Sana'a University Journal of Applied Sciences and Technology ISSN : 2958-9568
Vol. 1| No. 2 | Page 217 – 224 | 2023 | https://journals.su.edu.ye/index.php/jast

Web Caching Hybrid Algorithm by Semantic Similarity

Abdualmajed Ahmed G. Al-Khulaidi[1,*] , Mohammed Abdulbaqi Mohammed Ali Al-Sayani[2]
[1]Software Engineering, Faculty of Computer and Information Technology, Sana'a University, Sana'a, Yemen.
[2] Computing in Software Engineering, University of Science and Technology.
***Corresponding author**:*alkhulaidi@su.edu.ye*

ARTICLE INFO
Article history:
Received: March 20, 2023
Accepted: April 2, 2023
Published: April 30 2023

Keywords
1. NGD
2. GDFS
3. Hybrid
4. Cache

ABSTRACT: Recently, most companies have been conducting their business through the cloud network, so increasing the speed of data access via the Internet has become very important. Therefore, the importance of cache algorithms lies in increasing the speed of data access. As we know, there are cache substitution algorithms that work well with the cache when applied to the processor cache, but they hardly work when applied to caching for web purposes. The reason for this discrepancy is that it is not intended to increase the complexity of this type of storage. Considering the challenges of cache usage in terms of the large discrepancy in file size and the heterogeneous model of user access to data in a web environment, especially with pages with dynamic content, there is a real need to develop 'web cache' algorithms. In this paper, a hybrid algorithm for web caching using semantic similarity is developed. The GDFS algorithm was developed using NGD (Normalized Google Distance) to determine the semantic similarity between cache objects, which resulted in better performance in comparison to other algorithms. The results showed that the web cache hybrid algorithm using semantic similarity increased the hit rate compared to the basic algorithms by up to 80.10%. The proposed hybrid algorithm was able to overcome the problem of the low byte hit rate in the GDFS algorithm. The improvement in the byte hit rate reached 65%. This indicates an increase in the byte hit rate. The results showed that the web cache hybrid algorithm using semantic similarity reduced the page load time using distance measured from Google compared to the page load time using other algorithms that do not use semantic similarity and do not use cache. The results showed that the web cache hybrid algorithm using semantic similarity reduced the page load time from 4.17 seconds using GDFS-Line to 2.16 seconds using GDFS-NGD.

CONTENTS

1. Introduction

The retrieval of information through information retrieval systems as an abstract and unguided process is no longer feasible in light of the challenges facing these systems in terms of increasing size. The stored information and its diversity, the demand for or fast admittance to proper data, and the accessibility of assets Information within the range of recipients require the utilization of a bunch of apparatuses and strategies. That will coordinate electronic substances and their administration to improve the viability of the way toward recuperating information from them. All things considered, in the early beginnings of computer information retrieval systems, the system's ability to retrieve information was its primary and most important goal. However, after a short period of time, it turns out that information retrieval is an abstract and unguided process that cannot be the sole target of the system. Generally, there are three levels through which and on the basis of which information retrieval systems can be evaluated [1, 15, 17, 23, 25, 30].

• The first level: Evaluation of the effectiveness of the system as it relates to considerations of user-party satisfaction.

• The second level: Cost-effectiveness evaluation, which relates to user satisfaction with the internal efficiency of the system.

• The third level: Evaluating the cost return related to the relevance of the system versus its operating costs.

Many of these goals can be achieved through "web caching", and caching for the purpose of the web is a technology widely used in the various environments in which operations are performed as data access. Cache technology works by storing data from the slowest operations in memory faster but relatively smaller, so the access time to the cached data is significantly reduced. Thus, if the appropriate choice is made for the data to be stored, then subsequent access to this data will be done too quickly [4].

2. Existing Algorithms To Increase The Speed Of Data Access

2.1 GDFS (Greedy Dual Frequency Size)

The Greedy-Dual-Size-Frequency caching policy is used to maximize hit and byte hit rates for WWW proxies. The proposed caching strategy incorporates in a simple way the most important characteristics of the file and its accesses, such as file size, file access frequency, and the recentness of the last access. Greedy-Dual-Size-Frequency is an improvement of the Greedy-Dual-Size algorithm [1, 7, 8].

