Enhancing Web Surfing Experience in Conditions of Slow And Intermittent Internet Connection

Ľuboš Demovič, Martin Konôpka, Marek Láni, Matúš Tomlein[*]
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 3, 842 16 Bratislava, Slovakia
icup2012@gmail.com

Abstract

Despite of the advancements in information and telecommunication technologies, slow and intermittent Internet connection is still a serious issue in many places around the World, but mostly in developing countries. At the same time, Internet with its most popular service – the Web, became one of the very important parts of our everyday lives as more and more of human activity is taking place online. We believe that providing access to information on the Web is crucial for young people in developing countries to get the required skills and acquire experience in order to finally achieve significant progress in solving problems of their countries. In this paper, we propose a concept of software solution called OwNet which makes the Web surfing experience less frustrating even on slow and intermittent Internet connections. OwNet is based on using a local proxy server, which acts as an intelligent bridge between the client's browser application and the Internet and communicates with other clients and services in order to provide the best surfing experience. Although this concept is not bound to the quality of available connection, we mainly target the current situation in developing countries. The paper presents the overall concept and details on methods used for intelligent caching and prefetching of Web content.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*information filtering*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software; H.3.5 [**Information Storage and Retrieval**]: On-line Information Services—*data sharing*; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

Keywords

caching, collaborative tools, Internet, prefetching, proxy server, World Wide Web

[*]Bachelor degree study programme in field Informatics. Supervisor: Dr. Michal Barla, Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies, STU in Bratislava.
Work described in this paper was presented at the 8th Student Research Conference in Informatics and Information Technologies IIT.SRC 2012.

Demovič, Ľ., et al. Enhancing Web Surfing Experience in Conditions of Slow And Intermittent Internet Connection. Information Sciences and Technologies Bulletin of the ACM Slovakia, Special Section on Student Research in Informatics and Information Technologies, Vol. 4, No. 2 (2012) 25-29

1. Introduction

The Web became a mass media allowing its users to search for information. It provides a way to acquire knowledge from various study fields and allows people to stay in touch with the current situation in the World. Thanks to the Web, its users may express their thoughts and discuss their problems with people around the world without the need to know them personally or travel long distances.

Despite of the advancements in information and telecommunication technologies, slow and intermittent Internet connection is still a serious issue in many places around the World and is most visible in developing countries. Web surfers from these countries are extremely patient as there is not much they can do to deal with unexpected cut-offs, slow and incomplete downloads of webpages [6], all preventing the Web to become a really useful tool for dealing with everyday problems.

Even though the Internet connection is often shared among several users, it does not take into account repetitive accesses to the same resources. The same webpage or YouTube video is being downloaded again and again by different users even though it has not changed since last visit.

The browsing session is usually performed in subsequent peaks. The Internet link is idle most of the time when the user reads a downloaded webpage or during the night and is overloaded when the user decides to access another webpage. There are almost no "background" jobs taking advantage of slow and intermittent (but flat-rate) link [7].

Another aspect is that users surf on their own. However, a group of users, e.g., students in a computer class, share interests and information needs and could take advantage of collaborative surfing to achieve their goals more quickly and efficiently [10]. If a valuable resource is identified by a group member, others can be notified about it [9].

We propose a solution for improving utilization of slow and intermittent connection by addressing all these weak points in its usage.

2. Related work

Different software solutions have been proposed to utilize available Internet connection. The most common approach is to cache downloaded Web content on the *client side*, by the browser itself. User's requests for already cached objects are then resolved almost immediately without the need to use the available connection. Most Web browsers use the Least Recently Used algorithm to free space in cache if needed. It keeps track of all objects in cache and when there is a need for more free space, it deletes the least recently used one [11].

Another approach to minimize latency is by application of a prefetching algorithm on proxy server which client Web browsers connect to. The algorithm predicts users' next requests from currently visited webpage and advises proxy server to cache all the required objects before the real requests come. The prediction is based on users' browsing history, e.g., in the Refererrer Graph algorithm [4].

Caching and prefetching algorithms may also be implemented on the *server side*. Resolving requests of multiple clients on the same local proxy server with a larger dataset enables to minimize latency more than if it was done just locally [8].

