Caching with Relation

Mohamad Nour Moazzen, Stefan Liebald *
*Chair of Network Architectures and Services, Department of Informatics
Technical University of Munich, Germany
Email: mohamadnour.moazzen@tum.de, liebald@net.in.tum.de

Abstract—
The increase in network traffic across the Internet in recent days impose many challenges to web servers and Internet service providers regarding access latency and network bandwidth consumption. Web caching is one of the optimization methods used to face these challenges. The most important part of a web caching system is the cache replacement algorithm which decides which web objects should be evicted from the cache in order to make space for new ones when the cache is full. Traditional algorithms consider metrics such as recency and frequency to make the replacement decisions. In this paper we explore a type of algorithms which considers the semantics of the web objects to make the replacement decisions. We differentiate two categories of this type of algorithms according to how they interpret the contents of the web objects: subject-based and link-based. We present some algorithms of both categories and explain how they work.

Index Terms—**WWW, Web caching, Cache replacement algorithms, Semantic distance**

1. Introduction

The Internet is a valuable source of information and it is used by an increasing number of people for education, entertainment, business and almost every aspect of modern life. This led to an increase of network traffic across the Internet, which in turn causes increasing access latency. To face these challenges, several optimization methods have been developed [1]. Web caching is one of these methods. It aims to improve the performance of web servers and decrease access latency.

We can differentiate three types of Web caching depending on the place where the web cache is employed: Client-side caching, Server-side caching and Proxy caching. In this paper we focus on proxy caching.

The basic idea of web caching is to store copies of web objects (web pages, photos, videos, etc.) which are requested by users in intermediate web caches, so future requests for these web objects will be served by these web caches instead of the original servers. This decreases the number of requests to the original servers and reduces the amount of transmitted data over the network in addition to decrease access latency because these web caches are closer to the users.

Web caching system works as follows:

- When user requests an object from a web server, the system first checks if this object is already cached.
- If there is a copy of this object in the cache it will be send directly to the user.
- If the object is not cached then it will be requested from the original web server and forwarded to the user, then this object is stored in the cache.

Because web caches have limited size, we cannot store every web object for indefinite time. When the cache is full, and new objects need to be stored in it, the system has to remove one or more cached objects in order to make space for the new objects. The decision of what objects to be removed from the cache is taken by the *cache replacement algorithm.*

Cache replacement algorithms are very crucial for the performance of the web caching system because if the algorithm removes objects which maybe requested again in the near future, this degrades the performance of the system as these objects have to be requested again from the original server. There are several types of cache replacement algorithms, each one of them depends on one or several metrics to decide which object(s) to evict from the cache.

In this paper we explore a type which depends mainly on the semantics of the contents of the web objects to make the replacement decisions. Algorithms of this type measure the relationship between the cached objects and the new incoming objects which need to be stored in cache in terms of a semantic distance calculated based on the subject of the contents or the links between objects. The objects which are the furthest from the new objects in terms of the semantic distance , i. e., the objects which are the least related to the new objects, are marked for eviction.

The rest of this paper is structured as follows. Section 2 includes a background about cache replacement algorithms. Section 3 introduces semantic cache replacement algorithms and describes several algorithms from this type. In section 4 we conclude the paper.

2. Background

Many cache replacement Algorithms have been proposed in literature.

We can differentiate between them according to the metric they use to make the replacement decisions.

Two of the most important metrics are [2]:

- Recency: time of last request for an object.
- Frequency: number of requests for an object.

Podlipnig and Böszörmenyi in [2] proposed a classification for the cache replacement Algorithms as follows:

- Recency-based Algorithms: depend on the recency metric to make the replacement decisions. The most well-known algorithm from this category is LRU (Least Recently used), which evicts the least recently accessed objects from the cache.
- Frequency-based Algorithms: depend on the frequency metric to make the replacement decisions. The most well-known algorithm from this category is LFU (Least Frequently used), which evicts the least frequently accessed objects form the cache.
- Recency/Frequency-based Algorithms: depend on both recency and frequency metrics to make the replacement decisions. SLRU (Segmented LRU) [3] is an example of algorithms from this category.
- Function-based Algorithms: use a function to calculate the value of an object then use this value as a metric to make the replacement decisions. GD(Greedy Dual)-Size [4] is an example of algorithms from this category.
- Randomized Algorithms: these algorithms randomly make the replacement decisions. HARMONIC [5] is an example of algorithms from this category.

The most important metrics to take into account when evaluating or comparing the performance of cache replacement Algorithms are hit rate and byte hit rate.

