An Overview to Pre-fetching Techniques for Content Caching of Mobile Applications

Afiq Aisamuddin Mohd. Shariff, Norliza Katuk and Nur Haryani Zakaria
School of Computing, Universiti Utara Malaysia, Kedah, Malaysia
afiq.aisamuddin.my@gmail.com

***Abstract*— The Internet and web have been the main resource for various types of information for majority of people in the world since a decade ago. With the emerging of smartphone technology, the web content is also available for mobile users that connect to the Internet through cellular network. Although mobile users are able to access content from the web; nevertheless, they always experience long access latency due to the speed of the network. Many research and developments have been implemented to help users to access content faster when using mobile devices with cellular network. Caching is one of them; a commonly used method for storing recently-accessed contents so that they can be used in the future. Content caching (CC) can reduce latency, which in return guarantees faster access to the content. Recently, there is a growing interest among researchers and developers in studying a more proactive technique to improve CC called pre-fetching (PF). PF is a method that caches selected content before it is actually needed. By embedding PF in CC, it is believed that latency could be reduced significantly. By looking at this promising approach, this paper introduces the PF techniques that could be suitable for CC in mobile environment. The paper aims to assist researchers by providing a better understanding on the existing PF techniques so that improvements can be proposed where possible.**

***Index Terms*— Cache Management; Content Prediction; User Interest Degree.**

I. INTRODUCTION

The advancement in information and communication technology (ICT) contributes to an apparent change in the way how people work and communicate. It includes mobile computing; an emerging trend where people use mobile devices (e.g., smartphones and computer tablets) to perform various tasks in the same way they use desktop computing. Consequently, many applications are now developed with both web and mobile versions. Either web or mobile environment, users always anticipate rapid access to the speed and bandwidth, and intermittency of data transfers [8]. Hence, an approach to assist users in getting faster access to web content is needed. Therefore, PF is seen as a promising approach to improve access to web content in general and CC particularly. The following section introduces the readers to the basic concepts of CC. Then, the subsequent section elaborates PF and the existing techniques. The last section concludes the paper.

II. BASIC PRINCIPLE OF CONTENT CACHING

The cache technology started after the invention of virtual memory at the same time when the computer machines were introduced. As a result of semiconductor memory scarcity, content and applications.

Content caching (CC) has a crucial role in providing rapid access to web content despite the availability of the high-bandwidth network. Consequently, CC can enhance system performance as it stores content that could possibly be used in the near future. This method has proved to lessen server load, reduce access latency and decrease network traffic [1][2]. In this case, content that are accessed more frequently can be loaded faster.

In the context of mobile computing where users are connected to the Internet through wireless or cellular network; CC has become an important topic of discussion. Recently, there is a growing interest among researchers in studying a more proactive approach to improve CC called pre-fetching (PF). Parma and Verma [3] argued that CC is not beneficial for dynamic web content that is available in most of todays' web-based applications. Hence, prediction of content that users are highly likely to access could improve the overall CC efficiency. This is where PF takes its role.

Furthermore, many servers log their users' activities; consequently, these logs are beneficial as they can be used to create a model that can predict access to the future content. Frequent access patterns of web logs can be obtained based on these models. PF embedded in CC can improve system performances in terms of hit rate [4]. When the cache hit increases, it can result in fewer round trips to the server [5]–[7]. It implies fewer loads on the server, which improves scalability. When content loads faster, the users gain a better experience in accessing the mobile applications.

This paper aims to provide an overview to the CC and PF from the perspective of content access within mobile environment. It is known that mobile users connect to the Internet using cellular network which is expensive, limited in terms of

the mainframe computers, such as those used in the1960s, utilized a physical memory hierarchy that was complicated and normally mapped on virtual memory. Due to the resulting problems, innovators looked for a way to access data in the main memory faster which led to the innovation and development of cache [9].

Cache is an intermediate fast memory [10] which stores duplicated data items from the server [10]–[14]. It is a memory component that stores temporary data to be used for future access, and it serves for a faster loading of data and applications [14]. The purpose of the cache is also to reduce the average time used to access the main memory. The same concept has been applied in accessing web content from the Internet where CC comes in place.