3. OwNet Concept Overview

The concept of our OwNet solution is aimed at utilizing the available Internet connection by introducing an intermediary between the client applications (e.g., a browser) and the connection. This intermediary operates upon requests from client applications and responses from Web servers in order to deliver required functionality and experience. More precisely, it provides:

- *advanced caching mechanism* which prevents downloading the same content multiple times,
- *intelligent prefetching* to pro-actively download webpages of user's interest (user's next step) during the idle time resulting in perception of having a faster Internet connection,
- *collaborative tools* such as explicit webpages ratings and recommendations by and for a micro-community (e.g., students of same class).

The concept is implemented as a hierarchy of three modules communicating with each other using client-server architecture. These modules are (as shown on Figure 1): local client proxy application, local server proxy application, central service.

Local client proxy application is responsible for handling requests from client applications, e.g., Web browser. Locating the client proxy on a user's computer enables manipulation of results and resolving his or her requests almost immediately, as caching and prefetching can be applied. Local client proxy application also provides intranet portal to access collaborative tools and other functionalities, such as search within cached webpages. Users use their Web browser to access the Web in the same way they would without using our solution.

Local client proxy applications connect to a **local server proxy application**, preferably within a same organization, e.g., in a school. This enables caching of Web content within an organization with a local-network-only access to this cache. Local server proxy gathers browsing history from multiple users and enables each client to prefetch visited objects which were not visited yet by this client. Connecting clients to the local server also brings advantages of collaborative tools.

The last module in proposed hierarchy is the **central service** which is accessed by local servers over the Internet, mainly for recommendations on which cache objects to update. Central service is connected using a fast Internet connection and is also faster in terms of computing performance. In addition to cache update information, it is also capable of routing local servers to enable communication between them. For instance, physically distanced schools may share information sources, or government agencies may distribute variety of surveys to local servers and thus reach individual users. If the local client proxy application is not connected to a local server, it can connect directly to the central service.

4. Web Content Caching in OwNet

Caching of Web content is one of the most effective ways for minimizing network traffic. It allows us to save bandwidth, reduce latency and, if implemented locally, provide a way to access cached objects offline. In our project we mostly deal with local or client-side cache, that can be accessed without Internet connection.

One of the main issues when using a cache is to determine when a specific cache entry should be invalidated and/or refreshed by re-downloading it from the Web. Another issue is that cache cannot be left to grow endlessly as it could easily take over all the free disk space.

4.1 Caching Algorithm

We have come up with an algorithm that periodically ranks cached objects by their cache hits and removes the ones with the lowest rank. The interval should be set depending on the expected traffic. Each cache access is uniquely identified with a number according to the sequence of accesses. Cached objects store the identifier of their first access and also a counter of how many times they were accessed. The global identifier of the last cache access (L), the identifier of the first access of the cached object (C_F) and the counter of accesses of the cached object (C_C) are used to calculate the rank (C_R) for each cached object using formula (1).

$$C_R = \frac{C_C}{L - C_F + 1} \tag{1}$$

This rank can be calculated for each cached object in a single database query. If we order the results of this query by their rank, we can easily filter out a required number of cached objects to gain enough of free space.

To assign smaller ranks to new cached objects, the counter of cache accesses in formula (1) can be replaced by a number incremented adaptively after each visit based on the count of previous accesses by a value between 0 and 1. Thus, new cached objects would have a smaller advantage compared to older cached objects.

