or open for a longer time. Firstly, the user’s navigational path is utilized to make a graph G. Then the graph is expanded by including all those pages that point to the pages already in G and the weighted links between them is found. The length of the path of the sub graph depends on the order of the Markov model used. Until this predefined depth is reached, the same process is followed for the newly included pages in the same graph. Then the algorithm computes the local ranking for the pages in the sub graph and provides prediction list to the current user in decreasing order based on the ranking values. In this paper, authors used 1st order Markov chain model to expand the sub graph which is memory less to declare the weights in the directed graphs and to expand the sub graph.

2) Clustering Based Approach: In this paper [1] author proposed an approach based on rough set of clustering which is used to structure the clusters of sessions. It classifies uncertain, imprecise or incomplete information in terms of data acquired from past experience. Using rough set clustering only

meaningful sessions are obtained in which user spends his quality time. In this paper, author presented an algorithm called Rough Set Clustering (RST) which uses the concept of rough sets to calculate equivalence between objects and then finds lower approximation and upper approximation. Lower approximation is the union of all equivalence objects which are contained in the target set, which is generally supposed by the user. The higher estimation is the union of all similar objects which have non-empty intersection with intended set. Authors proposed the concept of PPE (Prediction Prefetching Engine) which resides at proxy server. When user requests for a page it matches that request with existing rough set clusters and then decides whether to prefetch the page or not. By clustering using RST, only the meaningful sessions of Web log are fed to rule generator phase of PPE and thus the complexity of PPE is also reduced.

Proposed System

We propose a system wherein the techniques of clustering and prefetching are used to assist in faster web object retrieval. Users interact with the web server in terms of Sessions. Clustering of web sessions is equivalent to creating clusters of users. Subsequently, the web content is separately prefetched for each cluster. A session is represented by (*session-id,* {*page-id, time*}), where time represents the amount of time the user spent in a particular session.

A. Clustering Phase

Web server's log files are scanned, and records of user access are obtained. Clustering involves two steps: Generalization of sessions and the application of BIRCH on these sessions.

1) Generalization of Sessions: Sessions are first generalized to reduce performance overheads and to increase the chances of obtaining user groups who share common interests. Generalization is applied by creating a page hierarchy and applying Attribute-oriented Induction to it.

Page hierarchy is created by initializing the home page as the root node, scanning the URL, creating a node for each prefix of the current page and linking the nodes with each other. The current page is called a simple page and the prefix pages are called general page.

In Attribute-oriented Induction, each page is replaced by its corresponding general pages. User defines the threshold for the level of the general pages. If two pages in a session have the same or higher level general page, then one is deleted followed by having its timesummed to the others. Thereby, generalizing the session.

2) BIRCH: The

3) generalized sessions are grouped utilizing the BIRCH calculation. BIRCH forms a Clustering Feature (CF) tree which is the after effect of grouping by gradually embeddings objects, shown by vectors, into the CF tree. A CF tree is a multidimensional structure in which a non-leaf node stores entries of (CF_i,

pointer_to_child_i) and a leaf node stores entries of CF_i where CF_i is a CF vector. A CF vector is a triple containing the number of data objects (n), the linear sum (LS) of the n objects and the square sum of the objects (SS) in the subtree rooted at a child.

CF = (n, LS, SS) (1)

The insertion of a new vector is based on a certain distance measure, say Euclidean distance, which is used to determine the closest child node. Diameter threshold (T) decides the incorporation of the new object into an existing leaf node. If permitted then the entry's CF vector is updated, otherwise it is inserted into an empty entry in the leaf node. At the point when there are no vacant sections left, the leaf node is part into two, and an unfilled entry in the parent node is utilized to record the new leaf node.

B. Prefetching Phase

The web objects are prefetched, using a suitable prediction mechanism after all the sessions have been clustered. The proxy server needs to maintain separate caching regions for each cluster of sessions, as the access history of each cluster is used to predict and prefetch web objects. After prefetching and its subsequent usage by the users of the cluster, a cache replacement policy is used to substitute it with the next predicted webpage.

1) Prediction Mechanism: Markov modeling is used to prefetch web objects. Access patterns are analyzed based on a each cluster basis, and this uses the combined access patterns for forecasting. A Markov graph is developed using these access patterns and it is used to make prefetching predictions. In the graph, a node is a Web Object; a link represents the order of web object visitation by a user in the cluster and a weight is assigned to the link with the transition probability from the first node to the second. A search algorithm then traverses the graph and computes the access probability for the consecutive nodes. This algorithm decides how many web objects to load based on the probability, bandwidth and cache size.

2) Cache Replacement Policy: When a web object has been used and is no longer needed, it needs to be replaced by another prefetched web object. Least Recently Used (LRU) cache replacement policy is used to replace the unneeded web objects by keeping track of which web object was used when. The least recently accessed web object is removed by this algorithm. This removal operation is performed until there is sufficient space for new web objects to be cached.