CC is a well-known performance enhancement technique that speeds up data look-ups. The technique allows the client to have fast access to the data without having to request from the source [15]. It also reduces the amount of delay to access the resources during a client-server interaction; hence, reducing the latency between the client and server [16], [2]. It makes the applications more responsive and speeds the access to get the resources [17], [18]. Furthermore, the server load can be reduced when the clients use less resource from the server.

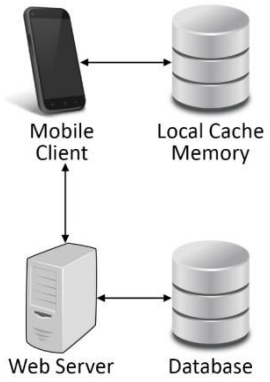

Figure 1: CC in mobile environment

Within a mobile environment, a mobile client (i.e., mobile devices such as smart mobile phones and tablet computers) communicates with web servers to access content from database. Once the interaction is completed, the local cache memory of the mobile devices stores and maintains the content. Figure 1 illustrates the basic CC process within a mobile environment.

A. CC Efficiency Metrics

There are two metrics used to measure the efficiency of a particular CC technique namely (i) cache hits ratio, and (ii) caches miss ratio. Cache hit ratio is used to measures the number of access to cached content at the cache memory. While, the cache miss ration measures the number of client requests in which the content is not available in the cache. Figure 2 illustrates the process of a cache hit for a mobile application which consists of the following activities:

1) The user requests data.
2) The mobile client checks its cache memory.
3) The mobile client retrieves the data from the cache memory.
4) The mobile client delivers the data to the user.

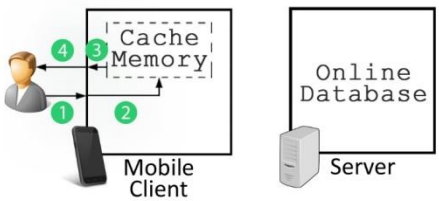

Figure 2: The cache hit process

Meanwhile, the state where cache memory does not contain the data requested by an application or component is called a cache miss. The client needs to communicate with the server to obtain the content. Figure 3 illustrates the process of a cache miss. From the figure, the process of a cache miss consists of the following activities:

1) The user requests content.
2) The mobile client checks its cache memory, but there is no cached content in the cache memory.
3) The mobile client requests the content from the server.
4) The server processes the request and delivers a response to the mobile client.
5) The mobile client stores the content into its cache memory.
6) The mobile client retrieves the content from its cache memory.
7) The mobile client displays the data to the user.

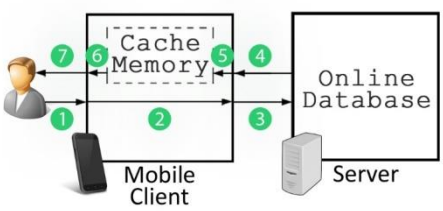

Figure 3: The cache miss process

B. Cache Invalidation

CC is attractive because it saves time by eliminating the cost of reading the source when users need the content again later. However, the original resources can have some changes. Cache invalidation is used to ensure every cached copy in the system eventually reflects the updated content, without incurring costs that exceed the benefit of CC. In other words, content that already cached in the mobile client can be outdated if there is no cache invalidation.

Cache invalidation plays an important role in maintaining the consistency of content between the client and the server [10]. In this regard, cache invalidation will help in caching valid resources. With cache invalidation, it will prevent any inconsistent content between clients and servers. With invalidation policy, whenever the resources from the server are changed or updated, the server will send an invalidation to the client's that have a same copy of the cache file [19]. After the client receives the invalidation, it will update the latest cache content from the server. Besides that, if the cache content is no longer valid, it will be substituted with a new content after the old one is removed [20].

III. PRE-FETCHING (PF) AND THE EXISTING TECHNIQUES

PF is an approach for predicting the future items of interests. It brings them asynchronously into the cache hence; when required, the content will be readily available [21], [22]. A perfect PF technique is able to predict the next set of requests and pre-load those content into the cache, and the cache would satisfy and fulfill such accesses, instead of using web servers to retrieve the content [23]. PF techniques predict the next set of records, files, and pages that will be requested by users and utilize this data to pre-load them into the server's cache.