2.2 NGD (Normalized Google Distance)

The number of visits returned by Google for a given set of keywords is used to compute the Google Natural Distance (NGD) measure, which is a semantic similarity metric. If keywords have many pages in common with respect to their respective independent frequencies, those keywords are thought to be semantically similar [11, 12, 13].

3. Proposed Hybrid Cache Algorithm

3.1 Processing Objects to be Used with Improved Cache Algorithms:

1. The most important stages of preparing the purposes for applying the measures of semantic similarity to them, which were applied in the research, are [2, 3, 4, 6, 24]:

Preprocessing is the stage of preliminary processing of the documents to obtain a value that can be handled by semantic similarity.

At this stage, the following is done:

• Standardization of cases.

• Delete words in a language other than English.

• Delete special characters.

• Delete numbers.

2. Tokenization is the processing stage of cutting the entire sequence of letters into single syllables of words. The text in the English

language is divided according to which the space.

3. Stop word Removal is to delete words that have no meaning, such as letters, prepositions, and words such as AS, His, The, Of, And, To, A, In, That, Is, Was, He, For, It, With, On, Be, At, By, I, etc.

4. Return the word to the root by removing suffixes and sometimes prefixes. For example, words like "Cats" "Catlike", and "Catty" returned to the word "Cat".

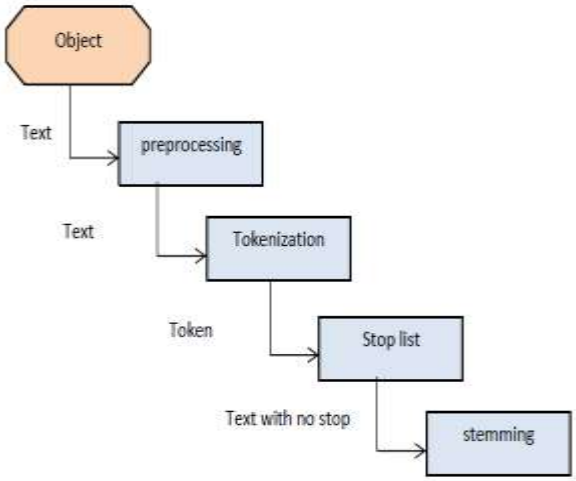

Figure (1): Processing stages applied to the object

The algorithm that has been used by Porter is a rule-based algorithm. It consists of four sets of rules that are applied in succession to obtain the root of the word [5].

Figure (1) is a box diagram of the processing stages applied to the object in the research before it is represented.

This algorithm adds the size and frequency factors of the item's request to the item's cost, so the equation that describes the item's key becomes: (Turney and Pantel, 2010)

K(o)=L+F(o)* ((CPU Cost + IO Cost)) S(o) (1)

The steps of the algorithm can be summarized as follows:

• Let the cache be in total bytes.

• Let be the amount of cache used to store cache files, and initially 0= used.

• For each object in cache 0, a frequency counter is created (o) Fr that determines how many times the object has been reached. When the object is not in the cache but is being prepared to be entered into the cache, a value of 1 is assigned to the frequency: 1 = (o) Fr.

• To specify the object that will be replaced when the cache size is exceeded, a priority frame of the objects is created**.**

When an object is jammed in this frame, it gives a priority computed by the equation:

Pr (o) = Clock + F(o) * C(o)/S(o) (2)

where the clock parameter is a "clock" of the message, starting with 0 and being updated for each object that is oevicted to the value of the priority key for that object in the priority Pr(oevicted).

The parameter (o) Fr is the frequency of the object. If the file is obtained from the cache, then Fr is incremented by 1, otherwise, it is assigned the initial value of 1.

The parameter (o) size is the size of the object 0.

The parameter cost (o) is the cost of acquiring the object 0.

1. If the object (o) is present in the cash, then the object is delivered to the customer from the cash, and it is

a. The amount of cash used does not change.

b. The frequency of the object (o) Fr is increased by 1.

c. The priority key (o) Pr is calculated according to the equation.

d. The clock parameter does not change.

e. The object (o) with the recalculated key is re-clamped into the priority frame to reflect the new state and object of the frame.