Simulation

We used the log files of different websites which consisted of different number of pages such as 50, 75 and 100. We used a college website for 50 pages and commercial websites for 75 and 100 pages. On average, each page was found to be of the size 3 MB. We obtained the log files from their proxy servers with the size of 150 MB. Each website's log file was collected after one (100 Users), two (200 Users), four (500 Users) and thirteen weeks (1000 Users) which were used as training set. After each of these weeks,

the prefetching algorithm suggested some pages, which were then prefetched and cached in the proxy server. The successive week's log file was analyzed for the number of accesses of the cached pages using which the bandwidth saved and the hit ratio were calculated.

Conclusion

This paper has presented an innovative approach for the prefetching of pages. It clusters users based on their historical preferences using BIRCH and prefetches pages for these clusters with Markov modeling. Unlike previous prefetching approaches, this approach predicts web pages for clusters of users which considerably reduces the consumption of bandwidth and internet traffic. It also increases the throughput of the network. Experimental outcomes demonstrated an accomplishment of around 60% hit-ratio due to the method of clustering which was followed by prefetching. It was found that the hit ratio increased, correspondingly, with increase in the number of users. Ideal situation was found to be encountered when the number of users increased proportional to the number of pages.

Future work will be focusing on: The increase in the average available cache size of the proxy servers will enable more pages to be cached which will increase the hit ratio. Additionally, the implementation of a more efficient cache replacement policy, with reduced Average Access Time, will drastically decrease the time taken for caching and retrieval of web pages.

Declarations

1. ETHICAL STANDARDS

I. This research is carried out by me and my research supervisor.

II. We have not received any financial grant from any organization or institution. During our research we have not done any damage or harm to any creature.

III. If our paper gets accepted, we will give copy right form to publish it in open access journal.

IV. This paper has not been published in any part of the conference or journal.

2. All the referred papers are cited in our work.
3. I have done collection of data, interpretation, design and implementation of the research, analysis of the results and to the writing of the manuscript and final approval is taken from supervisor.

References

[1] Ms. Jyoti, Dr. A. K. Sharma, Dr. Amit Goel, and Ms. Payal gulati "A Novel Approach for clustering web user sessions using RST" International Conference on Advances in Computing, Control, and Telecommunication Technologies, IEEE Computer, 2009.

[2] Cheng-Zhong Xu and Tamer I. Ibrahim, Towards Semantics-Based Prefetching to Reduce Web Access Latency, Proceedings of the 2003 Symposium on Applications and the Internet, 2003

[3] Sadhna Ahuja, Tao Wu and Sudhir Dixit, On the Effects of Content Compression on Web Cache Performance, International Conference on Information Technology:

[4] Poornalatha G, Prakash S Raghavendra, Web Page Prediction by Clustering and Integrated Distance Measure,2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 2012.

[5] Bin Wu, Ajay D. Kshemkalyani, Objective-Greedy Algorithms for Long-term Web Prefetching, Third IEEE International Symposium on Network Computing and Applications, 2010.

[6] Josep Dome`nech, Julio Sahuquillo, Jose A. Gil, Ana Pont, The Impact of the Web Prefetching Architecture on the Limits of Reducing User's Perceived Latency, IEEE/WIC/ACM International Conference on Web Intelligence, 2006.

[7] Aviploum N. Eden Brian W; Joh Trevor Mudge, Web Latency Reduction via Client-Side Prefetching, 2000

[8] Vlad Olaru Walter F. Tichy, Request Distribution-Aware Caching in Cluster-Based Web Servers, third IEEE International Symposium on Network Computing and Applications.

[9] Jung J, Lee D, Chon K. Proactive Web caching with cumulative prefetching for large multimedia data. Computer Networks 2000; 33(1–6):645–55.

[10] Alexandros Nanopoulos, Dimitrios Katsaros, and Yannis Manolopoulos; a Data Mining Algorithm for Generalized Web Prefetching, IEEE Transactions On Knowledge and Data Engineering, 2003.

[11] V. Almeida, A. Bestavros, M. Crovella and A. de Oliveira; "Characterizing Reference Locality in the WWW" Proc. IEEE Conf.Parallel and Distributed Information Systems (IEEE PDIS '96), pp. 92-103, Dec. 1996.

[12] J. Shim, P. Scheuermann and R. Vingralek (1999). Proxy Cache Algorithms: Design, Implementation, and Performance, IEEE Transactions on Knowledge and Data Engineering, v.11 n.4, p.549-562.

[13] L. Fan, P. Cao, J. Almeida, and A. Broder, Summary cache: A scalable wide-area web cache sharing protocol, in Proc. SIGCOMM'98, Feb. 1998, pp. 254–265.