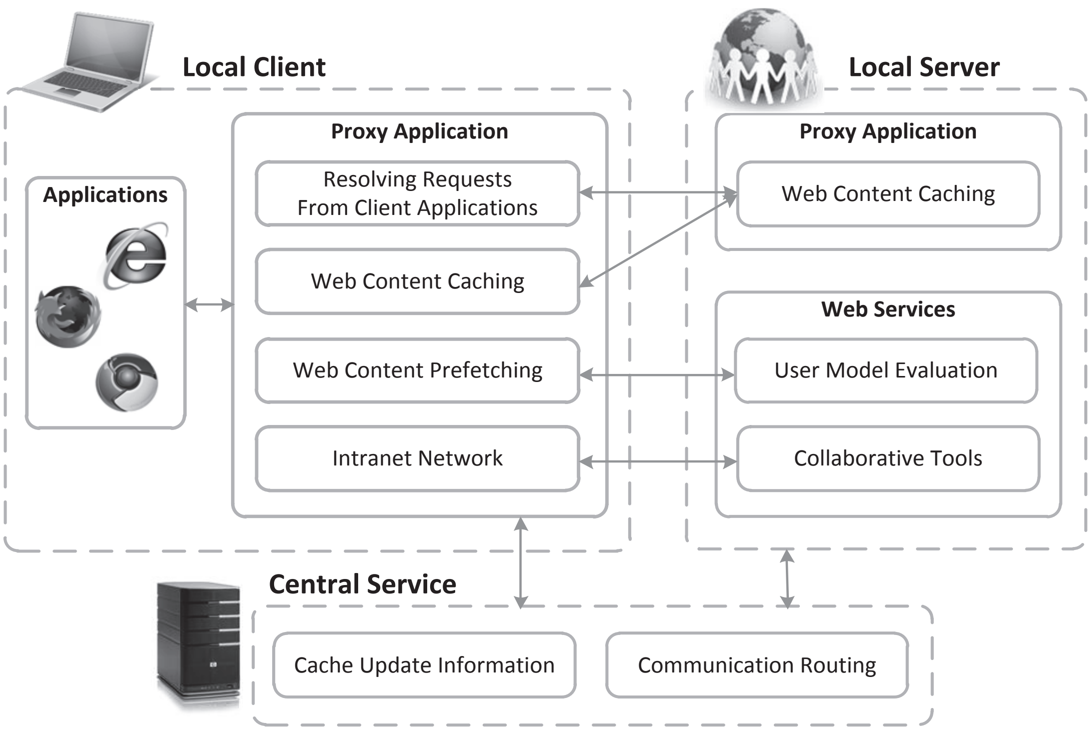

Figure 1: OwNet modules and their functionalities.

4.1.1 Evaluation

We compared the efficiency of our algorithm with the Least Recently Used algorithm (LRU) and Greedy-Dual-Size-Frequency algorithm (GDSF) [2, 3]. In the simulation, we used web traces from the University of California, Berkeley [1], containing 2,599,049 requests from 6,037 unique clients with a total response size of 15,661,492,343 bytes.

The results can be seen in Table 1.

Our algorithm produced slightly better cache hit ratio and cache byte hit ratio than the LRU algorithm. An advantage of our algorithm over the LRU algorithm is that periodically accessed cached objects are kept longer, as LRU prefers recently used cached objects. Neither our nor the LRU algorithm take the size of cached objects into consideration.

We also evaluated an implementation of the GDSF algorithm, which is an algorithm designed for Web proxies. It ranks objects in cache by their size, number of accesses and time of their last access. The evaluation resulted in cache hit and cache byte hit ratios that were superior to our algorithm and the LRU algorithm.

Based on this evaluation, we decided to implement the GDSF algorithm in our solution.

4.2 Updating Cache

Cache invalidation and checking for changes can be quite demanding for computing power and the connection bandwidth. There is no way to tell at which point a certain cached object should be refreshed as websites on the Web do not follow any rules and change their content at any time.

[1]Source: http://ita.ee.lbl.gov/html/contrib/UCB.home-IP-HTTP.html [accessed 2012-03-16]

Table 1: Evaluation of cache algorithms based on cache hits and cache byte hits. We also give results for a cache size that didn't require any objects to be deleted (referred to as Unlimited).

Algorithm	Cache hit ratio	Cache byte hit ratio
Unlimited	67,88%	66,17%
LRU	57,3%	56,85%
OwNet	59,81%	58,95%
GDSF	62,13%	59,37%

Cached objects are stored on local client and also on local server proxy applications, which are responsible for updating the cache. To determine which cached objects should be updated, the local server proxy communicates with the central service.

Local servers aggregate cache access logs from their client proxy applications and report them to the central service. Cache access logs contain information about new or updated cached objects and accesses to these cached objects.

When a report is received on the central service, new and updated objects are downloaded right away to get approximately the same version as the reporting server has. The central service counts accesses to cached objects and when the latest version of a cached object is accessed more than the predefined minimum number of times, an update is planned on the central service.