2. If the requested object (o) is not present in the cache, then the object is delivered to the client from the main server and is:

a. The object frequency (o) Fr is set to the first value of 1.

b. Purpose priority key (o) Pr is calculated using an equation.

c. The object (o) is crammed into the priority frame with the computed key value (o) Prd.
d. The amount of cash used is revalued according to the formula:
Used = used + size (o) (3)
Then one of two cases is addressed:
e. If used <= Total, this means that there is enough space in the cache to store the object (o), then no substitution of any object occurs in the cache, and the object is stored in the cache.
f. If used >= Total, this means that there is not enough space in the cache to store the object (o) and there are a set of objects that need to be replaced:
i. First, a minimum set of objects is selected to be removed from the cache using the following procedure:
The first k objects with a minimum priority key are selected in the priority frame o1, o2, ……, ok so that the property is:
Used Estimate <= Total (4)
Where:
Used Estimate = used- Σ_(i-1)^k size (oi) (5)
3. If the original object that we want to enter into the cache is not between objects o1, o2, ……, ok then the following procedure is done:
a. The clock parameter is calculated according to the equation:
Clock=max┬1sisk Pr (01) = Pr (ok) (6)
b. The amount of cash used is recalculated as follows:

Used=Used -Σ_(i-1)^k size (oi) (7)

c. Objects o1, o2, ……, ok are deleted.

d. The item is placed in the cache.
4. If the original object we want to add to the cache is between objects o1, o2, ……, then ok:
a. The object (o) is not stored in the cache, and (o) Pr is deleted from the queue.
b. No other purpose is removed from the cache.
The clock parameter is routinely incremented with each replacement of an object so that documents that have not been accessed for a long time will not change their clock value in the priority queue. At the same time, the clock value will become high so that any new object will be jammed in front of these objects, and it has not been accessed for a long time, so items of smaller size and higher frequency will be replaced if they are not accessed again. This mechanism prevents what is called cache pollution.
Figure **(2)** shows the activity diagram of the GDFS algorithm, showing how the Clock variable changes incrementally with the progress of deletes from the cache for the objects in the cache.

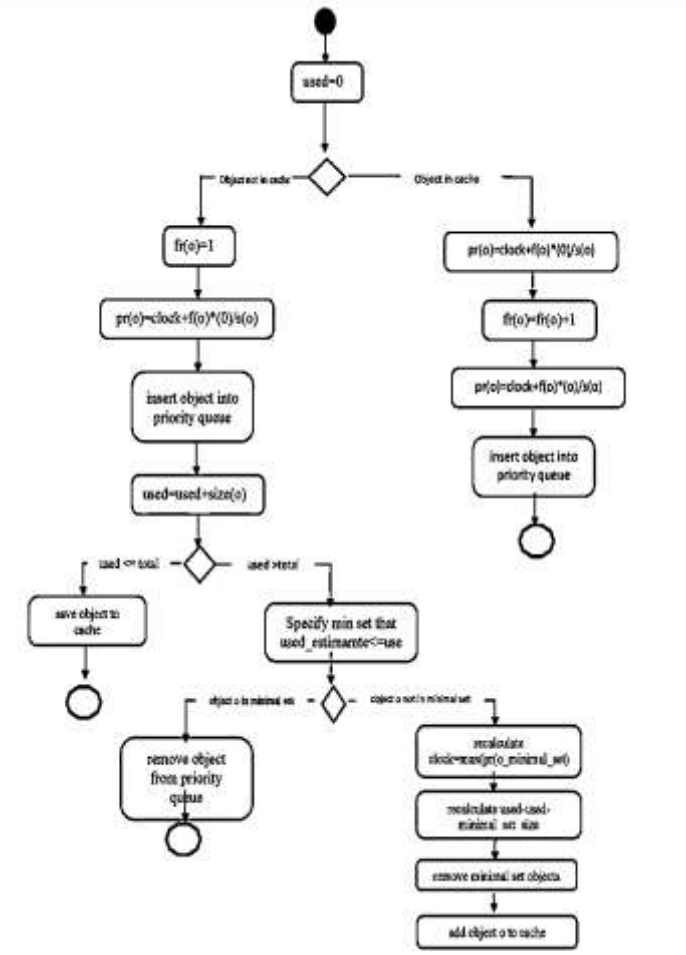

Figure (2): Proposed cache algorithm

3.2 Development of Web Cache Algorithms Using NGD:

This method depends on the number of pages returned by a particular search engine as a

result of the search to calculate the semantic similarity between words, and when using the Google search engine. This distance is called Google's measured distance to measure the similarity between words [10, 23].