[14] A. Chankhunthod et al., A hierarchical internet object cache, in Proc. 1996 USENIX Technical Conf., San Diego, CA, Jan. 1996.

[15] Rajeev Tiwari, Lalit Garg, Robust Distributed Web Caching Scheme, in International Journal of Engineering Science and Technology in ISSN: 0975-5462 Vol. 3 No. 2 Feb 2011, pp 1069-1076.

[16] Y. Z. Guo, K. Ramamohanarao, and A. F. Park, “Personalized PageRank for Web Page Prediction Based on AccessTime-Length and Frequency,” International Conference on Web Intelligence, July 2007.

[17] Qiang Yang and Zhen Zhang, Model based Predictive Prefetching, 2001.

Tables

TABLE I. SAMPLE USAGE DATA

Rec No.	User	Page
1	u90	p19
2	u99	p26
3	u9	p32
4	u55	p45
.	.	.
.	.	.
98	u31	p12
99	u63	p35
100	u50	p25

TABLE II. STATISTICS ON HIT RATIO AND BANDWIDTH SAVED FOR 50 PAGES

No. of Users	Page Frequency (Threshold = 3)	User Frequency (Threshold = 2)	No. of prefetched pages	Hits	Hit Ratio (%)	Band-width Saved (MB)
100	16	10	4	15	30	45
200	27	6	7	28	56	84
500	48	6	9	38	76	114
1000	50	6	10	45	90	135

TABLE III. STATISTICS ON HIT RATIO AND BANDWIDTH SAVED FOR 75 PAGES

No. of Users	Page Frequency (Threshold = 3)	User Frequency (Threshold = 2)	No. of prefetched pages	Hits	Hit Ratio (%)	Band-width Saved (MB)
100	10	9	6	17	34	51
200	23	5	6	24	48	72
500	60	5	6	34	68	102
1000	75	5	8	39	78	117

TABLE IV. STATISTICS ON HIT RATIO AND BANDWIDTH SAVED FOR 100 PAGES

No. of Users	Page Frequency (Threshold = 3)	User Frequency (Threshold = 2)	No. of prefetched pages	Hits	Hit Ratio (%)	Band-width Saved (MB)
100	9	5	5	12	24	36
200	19	5	5	20	40	60
500	61	6	6	30	60	90
1000	97	6	8	35	70	105

Figures

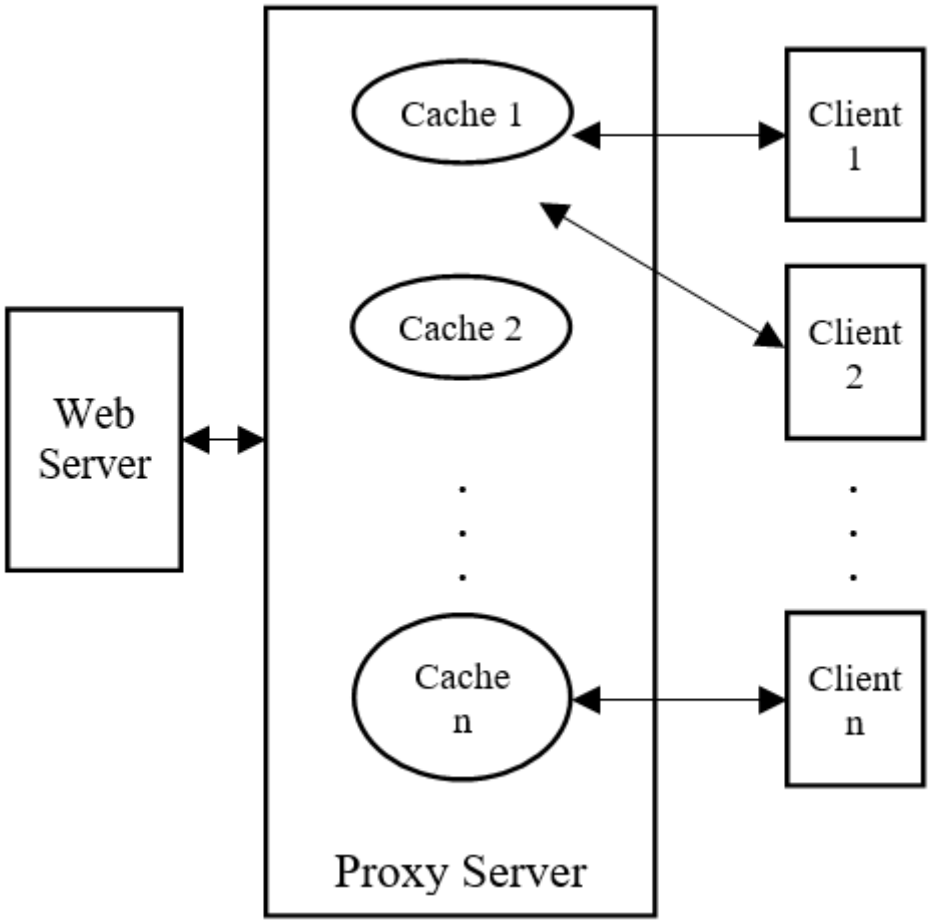

Figure 1

Architecture of the System.