- Hit rate: "measures the percentage of requests that are served from the cache (i. e., requests for pages that are cached)" [6].
- Byte hit rate: "measures the amount of data (in bytes) served from the cache as a percentage of the total amount of bytes requested" [6].

The algorithms which we have mentioned in this Section depend on the metadata of the cached objects to decide which ones to evict from the cache (time of last access, frequency of access, etc.).

In the next section we explore a different type of cache replacement algorithms which take into account the contents of the objects [7] when making the replacement decisions.

3. Semantics based cache replacement algorithms

Semantics based cache replacement algorithms depend mainly on semantics of the contents of the cached web objects in order to decide which ones to evict from the cache [8]. They measure the relationship between the cached objects and the new incoming objects which need to be stored in cache (or the most recently accessed cached objects).

When the replacement process is triggered, the algorithm evicts the objects in cache which are less related to the new incoming objects regarding the semantics of their contents. This type of algorithms relies on the intuition that the cached objects which are less related to the new objects regarding the semantics of their contents are less likely to be requested in the near future, therefore they can be evicted from the cache.

If the algorithm cannot decide what objects to evict because all of them are closely related to the new incoming objects, then these algorithms make use of other metrics such as recency, frequency, etc., to make the replacement decision. We can differentiate two categories of this type of algorithms according to how they interpret the contents of the web objects:

- Algorithms which calculate the semantics for the objects based on the subject of their contents (LSR [7], LSR/H [8]).
- Algorithms which calculate the semantics for the objects based on the links which are contained in these objects (SACS) [6].

In the following we describe several semantics-based algorithms.

3.1. LSR

Alcides [7] proposed an algorithm called LSR (Least Semantically Related). This algorithm relies on the assumption that every user is inclined, for a period of time, to request objects whose contents belong to a specific subject, i. e., semantically related objects. Also, it assumes that during a period of time all the objects which are requested from a cache belong to the same subject, that is, "LSR supports single thread of interest" [8].

It depends on a metric called semantic distance to make the replacement decisions. In order to calculate the semantic distance, the algorithm associates semantics to each object according to its contents.

When the replacement process is triggered, the algorithm calculates the semantic distance between each cached object and the new object(s) which need(s) to be stored in the cache. The cached objects which are less related to the new objects, i. e., the objects which have the highest semantic distance to the new objects, are marked for eviction.

According to [7], one way to associate semantics to each object is through a taxonomy, objects are organized into a tree of subjects. It starts from the root and branches out into nodes where each node represents a subject, the nodes may branch out into children nodes which represents more specific subjects, objects are distributed on the nodes according to the subject of their contents.

Using this taxonomy, we can get the semantics of any object in the form of a sequence of tree nodes from root to the specific subject which match the subject of its contents. Then the algorithm can calculate the semantic distance between any two objects by measuring the shortest path between the corresponding subject nodes.

It is hard to implement such taxonomy for the whole Internet, but there are several efforts toward this like DMOZ Open Directory Project [9] and by Schmidt et al. in [10] which proposed creating web servers which can be queried using the URL of an object to respond by its semantics.

Figure 1 shows an example of a tree of subjects, it is a real-world example represents a partial view of a taxonomy defined by the DMOZ Open Directory Project [8]. Using this figure, we can get the semantics of the object which is titled "Foundations of the Internet Protocol" in the following form: *Top.Computers.Internet.Protocols.IP*. So, for example if this object is the new object which

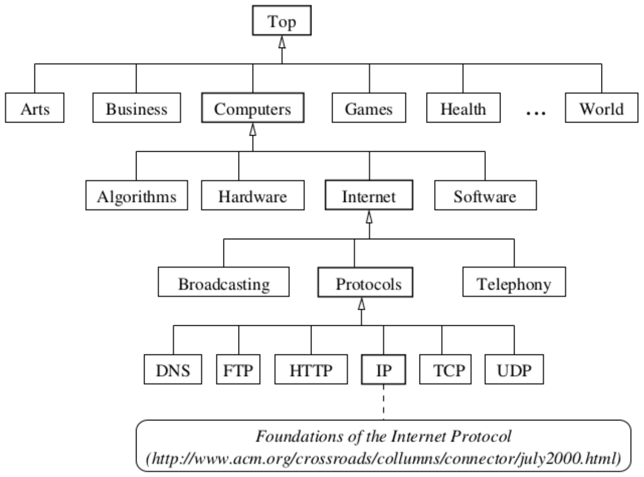

Figure 1: Partial hierarchy of subjects defined by the DMOZ Open Directory [8].

subject semantics	weight
Top.Computers.Algorithms	10
Top.Computers.Software	9
Top.Business	8
Top.Games	7
Top.Computers.Algorithms	6
Top.Computers.Internet.Protocols.IP	5
Top.Computers.Internet.Protocols.HTTP	4
Top.Games	3
Top.Business	2
Top.Computers.Internet.Protocols	1

Figure 2: A history of 10 subject accesses [8].

needs to be stored in the cache, the algorithm starts evicting objects which corresponds to subject nodes which are the furthest from *IP* node like *Art*, *Business*, *Algorithms*, etc.