The measured Google distance is known by law:

$$NGD(x,y) = \frac{\max\{logf(x), logf(y)\} - logf(x,y)}{logN - \min\{logf(x), logf(y)\}} \quad (8)$$

Where

f(x), f(y) is the number of Google search results for the terms x,y, f(x,y). The number of Google search results for both x,y, N is the total number of pages indexed by the search engine.

The measured Google distance has been adopted to study semantic similarity because it is one of the most recent measures of similarity within scale ratings and also gives better results to determine semantic similarity [2, 14].

4. Experimental Results of the Proposed Hybrid Cache Algorithm

The characteristics of the time required to retrieve results from the search engine were studied through the search application more than once, and as a result, the following characteristics were revealed:

- The time required to retrieve the result varies depending on whether you search for two keywords.
- Time is not affected by parallel requests, so the replacement function can be executed quickly.
- The length of the string returned by the search engine to the cache server is 75 bytes for a one-search process, which is very small in size and does not constitute noticeable traffic on the network. One of the important goals of the cache application is to ease the exchange of data on the network by distributing the load between the primary data server and the cache server.

Table (1): Comparison of byte hit rate, page load time, and hit rate of the developed algorithm

	GDFS-Line	GDFS-NGD
Load_time(Sec)	4.17	2.16
Hit_Rate(%)	76.48	80.10
Byte_Hit_Rate(%)	55	65

It can be seen that the higher the byte hit rate, the higher the hit rate of the GDFS algorithm, which confirms that the development of the algorithm using the measured Google distance led to an improvement in the performance of the algorithm in terms of hit rate and bypassing the problem of the low hit rate byte.

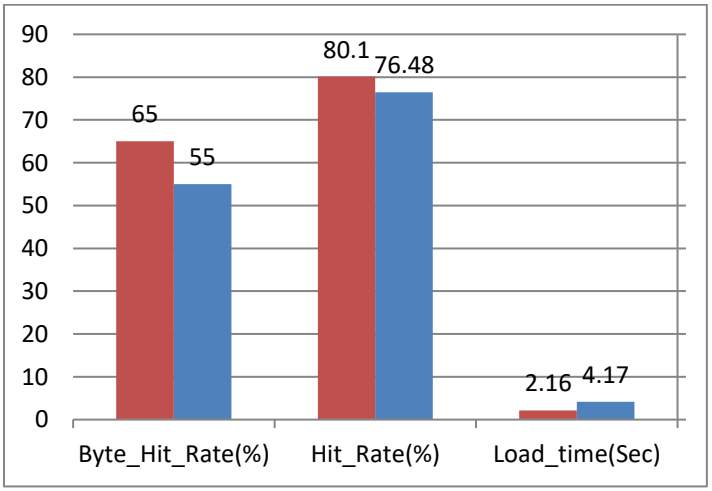

Figure (3): Page Load Times, Hit Rate, and Hit Rate Bytes for Algorithms Developed with NGD

Compared to the page load time without using the cache and with algorithms using the base replace function, the developed methods and the existing values confirm a significant improvement in page load time, which confirms that the application of the developed algorithms in the search will lead to a clear improvement in the performance of the

information system, as the page load time is reduced from 4.17 seconds using GDFS-Line to 2.16 seconds using GDFS-NGD.

Table (2): Comparison of page load time values without cache and with basic and developer algorithms

	LRU	LFU	GD	GDFS
No cache	14	14	14	14
Original	10.66	12	9	7
Lin	---	---	4.2	4.17
NGD	---	---	4.1	2.16

4.1 Discussion of the Results

• Adopting the “Web Cache” technologies in the development of information systems as much as possible due to the development they provide, whether at the level of improving the service provided to the customer by reducing page load time or by reducing network traffic by increasing byte hit rates.