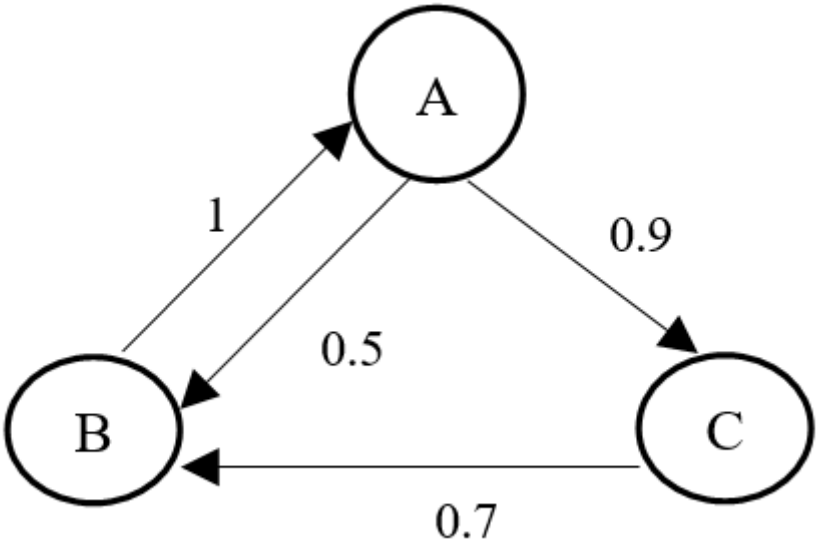

Figure 2

Markov Graph

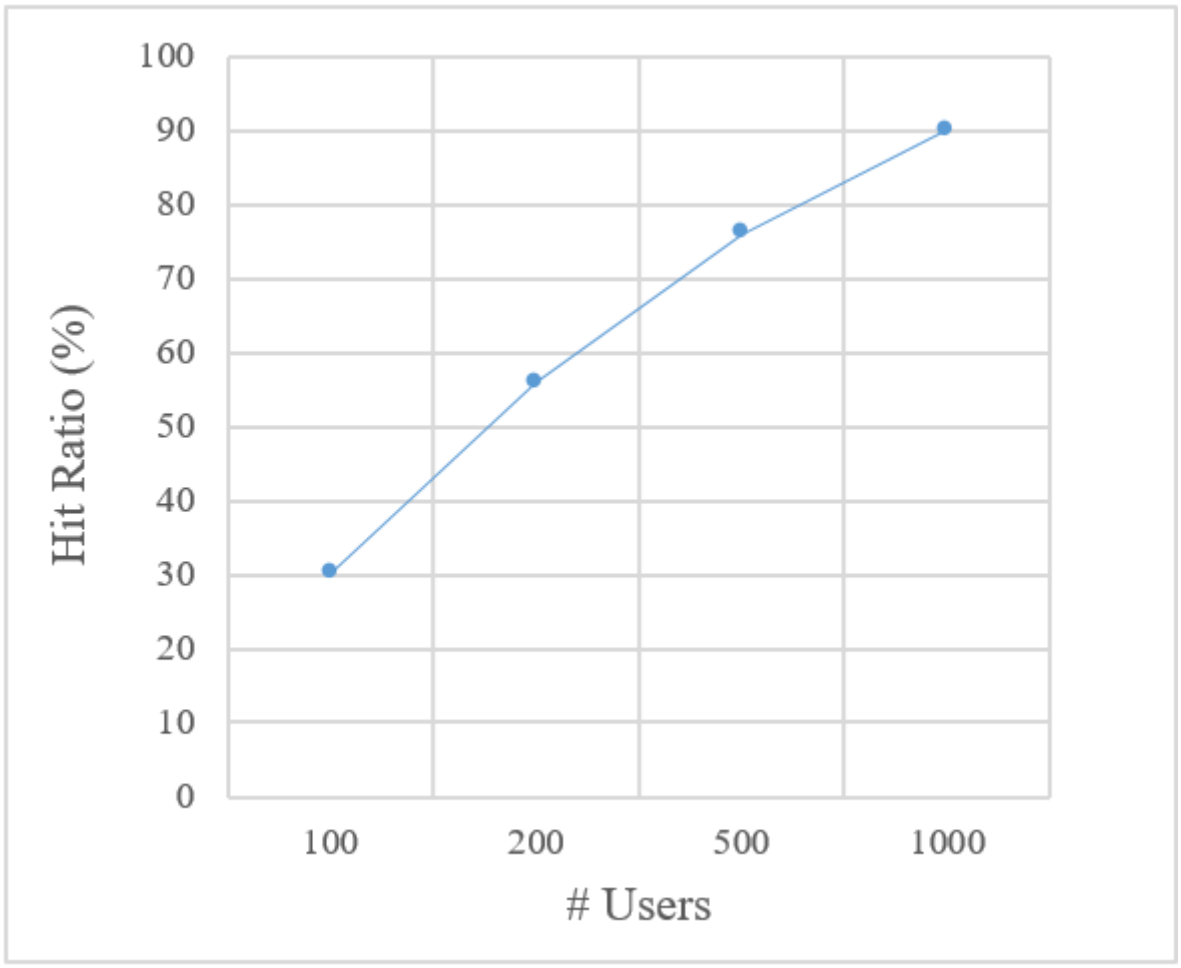

Figure 3

Bandwidth Saved against varied number of