3.2. LSR/H

Calsavara and Schuck [8] proposed an algorithm called LSR/H (Least Semantically Related + History of subject accesses) This algorithm is a developed version of LSR. It adds the dependency on a metric called history of subject accesses to make the replacement decision. It supports "multiple threads of interest" [8], i. e., there can be, for a period of time, multiple subjects of interest for the users. These subjects are called "hot subjects" [8]. A subject is included in the hot subjects when, for a period of time, there is a substantial number of accesses to objects whose contents are related to that subject [8].

To keep track of the hot subjects, the algorithm records the history of subject accesses. Whenever an object is accessed, it adds a weight to its corresponding subject. Recency plays a role in the weighting process, recent accesses weight more than old accesses [8]. The cached objects whose contents are less related to the hot subjects are marked for eviction, i. e., objects whose contents are related to subjects of less weight.

LSR/H, like LSR, depends on "tree of subjects" taxonomy to map semantics to objects. As we described

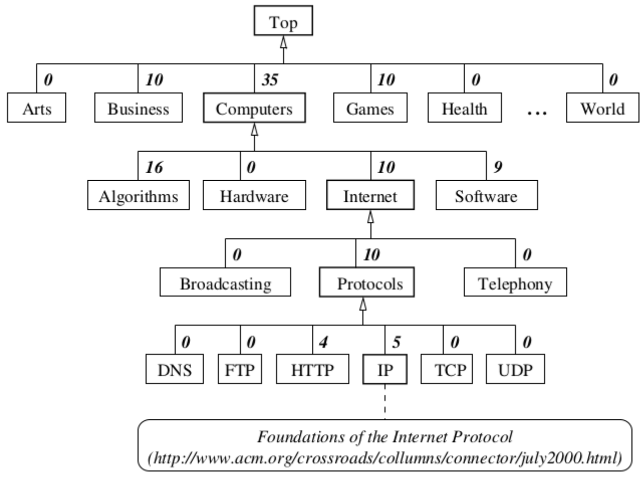

Figure 3: Hierarchy of subjects of Figure 1 with weights [8].

earlier, each request or access to an object adds a weight to its corresponding subject. In this taxonomy, when the algorithm adds a weight to a subject node, it should also add this weight to the parent nodes of this subject node recursively until reaching the root.

To clarify this, we take the tree of subjects in Figure 1 as an example. We assume that the record of history of subject accesses is of length 10, i. e., the algorithm is configured to record the subjects of the last 10 accesses. Also, we assume that when the replacement process is triggered the history of subject accesses is as shown in Figure 2.

The weight values are distributed according to the recency of access, i. e., 10 corresponds to the most recently accessed subject and 1 corresponds to the least recently accessed subject [8]. Using these values, the algorithm calculates the weights of the subjects.

For example: according to Figure 2, the weight of *Software* node is 9, the algorithm adds 9 to the node *Software* and also adds 9 to its parent node *Computers*. Figure 3 shows the tree of subjects after calculating the weights. We can see from Figure 3 that the hot subjects are mainly *Computers*, *Business* and *Games*. The algorithm starts evicting objects whose contents are related to the subjects of *Art*, *Health* and *World*, then *Business* and *Games* and so on.

3.3. SACS

André et al. [6] proposed an algorithm called SACS (Semantic Aware Caching System). This algorithm relies on the assumption that the cached objects which can be reached by links from a recently accessed object will most likely be requested in the near future, so, they should not be evicted from the cache. It depends on 3 metrics: recency, frequency and semantic distance to make the replacement decisions. The semantic distance between two objects is calculated by measuring the minimum number of links which is needed to be followed to reach one from the other [6]. SACS differentiates between two types of links when calculating the semantic distance:

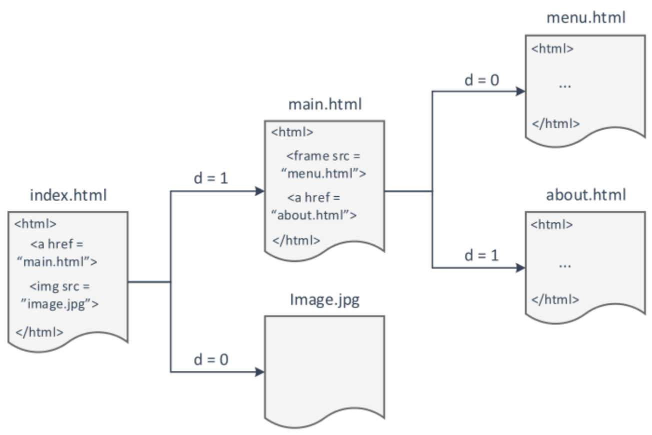

Figure 4: Example of a web site with distance assigned based on link information [6].