• Using the GDFS algorithm due to the excellent rates it offers in terms of page load time and high hit rate.

• Using the semantic similarity techniques to support different algorithms, especially the cache algorithms.

- Use of Google's measured distance to support the "Web Cache" algorithm.

• The proposed development in the research makes the replacement function affected by the content of the object. Therefore, this development is appropriate for purposes with textual content, excluding images and video purposes, which can use the expressed keywords to apply the algorithms developed in the search.

5- Conclusion

As we know, there are cache substitution algorithms that work well with cache when applied to processor cache but hardly work when applied to web-purpose cache. The reason for this discrepancy is that it is not intended to increase the complexity of this type of storage. Considering the challenges of cache usage in terms of the large discrepancy in file size and the heterogeneous model of user access to data in a web environment, especially with pages with dynamic content, there is a real need to develop 'web cache' algorithms. In this paper, a hybrid algorithm for web caching using semantic similarity is developed. The GDFS algorithm was developed using NGD (Normalized Google Distance) to determine the semantic similarity between cache objects, which gave it better performance in comparison to other algorithms. The results showed that the web cache hybrid algorithm using semantic similarity increased the hit rate compared to the basic algorithms by up to 80.10%. The proposed hybrid algorithm was able to overcome the problem of the low byte hit rate in the GDFS algorithm. The improvement in the byte hit rate reached 65%, and this indicates an increase in the byte hit rate. The results showed that the web cache hybrid algorithm using semantic similarity reduced the page load time using distance measured from Google compared to the page load time using other algorithms that do not use semantic similarity and do not use cache. The results showed that the web cache hybrid algorithm using semantic similarity reduced the page load time from 4.17 seconds using GDFS-Line to 2.16 seconds using GDFS-NGD.

6. References:

[1] Kirilin, V., Sundarrajan, A., Gorinsky, S., & Sitaraman, R. K. (2020). RLCache: Learning-based cache admission for content delivery. IEEE Journal on Selected Areas in Communications, 38(10), 2372-2385.