Users for a website of 50 pages

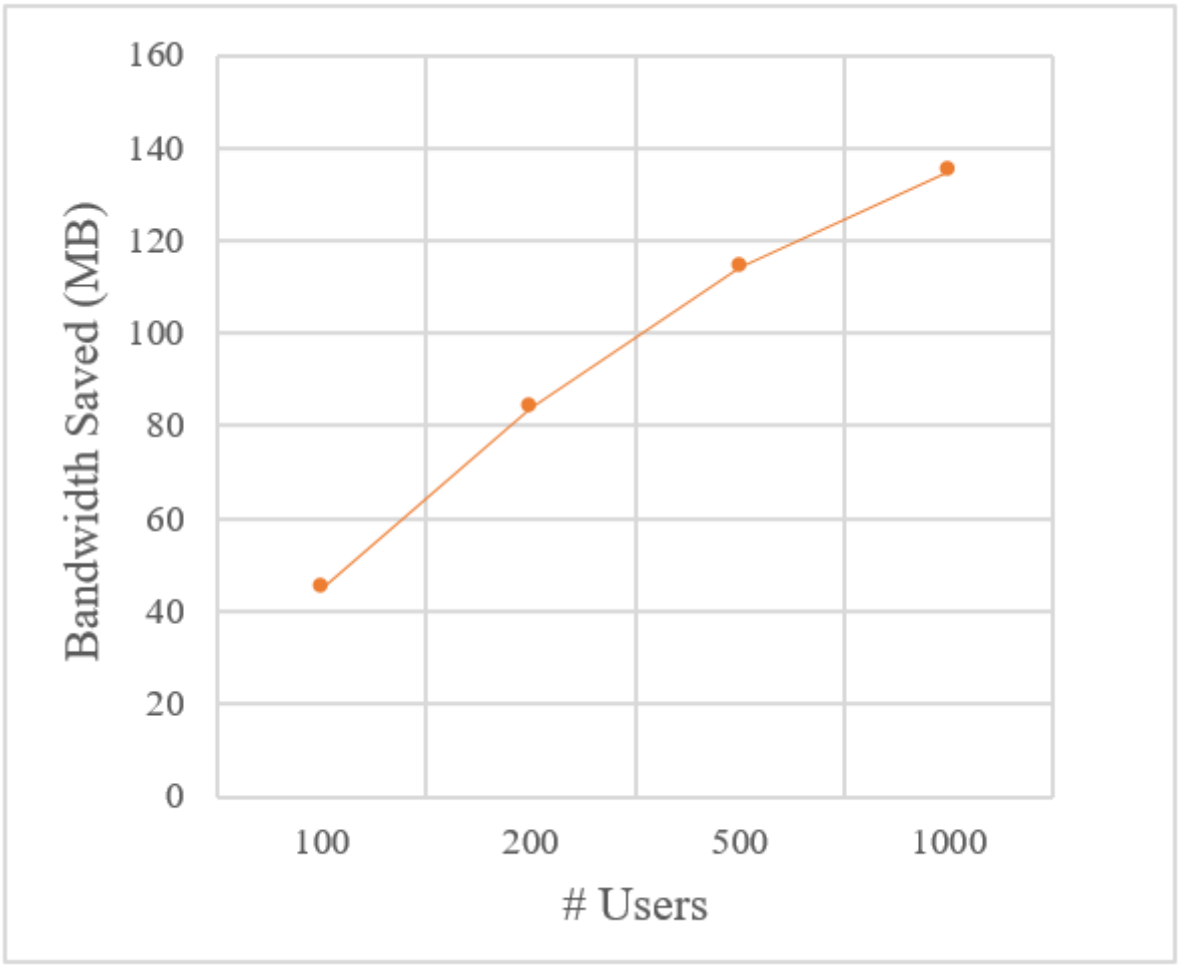

Figure 4

Hit Ratio against varied number of Users for a website of 50 pages