In web PF, experts and scholars have studies various aspects of it. For example, Padmanabhan and Mogul [24] proposed a hyperlink-based PF mechanism. Researchers have also used data mining techniques for suggesting the predicted content [25], [26]. Based on the log records for web access, association rules can be established to predict access to object sets. Data mining method is suitable for the server-based PF in which it will analyze and pre-fetch data in advance. Pal et al. [27], [28] proposed a method, which

uses Markov model for identifying the predicted content.

Manoharan and Deng [29] show the advantage of PF in terms of reading an e-book. An Internet user reads a twenty-page HTML formatted online book, one page will take five minutes to finish. Assuming that downloading a new page in the web browser will take ten seconds. If no PF occurs earlier, the reader will wait for the next page to load when he/she clicks the next page button. Hence, the total waiting time will be two hundred seconds. If the browser can predict the user's next movement, it will pre-cache the next page; hence, as the user clicks to the next page, the browser will retrieve and display contents from its local cache. In this case, the user can reduce the waiting time for downloading every new page. In the back end, it still cost the browser two hundred seconds to pre-cache the twenty pages.

Table 1
Comparison of CC and PF in terms of locality, architecture and content placement

Approach	Locality	Architecture	Content Placement
CC	Temporal	Pull-based	Reactive
PF	Spatial	Push-based	Proactive

PF is different from the regular CC particularly the way of the mechanism used to determine the content to be cached. The cache memory of the mobile devices is pre-loaded with content even before they are being used. The basic principle of PF is that the technique determines and stores content that are expected to be applied based on the communication with the server. On the other hand, the common CC mostly deals with content that have already been requested by the users, and the client machine performs the caching. Table 1 shows the comparison of CC and PF in terms of locality, architecture and content placement.

Figure 4 illustrates the PF process. From the figure, the process consists of the following activities:

1) The PF engine requests content through the mobile client.
2) The mobile client contacts the server.
3) The server requests the data from the prediction engine.
4) The request is processed by the prediction engine, and the query is sent to the online database.
5) The online database processes the query, and the data is sent to the server.
6) The data will be retrieved by the server, and a response will be delivered to the mobile client.
7) The mobile client will send the data to the PF engine.
8) The PF engine updates the data into the mobile client's cache memory.

Apart from the cache memory and online database, there are two main components in a PF process; (i) PF engine, and (ii) prediction engine [30]. The PF engine chooses and decides the pre-fetch actions. Several factors have been considered to predict what to pre-fetch, such as location, log data, and user behavior. Du and Wang [31] analyzed the log data to obtain the users' behavior, calculated and ranked the item that needs to be pre-fetched based on the degree of interests. By analyzing the log data, the algorithm can predict what clients are interested in. Then the pre-fetched content is combined with the CC algorithm to achieve cache hit.

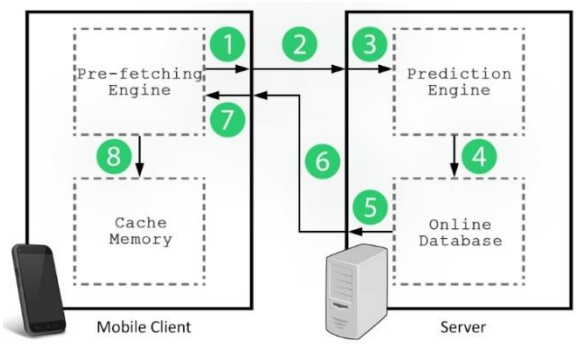

Figure 4:. The PF process

Besides that, there are other studies conducted on the degree of user group's interest. Zhang et al. [32] proposed Prediction Greedy-Dual Size Frequency (P-GDSF) which used the client logs data and patterns of navigation in order to predict the browsing paths of the users. With this enhancement, P-GDSF improves hit ratio.

Users' interest can also be predicted based on their locations. Location Based Services (LBS) is used to determine the user's location so that the algorithm can predict the future movements; thus will pre-fetch the data ahead. Chavan et al. [33] proposed a Markov Graph Cache Replacement Policy (MGCRP) that predicts the users' location ahead by using Markov Model. The user's location will be recorded by the server to analyze the patterns and clustered the location. After that, the data will be ready to be pre-fetched by a client. This process can reduce the latency between client and server.

The user interest degree can be determined according to the characteristics, and by the characteristics of the mobile network. It is able to forecast current users' behaviour, based on the user's profile or factors, such as location, and time. All web access log can be used to predict what users will access in the future [31].