Cached objects on the central service have a calculated update interval that needs to pass between their last and next update. The interval is calculated depending on whether there were any changes in previous cache updates. The current implementation looks at the last three updates to cached objects and their update intervals to calculate the next update interval.

Independently of receiving reports from local servers, the central service downloads planned updates, compares them to their previous versions and if there is a change, it creates an update recommendation for local server. Local servers periodically ask the central service whether there are any new recommended updates. If so, they keep the information and download the updates in time when their Internet connection is least used.

When a cached object on a local server is updated, there still might be outdated versions of the cached object on connected client proxy applications. To get rid of those, the client proxy applications periodically ask the local server for a list of updated cached objects, check if they have older versions of the objects and if they do, they remove them.

5. Web Content Prefetching in OwNet

Prefetching the Web content before it is requested may induce latency reduction by more than 50% [8].

5.1 User Model

We have based our prefetching method on the Referrer Graph algorithm [4] which builds a graph from recorded browsing history. The algorithm considers the complexity of the Web content (i.e., webpages are not plain text files only but rather a collection of texts, embedded media, linked scripts, etc.) by differentiating between objects requested by user and objects required by Web browser to complete user's requests. In OwNet, we use the prediction method of user's next steps from the Referrer Graph algorithm, but we have decided to simplify the graph and not to keep track of every requested Web object.

The graph contains nodes that represent webpages requested and accessed by user only. The nodes are valued by number of user's visits and edges with number of user's traverses of hyperlinks. The graph with browsing history is gathered on the local server proxy application, so we may distinguish between users and model their history individually. Moreover, we take into account information about particular webpages given explicitly by users via collaborative tools. Explicit ratings of webpages increase their value within an evaluation made by our prefetching algorithm. Webpages which have not been periodically visited, but have been positively rated by multiple users, may become more important than regularly visited webpages with no or lower rating.

It is important to note that when working with single user's browsing history only, existing prefetching algorithms are able to prefetch any object only if it has already been captured in this history. However, by gathering browsing history from multiple users prefetching algorithm may also provide content that a user has not yet visited. We have applied neighbour-based collaborative filtering method [5] to find similar users to enhance user's history by (not yet included) webpages visited by other users. The prefetching algorithm is then able to prefetch new Web content because it is included in the user's history.

5.2 Prediction Algorithm

When user accesses webpage which is already stored in the graph, we predict user's next steps using formula (2). Prediction t_i of user's traverse to each next webpage i, accessible from the current webpage, is evaluated by combining user's rating for it t_i^r and transition confidence t_i^f (evaluated by the Referrer Graph algorithm), α constant is set to 0.6.

$$t_i = \alpha t_i^r + (1 - \alpha) t_i^f \tag{2}$$

5.3 Prefetching Using a Simulation of Browsing

Local client proxy application receives predictions of user's next steps from the local server (see Figure 1) and downloads them when the available Internet connection is not currently used by user. The webpages link to many embedded objects and because we do not keep track of them in the graph, prefetching would not induce the expected latency reduction. That is why we simulate user's access to the predicted webpages by opening them in hidden Web browser. The webpage is downloaded in the same way as the user would do it but without user's control over the download process. Local client proxy application receives requests for embedded objects and downloads them to the cache. The prefetched webpage is completely available in the cache and thus accessible without using the Internet connection.

5.4 Complexity Measure

We analysed space complexity of the graph we use for prefetching in OwNet and compared it to two other existing solutions – Referrer Graph and Prediction By Partial Match algorithms [1].

The results can be seen in Table 2.

Table 2: Space complexity measure and comparison of the graphs used for prefetching in OwNet, Referrer Graph (RG) and Prediction By Partial Match (PPM) algorithms.

Data	OwNet	RG	PPM
Users	$O(p)$	1	1
Nodes	$O(n)$	$O(h + v)$	$O(n^k)$
Node Visits	$O(pn)$	$O(h + v)$	$O(n^k)$
Edges	$O(n^2)$	$O(hv + h^2)$	$O(nk)$
Edge Traverses	$O(pn^2)$	$O(hv + h^2)$	$O(nk)$

Our graph keeps track of each user's visits of webpages and traverses between them, but it does not distinguish between object types like the Referrer Graph algorithm does (h and v, $n \leq h$). The Prediction By Partial Match algorithm distinguishes between browsing sessions what leads to its exponential space complexity.