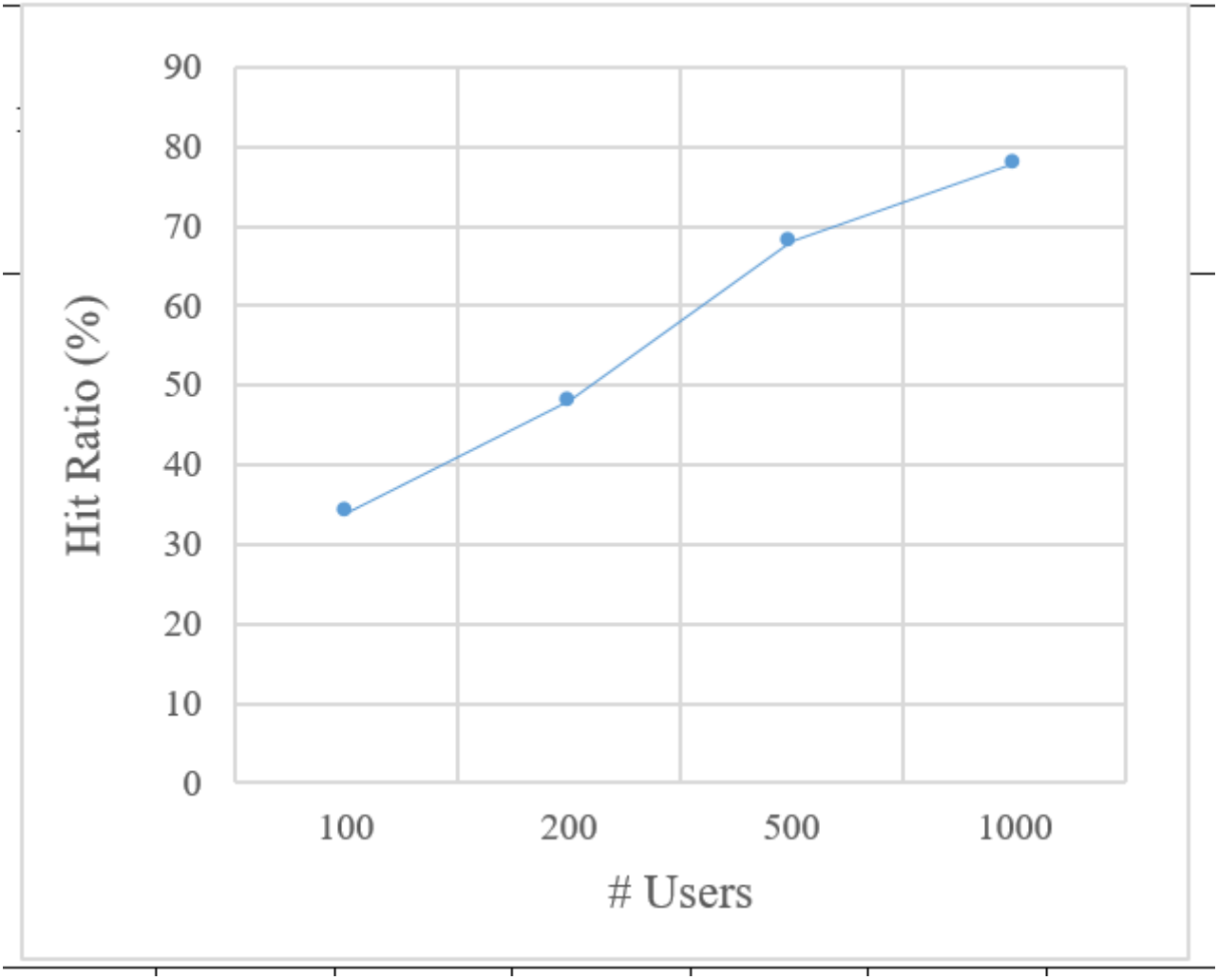

Figure 5

Bandwidth Saved against varied number of Users for a website of 75 pages

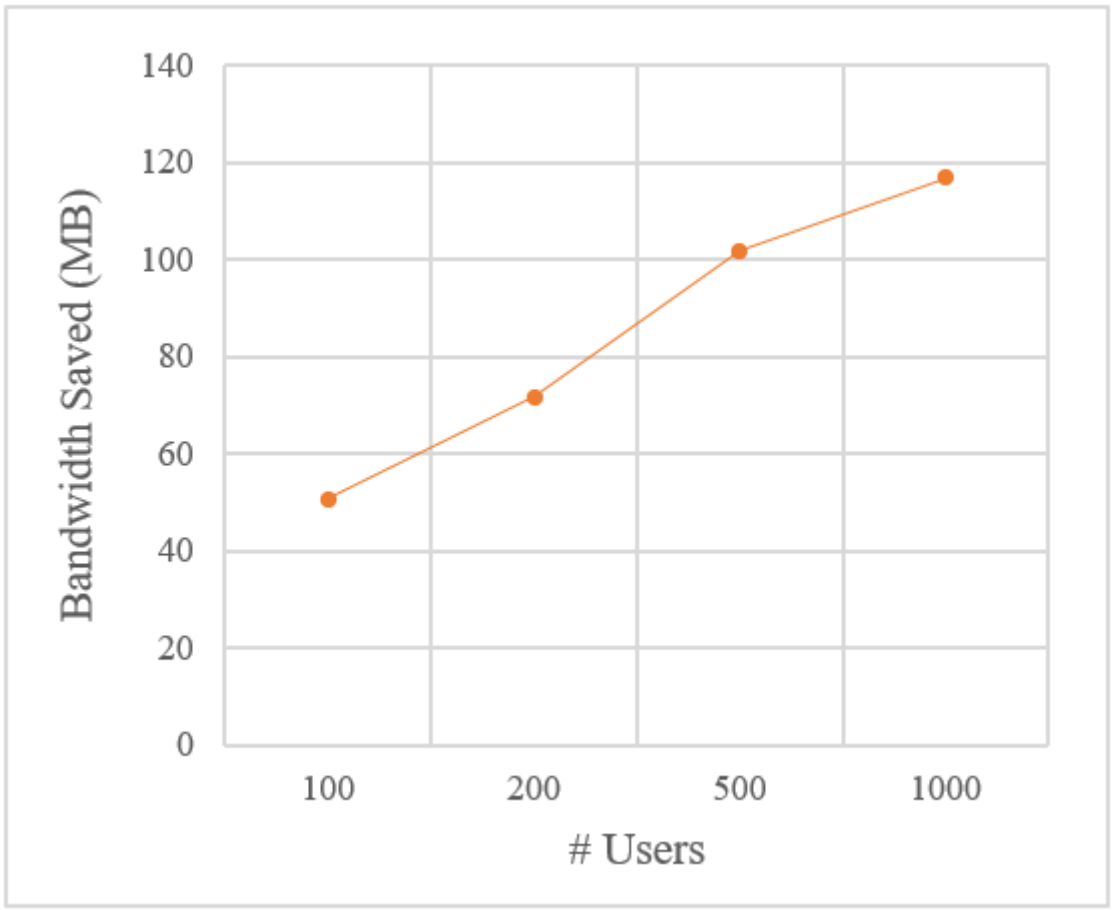