A. Classification of PF Techniques

PF techniques can be categorized into three; (i) probability based PF, (ii) clustering based PF, and (iii) weight-functions based PF. Probability based PF presents a natural approach for prediction, where the content is pre-fetched based on probability, which is determined by past data access. Furthermore, in this method, the probabilities will be determined by the request sequence which follows a specific pattern.

Meanwhile, clustering based PF techniques combines CC and PF [34]. Decisions are made based on information about the content with clusters previously fetched. This technique assumes that content which has been fetched previously will have more probability to be requested next. The clustering-based PF can integrate CC and PF effectively. Lastly, the weight-function-based PF adopts a function which consists of several factors such as prediction by partial match model (PPM), weblogs and web page size. Table 2 presents the advantages and disadvantages of the three techniques.

Table 2
Classifications of PF techniques.

Techniques	Advantages	Disadvantages
Probability Based	• Use history access data to calculate PF. • Can be used for content with close links and a clear pattern of request history. • Has good prediction rate and increase the cache hit ratio particularly for small size caches. • Can control the number of pre-fetched content. • The tree structure is used to record probability.	• Prediction algorithm could sometimes be problematic. • When a request comes, the most probable content that might be requested afterward must be determined
Clustering Based	• Use information about the previously fetched clusters containing pages. • Has more advantages if the links among the web pages are unclear. • At run time, the amount of memory available to the algorithm can be modified. • Compatible with low memory devices.	• Does not consider server workload and the cost of the network traffic and server workload. • Heavily reliant on cache replacement policies.
Weight-Functions	• Take into account the size of the web page. • Considers more than one factor, and not limited to user access pattern. This can decrease server workload and the cost of the network.	• Time-consuming and have a complex process.

B. PF Infrastructure

Extensive studies have been done on web content PF in the context of the desktop environment. However, web content on a desktop environment is different in its platform and infrastructure as compared to the mobile applications. There are few factors need to be considered, such as memory size, latency, and mobile devices capacity itself. However, PF is still one of the solutions to reduce latency between client and server.

PF techniques can be used in different locations, including the proxy, server, or client [35]. PF that is client-based works on single client user's exploration patterns across distinguished web servers. Meanwhile, sever-based PF focuses on all users' exploration patterns in retrieving a single website. Finally, proxy-based PF focuses on a group of users' exploration patterns across distinguished web servers. Subsequently, such method is able to illustrate the mutual interest among the community of users where several users will be sharing the PF contents.

The advantage of client data is that it is easy to partition user session and realize the personalized PF. However, the client PF location does not share the PF content with a different user, and it requires a lot of network bandwidth. On the other hand, the proxy PF location uses proxy log and current user request as the data for the prediction model. It reflects the common interest for a group of users, rather than just one individual, Furthermore, the proxy PF location shares PF content from different servers among servers; however, it does not reflect the common interests for a single website from all users. The last location is the server, where the prediction model uses data from server log and current user request. In this regard, this location is useful as it records single website access information from all users and can reflect all users' common interests better. On the other hand, this location is not able to reflect users' real browsing behavior, and it is difficult to partition user session. Moreover, additional communications between clients and server are needed to decide the PF content. Table 3 summarizes PF based on these locations where it is performed.

Table 3
Types of PF based on location.

Location for Prediction Engine	Data for Prediction Model	Advantages	Disadvantages
Client	Only log files of the single user.	Can personalize the PF based on user need.	• Not share PF content among users. • Needs a lot of network bandwidth.
Proxy	Only log files from a group of users.	Can set the PF based on a group common interest.	• Not reflect common interests for a single Website from all users.
Server	Log files from all users.	Can set the PF based on all user's common interest.	• Not reflect real browsing behavior from the user. • User sessions difficult to partition. • Between clients and server will have extra communications to decide which content to pre-fetch.

C. Examples PF Techniques

There are many PF techniques have been proposed in identifying the pre-fetch content especially for web environment. This section reviews ten PF techniques that have been reported in the literature.

1) Prediction by Partial Match

Prediction by Partial Match (PPM) [36] take into account the past URLs in a Markov prediction tree that is dynamically maintained. In this light, the present approaches either save the branches that have frequently accessed URLs or widely save the URL nodes through forming a tree with a fixed height in each branch. From

here, a new model for PF is proposed by creating popularity information in the Markov prediction tree. This model is called popularity-based PPM, where the tree is dynamically updated with a variable height in each set of branches. As a result, a less popular document will lead a set of short branches, while a popular URL can lead a set of long branches.