6. Conclusion and Future Work

In this paper we proposed OwNet – solution for enhancing Web surfing experience, mainly in conditions of slow and intermittent Internet connection.

We employ our own caching algorithm to save bandwidth and reduce latency when fetching requested Web objects. Being aware that our application will be used on older computers, we tried to optimize it to be light, simple and efficient. We also proposed a combination of existing prefetching approaches to get an algorithms tailored to our needs.

We also took indirect approach by applying collaborative tools in our solution to solve the problem when users of the same group browse individually. By recommending and rating visited webpages they can direct others to the result they were searching for.

OwNet solution consists of three modules which are individually independent but they complement one another. For instance, employing the central service as a means to find out which cached objects are outdated, reduces the load on local proxy servers and their Internet connection. It ensures that cached objects are updated only if they were changed on the Internet. On the other side, using a single central service by multiple local server can lead to a bottleneck effect. This issue can be solved by using multiple central services in places with a good Internet connection or a high-performance cloud computing platform, e.g., Windows Azure or Amazon Web Services.

We contacted few Slovak NGOs that operate in rural areas of Africa to discuss possibilities of OwNet deployment in these places. Our offer was warmly welcomed and we are currently in progress of deploying OwNet to computer lab in Nanyuki High Shool in Kenya. In addition, we have established a partnership with Institute of Information and Prognoses of Education in Slovakia in order to propose OwNet also to Slovak schools, which can also take advantage of its caching and collaborative features.

Acknowledgements. This work was partially supported by the Slovak Research and Development Agency under the contract No. APVV-0208-10.

References

[1] Z. Ban, Z. Gu, and Y. Jin. An Online PPM Prediction Model for Web Prefetching. In *Proc. of the 9th annual ACM int. workshop on Web Information and Data Management*, WIDM '07, pages 89–96, New York, NY, USA, 2007. ACM.

[2] L. Cherkasova. Improving WWW Proxies Performance with Greedy-Dual-Size-Frequency Caching Policy. *Computer*, (HPL-98-69 (R.1)), 1998.

[3] L. Cherkasova and G. Ciardo. Role of Aging, Frequency, and Size in Web Cache Replacement Policies. In *Proc. of the 9th int. conf. on High-Performance Computing and Networking*, HPCN Europe 2001, pages 114–123, London, UK, UK, 2001. Springer-Verlag.

[4] B. de la Ossa, A. Pont, J. Sahuquillo, and J. A. Gil. Referrer Graph: a Low-cost Web Prediction Algorithm. In *Proc. of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 831–838, New York, NY, USA, 2010. ACM.

[5] C. Desrosiers and G. Karypis. A Comprehensive Survey of Neighborhood-based Recommendation Methods. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 107–144. Springer US, 2011.

[6] B. Du, M. Demmer, and E. Brewer. Analysis of WWW Traffic in Cambodia and Ghana. In *Proc. of the 15th int. conf. on World Wide Web*, WWW '06, pp 771–780, New York, USA, 2006. ACM.

[7] D. L. Johnson, V. Pejovic, E. M. Belding, and G. van Stam. Traffic Characterization and Internet Usage in Rural Africa. In *Proc. of the 20th int. conf. companion on World Wide Web*, WWW '11, pages 493–502, New York, NY, USA, 2011. ACM.

[8] T. M. Kroeger, D. D. E. Long, and J. C. Mogul. Exploring the Bounds of Web Latency Reduction from Caching and Prefetching. In *Proc. of the USENIX Symposium on Internet Technologies and Systems on USENIX Symposium on Internet Technologies and Systems*, Berkeley, CA, USA, 1997. USENIX Association.

[9] M. R. Morris. A Survey of Collaborative Web Search Practices. In *Proc. of the twenty-sixth annual SIGCHI conf. on Human Factors in Computing Systems*, CHI '08, pages 1657–1660, New York, NY, USA, 2008. ACM.

[10] M. Šimko. Automated Domain Model Creation for Adaptive Social Educational Environments. In *Information Sciences and Technologies Bulletin of the ACM Slovakia*, Vol. 3, No. 2, 2011.

[11] A. S. Tanenbaum. *Modern Operating Systems*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2007.