Figure 6

Hit Ratio against varied number of Users for a website of 75 pages

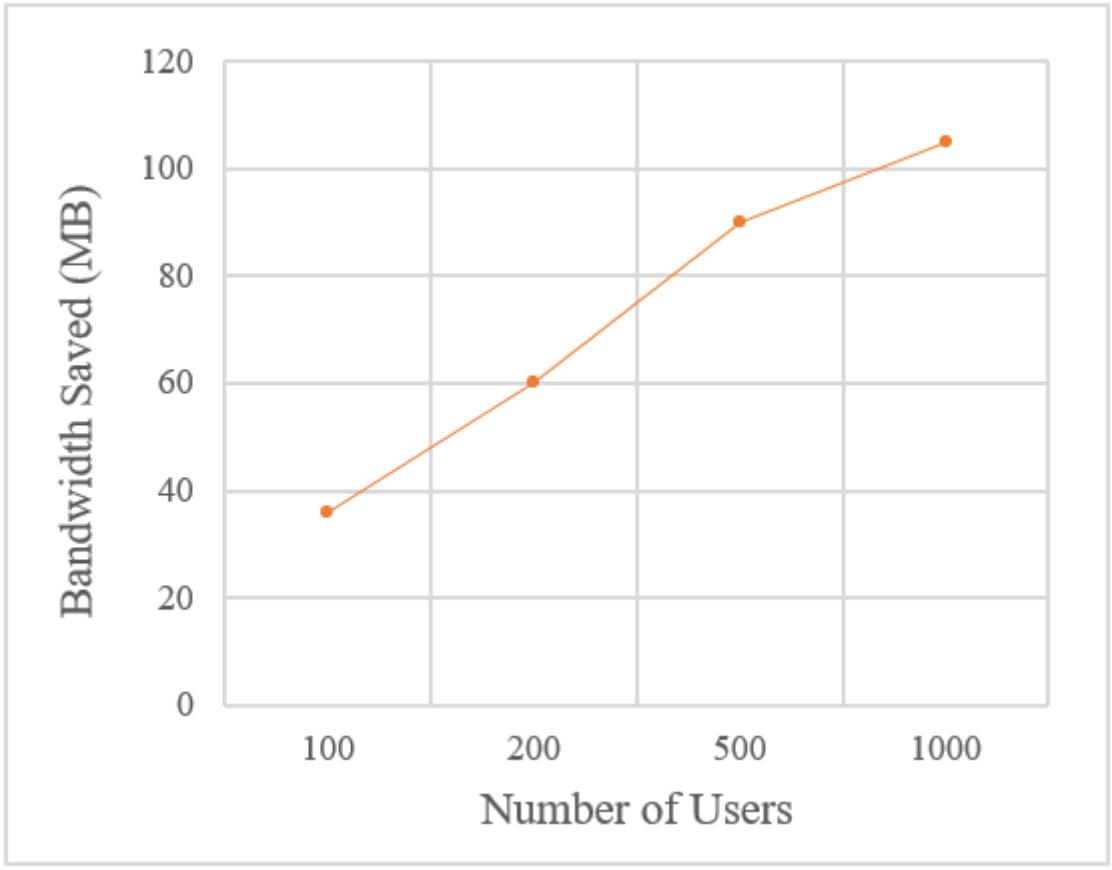

Figure 7

Bandwidth Saved against varied