- Explicit link: the link that should be clicked by the user to get the referenced object. A link of this type is assigned a distance of 1 [6].
- Implicit link: the referenced object by this link is loaded automatically. A link of this type is assigned a distance of 0 [6].

Figure 4 shows an example of how SACS assigns the distance between pages of a website. SACS continuously keep track of a set of objects called pivot, they are the most recently accessed cached objects [6]. When the replacement process is triggered, the algorithm calculates the semantic distance between each cached object and the nearest member of pivot. The objects which are the most distant from pivot are marked for eviction. In case multiple objects are in the same distance from pivot, the algorithm orders them according to their frequency information (the objects which are less frequently requested are more likely to be evicted).

4. Conclusion

Web caching is one of the optimization methods used by the web servers to improve their performance and decrease access latency.

Cache replacement algorithm decides the web objects that should be evicted from the cache in order to make space for new ones when the cache is full.

Traditional algorithms consider metrics such as recency and frequency to make the replacement decisions.

In this paper we explored a type of cache replacement algorithms which considers the semantics of the contents of the web objects to make the replacement decisions. We divided it into two categories according to how they interpret the contents of the web objects: subject-based and link-based. Then we described several algorithms of this type.

According to [6], [7], [8] these algorithms perform better than the other well-known replacement algorithms such as LRU and LFU.

LSR/H builds upon LSR by adding additional metrics to improve performance.

LSR and LSR/H are hard to be implemented on the Internet scale because they need a global representation of the semantics of web objects (based on the subjects of their contents) which enables them to associate semantics to objects. This is not available for the all the objects on the Internet, though there are some efforts toward that like DMOZ open directory project [9] and by Schmidt et al. in [10] which proposed creating web servers which can be queried using the URL of an object to respond by its semantics.

SACS algorithm on the other hand does not require such condition to be implemented.

Also, LSR and LSR/H consider that each web object can be associated with only one specific subject, therefore, these algorithms are not applicable in cases where we have for example a web page or an article which discusses several subjects.

References

[1] Y. Zhang, N. Ansari, M. Wu, H. Yu, On wide area network optimization, Commun Surv Tutor IEEE 14(4):1090-1113, (2012).

[2] S. Podlipnig, L. Böszörmenyi, A survey of Web cache replacement strategies, ACM Comput. Surv. 35, 4, 374-398, (2003).

[3] R. Karedla et al, Caching Strategies to Improve Disk System Performance, Computer, vol. 27, no. 3, pp. 38-46, (1994).

[4] P. Cao, S. Irani, Cost Aware WWW Proxy Caching Algorithms, Proceedings of USENIX Symposium on Internet Technologies and Systems (USITS), Monterey, CA, pp. 193-206, (1997).

[5] S. Hosseini-Khayat, Investigation of generalized caching, Ph.D. dissertation, Washington University, St. Louis, MO, (1997).

[6] N. P. André, R. Carlos, F. Paulo, V. Luis, An adaptive semantics-aware replacement algorithm for web caching. Journal of Internet Services and Applications, 6. 10. 1186/s13174-015-0018-4, (2015).

[7] C. Alcides, The least semantically related cache replacement algorithm, In Proceedings of the 2003 IFIP/ACM Latin America conference on Towards a Latin American agenda for network research (LANC '03), ACM, New York, NY, USA, 21-34, (2003).

[8] A. Calsavara, M. R. Schuck, Internet object caching based on semantics and access history, Programa de Pos-Graduacao em Informatica Aplicada Pontificia Universidade Catolica do Parana Rua Imaculada Conceicao, 1155, Prado Velho 80215-901 Curitiba, PR.

[9] AOL Inc. "Archive of dmoz.org provided by Internet Marketing Ninjas", Last visited 2019-02-06, http://dmoz-odp.org/

[10] A. Calsavara, G. Schmidt, Semantic search engines, In F. Ramos, F. Unger and V. Larios, editors, Advanced Distributed Systems: Third International School and Symposium, ISSADS 2004, Guadalajar a, Mexico, January 24-30, 2004, Revised Selected Papers, volume 3061 of Lecture Notes in Computer Science, pp 145-157. (2004).