2) Semantic Web PF

Semantic Web PF technique [37], [38] can facilitate accurate predictions of pre-fetched web objects to fulfill the user's future requests with low latency. This technique is grounded client-side PF concept. Here, to service the user's request, the client pre-fetches web documents from the server directly and stores it in the local cache. This technique can significantly decrease latency in fulfilling user requests, as network latency is a presence to retrieve documents that are locally cached [22].

3) Domain Top Approach

Shin, Seong and Park [39] proposed the domain top approach for web PF. The proxy's active knowledge of most popular domains and documents are combined and the proxy calculates the most popular domains, as well as most popular documents in those domains to prepare a PF rank list. This method aims to enhance the hit ratio through proxy PF and creates a small burden on the network proxy and the proxy. Further, the proxy searches popular domains through accessed profiles to determine popular documents in each domain. Consequently, the proxy will construct rank list grounded on Top-Domain and Top-Documents. This technique can be implemented without making changes to the client and server.

4) Link PF

Fisher and Saksena as cited in [40] proposed link PF through a server-driven approach. Link PF comprises of proprietary syntax that provides web browser a clue on why a document should be pre-fetched. This is based on whether it will be demanded again the client. Here, the browser pre-fetch specific documents by following the instruction and directives for the proxy or web server. Thus, servers can control what content the browser is going to pre-fetch. In this regard, a browser's idle time is used to download a document that might be visited by the users. The web pages provide a set of PF hints to the browser and after the page is loaded successfully, the specified documents will be pre-fetched and stored by the browser in its cache. The pre-fetched document can be served up out of the browser's cache quickly when it is being visited by the user. In this regard, an idle browser would observe these hits and line up each unique request that needs to be pre-fetched.

5) Top 10 Approach

A top 10 approach [35] calculates the list of most frequently downloaded documents [35]. This technique is easily implemented in a client-server architecture, as it predicts the web object instead of the client characteristics on the web based on the access frequency. This approach integrates client access profiles and servers' active knowledge of the most frequently (top 10) downloaded documents. Servers will forward them these documents based on client requests and their access profiles. In this light, effective PF operations in this approach are based on the clients-servers' cooperation, where the server is responsible for serving the documents to its clients and calculating a list of the most popular documents (the Top-10) periodically.

6) Dynamic Web PF

The dynamic PF technique [41] utilizes proxy server's database to store the users' preference list. This enables each user to save a list of sites that need to be accessed immediately. To parse the web page, the intelligent agents are adapted to monitor the use of bandwidth usage and to maintain cache consistency, hash table, and preference list. Web traffic is controlled by increasing PF during light traffic and decreasing the PF at heavy traffic. Consequently, idle time in the existing network is decreased to ensure almost constant traffic. To store the list of URL accessed and its weight information, a hash table is maintained. The prediction mechanism will identify the URLs that need to be pre-fetched and construct a PF list to accommodate web page prediction based on the bandwidth weights and usage in the hash table. Consequently, the pre-fetch web pages will be stored in the pre-fetch area by the proxy server after the PF process. This technique is beneficial as the number of pre-fetched links is based on the current bandwidth usage, which increases proxy and client overhead.

7) Model-based Predictive PF

The Model-based Predictive PF [42] uses the combined model of web PF and web CC. The statistical correlation between web objects is the basis of this time-based prediction model. Instead of numbers, the prediction window shows a particular period of time. A logical graph called 'correlation graph' is constructed by the algorithm which presents the high correlation between pre-fetch web objects and web objects, which highly correlated to the currently requested object. The model creates the pre-Greedy-Dual-Frequency (GDF), which is a combination of the CC and PF algorithm is based on the Greedy-Dual (GD)-Size algorithm [43] and the enhanced GDSF [44]. The algorithm's key parts include cache, replacement manager, prediction queue and PF agent.