number of Users for a website of 100 pages

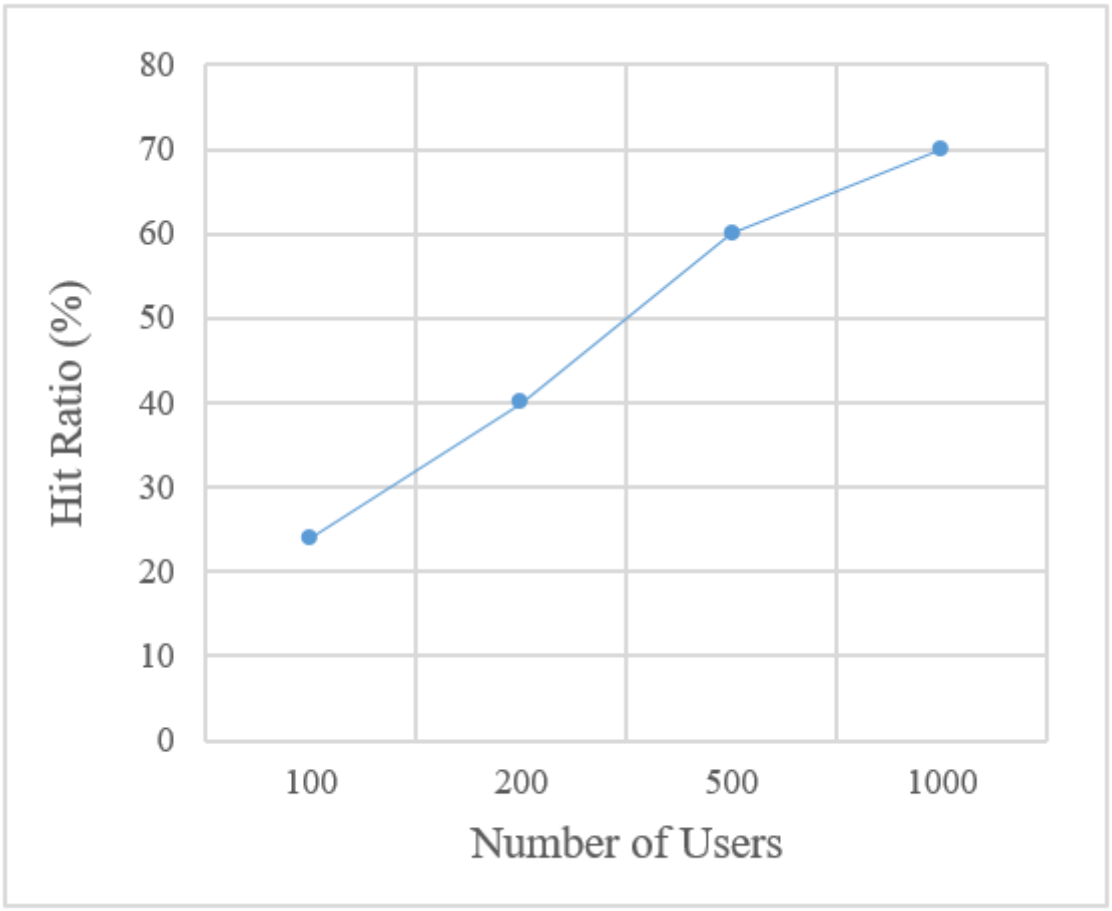

Figure 8

Hit Ratio against varied nunber of Users for a website of 100 pages