8) A Keyword-based semantic PF approach

The Keyword-based semantic PF [37] identifies the semantic preferences through the analysis of the keywords of URL anchor text contained in the documents that have been previously accessed in various categories [38], [37]. A keyword-based semantic PF is proposed, where, future requests prediction is grounded on the semantic preferences of web documents retrieved previously. This scheme proposes that a web browser's background proxy is able to trace the client's characteristics and identifies the semantic link between the documents [37]. Here, a neural network based semantic model is used and is capable of self-learning. Therefore, this technique has the capacity to pre-fetch never accessed URLs.

9) Adaptive PF Scheme

An adaptive pre-fetch scheme [45] can modify the aggressiveness of PF in web servers dynamically and utilizes a threshold to adjust it. Meanwhile, Fagni, Perego, Silvestri and Orlando [46] suggest an approach that can increase search engine performance by exploiting the temporal and spatial locality in the stream of queries processed. In addition, Jiang and Kleinrock [47] created an

adaptive pre-fetch scheme, where the number of pre-fetched files is determined by network conditions and the user access history. Further, it adapts user's browsing habits and history [48]. It comprises of two modules, the threshold module, and prediction; the prediction module revises the history and computes each file's access probability. In this regard, only files that have the equal or greater access probabilities to the pre-fetch threshold will only be pre-fetched.

10) Markov model for predicting web access

The Markov model and hidden Markov model [49] can be used to predict pre-fetched web pages based on the probability of web access probability. In the conventional Markov model, the sequence of web pages that has been accessed by the user is considered as an input to predict the page that the user will access next. On the other hand, low order Markov models are not capable of predicting the user's subsequent requests accurately, and the memory used will increase rapidly if higher order Markov models are used the memory required increased rapidly. In this regard, Xing a hybrid order tree, like Markov model is proposed by Dongshan and Junyi [49]. It is posited that this model can precisely predict web access and provides high coverage and good scalability. Furthermore, Jin and Xu [50] proposed a novel web PF approach that is grounded on the Hidden Markov Model. The HMM can be analyzed through user's browsing history and make decisions on semantic-based web PF. However, these models cannot predict a new web request as they are based on historical data.

IV. CONCLUSION AND FUTURE WORKS

This paper aimed at providing a better understanding towards CC and PF within mobile environment. An overview to basic CC has been given followed by the concept of PF. The examples of PF techniques that have been used for web environment were also discussed. As stated earlier, the mobile applications that runs on smart devices and connected to the network through cellular network may need different approach in PF. Hence, it is suggested that future works investigate the PF techniques for mobile environment particularly the cellular network. Two main perspectives can be focused that are efficiency and performance.

ACKNOWLEDGMENT

This work was supported by Ministry of Higher Education, Malaysia under the Trans-Disciplinary Research Grant Scheme (Ref: TRGS/2/2014/UUM/01/3/4, UUM S/O Code: 13170).

REFERENCES

[1] J. Xu, J. Liu, B. Li, and X. Jia, "Caching and prefetching for web content distribution," *Comput. Sci. Eng.*, vol. 5, no. 4, pp. 54–59, 2004.

[2] P. Cao, "Opportunities and Challenges for Caching and Prefetching on Mobile Devices," in *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop*, 2015, pp. 49–53.

[3] J. Parmar and J. Verma, "State-of-artsurvey of various web prefetching techniques," in *International Conference onInventive Computation Technologies (ICICT)*, 2016, pp. 1–7.

[4] S. Sulaiman, S. Mariyam, and A. Abraham, "Intelligent Mobile Web Pre-fetching (IMWeP) Using XML Technology," in *Computer Information Systems and Industrial Management Applications (CISIM), 2010 International Conference*, 2010, pp. 475–480.

[5] D. H. Alassi and R. Alhajj, "Enhanced Scalable Asynchronous Cache Consistency Scheme for Mobile Environments," 2011.

[6] Y. Eom, J. Kim, and B. Nam, "Multi-dimensional multiple query scheduling with distributed semantic caching framework," *Cluster Comput.*, vol. 18, no. 3, pp. 1141–1156, 2015.

[7] Y. Guan, Y. Xiao, H. Feng, C.-C. Shen, and L. J. Cimini, "MobiCacher: Mobility-aware content caching in small-cell networks," in *In 2014 IEEE Global Communications Conference*, 2014, pp. 4537–4542.

[8] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das, "Understanding traffic dynamics in cellular data networks," in *Proceedings IEEE INFOCOM 2011*, 2011, pp. 882–890.

[9] J. Huang, Q. Xu, B. Tiwana, Z. M. Mao, M. Zhang, and P. Bahl, "Anatomizing application performance differences on smartphones," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, 2010, pp. 165–178.

[10] J. C. M. J. Pamila and K. Thanushkodi, "Cache Management for Concurrent Transaction Execution in Mobile Wireless Environment," *J. Comput. Sci.*, vol. 7, no. 3, pp. 374–378, 2011.

[11] X. Wang, M. Chen, and S. Member, "PreFeed : Cloud-Based Content Prefetching of Feed Subscriptions for Mobile Users," *IEEE Syst. J.*, vol. 8, no. 1, pp. 202–207, 2014.

[12] M. Tradeoff, "Decentralized Coded Caching Attains Order-Optimal," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1029–1040, 2015.

[13] J. Dai, B. Li, F. Liu, B. Li, and J. Liu, "Collaborative caching for video streaming among selfish wireless service providers," in *GLOBECOM - IEEE Global Telecommunications Conference*, 2011, pp. 1–5.

[14] N. Zaidenberg, R. Gan, Y. Meir, and R. Aviv, "New caching algorithms performance evaluation," in *Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2015 International Symposium*, 2015, pp. 1–7.

[15] A. Parate, M. Böhmer, D. Chu, D. Ganesan, and B. M. Marlin, "Practical Prediction and Prefetch for Faster Access to Applications on Mobile phones," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2013, pp. 275–284.

[16] H. Huang, H. Sun, G. Ma, X. Wang, and X. Liu, "Poster : A Framework for Instant Mobile Web Browsing with Smart Prefetching and Caching," in *Proceedings of the 20th annual international conference on Mobile computing and networking*, 2014, pp. 367–369.

[17] P. K. Shah, A. Mitra, and D. Matani, "An O (1) algorithm for implementing the LFU cache eviction scheme," 2010.

[18] J. Yan, J. Chen, and W. Jiang, "Data Caching Techniques in Web Application," in *2014 Enterprise Systems Conference (ES)*, 2014, pp. 289–293.

[19] V. J. Sosa, G. González, and L. Navarro, "Building a Flexible Web Caching System," in *Computer Science, 2003. ENC 2003. Proceedings of the Fourth Mexican International Conference*, 2003.

[20] B. N. Thyamagondlu, V. W. Chu, and R. K. Wong, "A Bandwidth-Conscious Caching Scheme for Mobile Devices," in *2013 IEEE International Congress on Big Data*, 2013.

[21] W. Ali, S. M. Shamsuddin, and A. S. Ismail, "A survey of web caching and prefetching," *Int. J. Adv. Soft Comput. its Appl.*, vol. 3, no. 1, pp. 18–44, 2011.

[22] K. Ramu, R. Sugumar, and B. Shanmugasundaram, "A Study on Web Prefetching Techniques," *J. Adv. Comput. Res.*, vol. 1, no. 1, pp. 39–46, 2012.

[23] R. Chandrakar, "Web Page Prioritization For Fast Accessing - Using Web Log, Prefetching Technique And Markov Model," *Int. J. Comput. Technol. Appl.*, vol. 7, no. 3, pp. 465–476, 2016.

[24] V. N. Padmanabhan and J. C. Mogul, "Using predictive prefetching to improve World Wide Web latency," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 26, no. 3, pp. 22–36, 1996.

[25] A. Nanopoulos, D. Katsaros, and Y. Manolopoulos, "A data mining algorithm for generalized Web prefetching," *Knowl. Data Eng. IEEE Trans.*, vol. 15, no. 5, pp. 1155–1169, 2003.

[26] I. Zukerman, D. W. Albrecht, and A. E. Nicholson, "Predicting users' requests on the WWW," *UM '99 Proc. seventh Int. Conf. User Model.*, pp. 275–284, 1999.

[27] A. Venkataramani, P. Yalagandula, R. Kokku, S. Sharif, and M. Dahlin, "The potential costs and benefits of long term prefetching for content distribution," *Tech. Rep. TR-01-13, UT, Austin, 2001.*, no. March 2001, pp. 1–22, 2001.

[28] M. B. Pal and D. C. Jain, "Web Service Enhancement Using Web Pre-Fetching By Applying Markov Model," in *Communication Systems and Network Technologies (CSNT), 2014 Fourth International Conference on*, 2014, pp. 393–397.

[29] S. Manoharan and Y. Deng, "A Review of Web Cache Prefetching," *J. Inform. Commun. Converg. Eng.*, vol. 12, no. 3, pp. 161–167, 2014.

[30] H. Li, J. Zhang, H. Lv, and Q. Lin, "Web prefetching of smart

wireless access point," in *2016 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, 2016, pp. 211–216.
[31] C. Du and S. Wang, "Research on Mobile Web Cache Prefetching Technology Based on User Interest Degree," *Res. Mob. Web Cache Prefetching Technol. Based User Interes. Degree*, pp. 1253–1258, 2015.
[32] S. Zhang, Y. Lu, and X. Zhang, "Cache Replacement Improving Based on User Group Interest Degree," in *Trustworthy Computing and Services*, vol. 520, Springer Berlin Heidelberg, 2015, pp. 1–7.
[33] H. Chavan, S. Sane, and H. B. Kekre, "A Markov-Graph Cache Replacement Policy for Mobile Environment," in *Communication, Information & Computing Technology (ICCICT), 2012 International Conference on*, 2012, pp. 1–6.
[34] B. Priyansha and S. K. Nath, "A Survey on Web Pre-Fetching Techniques," vol. 5, no. 6, pp. 7012–7021, 2014.
[35] E. P. Markatos and C. E. Chronaki, "A top-10 approach to prefetching on the web," *Proc. INET*, vol. 98, pp. 276–290, 1998.
[36] T. Palpanas and A. Mendelzon, "Web prefetching using partial match prediction," 1998.
[37] C. Z. Xu and T. I. Ibrahim, "A keyword-based semantic prefetching approach in internet news services," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 5, pp. 601–611, 2004.
[38] C. Z. Xu and T. I. Ibrahim, "Towards semantics-based prefetching to reduce Web access latency," *Proc. - 2003 Symp. Appl. Internet, SAINT 2003*, pp. 318–325, 2003.
[39] S. W. Shin, B. H. Seong, and D. Park, "Improving World-Wide-Web performance using domain-top approach to prefetching," *Proc. - 4th Int. Conf. High Perform. Comput. Asia-Pacific Reg. HPC-Asia 2000*, vol. 2, pp. 738–746, 2000.
[40] D. Fisher and G. Saksena, "Link prefetching in Mozilla: A server-driven approach," *Web content caching Distrib.*, pp. 283–291, 2002.
[41] A. S. Nair and J. S. Jayasudha, "Dynamic Web pre-fetching technique for latency reduction," *Proc. - Int. Conf. Comput. Intell. Multimed. Appl. ICCIMA 2007*, vol. 4, pp. 202–206, 2008.
[42] Q. Yang and Z. Zhang, "Model based Predictive Prefetching," pp. 291–295, 2001.
[43] P. Cao and S. Irani, "Cost-aware WWW proxy caching algorithms," *Proc. USENIX Symp. Internet Technol. Syst. USENIX Symp. Internet Technol. Syst.*, no. December, p. 18, 1997.
[44] L. Cherkasova, "Improving www proxies performance with greedy-dual-size-frequency caching policy," 1998.
[45] X. Chen and X. Zhang, "Accurately Modeling Workload Interactions for Deploying Prefetching inWeb Servers *," *Proc. Int. Conf. Parallel Process.*, vol. 2003–Janua, pp. 427–435, 2003.
[46] T. Fagni, R. Perego, F. Silvestri, and S. Orlando, "Boosting the Performance of Web Search Engines: Caching and Prefetching Query Results by Exploiting Historical Usage Data.," *ACM Trans. Inf. Syst.*, vol. 24, no. 1, pp. 51–78, 2006.
[47] Z. Jiang and L. Kleinrock, "An adaptive network prefetch scheme," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 3, pp. 358–368, 1998.
[48] B. D. Davison, "Adaptive Web Prefetching," 1999.
[49] X. Dongshan and S. Junyi, "A new Markov model for web access prediction," pp. 34–39, 2002.
[50] X. Jin and H. Xu, "An approach to intelligent Web pre-fetching based on hidden Markov model," *Decis. Control. 2003. Proceedings. 42nd …*, vol. 3, no. December, p. 2954–2958 Vol.3, 2003.