[2] Wang, Y., Yang, Y., Han, C., Ye, L., Ke, Y., & Wang, Q. (2019). LR-LRU: a PACS-Oriented intelligent cache replacement policy. IEEE Access, 7, 58073-58084.
[3] Benhamida, N., Bouallouche-Medjkoune, L., & Aïssani, D. (2018). Simulation evaluation of a relative frequency metric for web cache replacement policies. Evolving Systems, 9(3), 245-254.
[4] Sara, M., & Rabiaa, M. (2016). IS/IT performance measurement system: Literature review and a comparative study. Paper presented at the 2016 International Conference on Information Technology for Organizations Development (IT4OD).
[5] Feng, Y., Bagheri, E., Ensan, F., & Jovanovic, J. (2017). The state of the art in semantic relatedness: a framework for comparison. The Knowledge Engineering Review, 32.
[6] Lee, J. S., Hah, H., & Park, S. C. (2017). Less-redundant text summarization using ensemble clustering algorithm based on GA and PSO. Wseas Transactions on Computers, 16.
[7] Taha, K., Peyman, R.-H., & Leila, G. (2016). DESIGNING AND EVALUATING THE WEB-BASED INFORMATION SYSTEM OF PRIMARY HEALTH CARE IN ACCORDANCE WITH THE ELECTRONIC HEALTH RECORDS OF IRAN. ACTA MEDICA MEDITERRANEA, 32, 2051-2054.
[8] Sugumar, R. (2018). Improved performance of stemming using efficient stemmer algorithm for information retrieval. Journal of Global Research in Computer Science, 9(5), 01-05.
[9] Nayak, A. S., Kanive, A. P., Chandavekar, N., & Balasubramani, R. (2016). Survey on pre-processing techniques for text mining. International Journal of Engineering and Computer Science, 5(6), 16875-16879.
[10] Panchenko, A., & Morozova, O. (2012). A study of hybrid similarity measures for semantic relation extraction. Paper presented at the Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data.
[11] Turney, P. D., & Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research, 37, 141-188. Arora, K., & Ch, D. R. (2014). Web cache page replacement by using lRU and lFU algorithms with hit ratio: a case unification. Int. J. Comput. Sci. Inf. Technol, 5 (3), 3232-3235.
[12] Aslam, Z. (2016). Semantic Web Mining in E-Commerce Websites. International Journal of Computer Applications, 137 (2).
[13] Sum sumathy, K., & Chidambaram, D. (2016). A hybrid approach for measuring semantic similarity between documents and its application in mining the knowledge repositories. International Journal of Advanced Computer Science and Applications, 7 (8).
[14] Benhamida, N., Bouallouche-Medjkoune, L., & Aïssani, D. (2018). Simulation evaluation of a relative frequency metric for web cache replacement policies. Evolving Systems, 9(3), 245-254
[15] Daoui, A., Gherabi, N., & Marzouk, A. (2017). An Enhanced Method to Compute the Similarity Between Concepts of Ontology.
[16] Paper presented at the International Conference on Information Technology and Communication Systems**.**
[17] Feng, Y., Bagheri, E., Ensan, F., & Jovanovic, J. (2017). The state of the art in semantic relatedness:a framework for comparison.The Knowledge Engineering Review, 32 **.**
[18] Fodeh, S., Punch, B., & Tan, P.-N. (2011). On ontology-driven document clustering using core semantic features. Knowledge and information systems, 28(2), 395-421.
[19] Geetha, K., & Gounden, N. A. (2017). Dynamic semantic lfu policy with victim tracer (dslv): a customizing technique for client cache. Arabian Journal for Science and Engineering, 42(2), 725-737.
[20] Ghofir, A., & Ginanjar, R. (2017). Distributed Cache with Utilizing Squid Proxy Server and LRU Algorithm. Indonesian Journal of Electrical Engineering and Computer Science, 7(2), 474-482.
[21] Hasslinger, G., Ntougias, K., Hasslinger, F., & Hohlfeld, O. (2018). Comparing web cache implementations for fast O (1) updates based on LRU, LFU and score gated strategies. Paper presented at the 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)**.**
[22] Kirilin, V., Sundarrajan, A., Gorinsky, S., & Sitaraman, R. K. (2020). RL-Cache: Learning-based cache admission for content delivery. IEEE Journal on Selected Areas in Communications, 38(10), 2372-2385.
[23] Kushwah, J. S., & Tamrakar, S. (2017). An extensive review of webs caching techniques to

reduce cache pollution. Imp J Interdiscip Res, 3, 111-118 **.**

[24] Lee, J. S., Hah, H., & Park, S. C. (2017). Less-redundant text summarization using ensemble clustering algorithm based on GA and PSO. Wseas Transactions On Computers, 16.

[25] Ma, T., Hao, Y., Shen, W., Tian, Y., & Al-Rodhaan, M. (2018). An improved web cache replacement algorithm based on weighting and cost. IEEE Access, 6, 27010-27017 **.**

[26] Sumathy, K., & Chidambaram, D. (2016). A hybrid approach for measuring semantic similarity between documents and its application in mining the knowledge repositories. International Journal of Advanced Computer Science and Applications, 7 (8.)

[27] Sumit, Rajoriya and Varsha Zokarkar. (2016). Improving Caching Technique through Innovative Replacement Algorithm to Page for Web Proxy

[28] Wang, S., & Koopman, R. (2017). Clustering articles based on semantic similarity. Scientometrics, 111(2), 1017-1031.

[29] Wang, Y., Yang, Y., Han, C., Ye, L., Ke, Y., & Wang, Q. (2019). LR-LRU: a PACS-Oriented intelligent cache replacement policy. IEEE Access, 7, 58073-58084.

[30] Li, J., Wu, J., Dán, G., Arvidsson, Å., &Kihl, M. (2014). Performance analysis of local caching replacement policies for internet video streaming services. Paper presented at the 2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM).

[31] Lofi, C. (2015). Measuring semantic similarity and relatedness with distributional and knowledge-based approaches. Information and Media Technologies, 10(3), 493-501.

[32] Ma, T., Hao, Y., Shen, W., Tian, Y., & Al-Rodhaan, M. (2018). An improved web cache replacement algorithm based on weighting and cost. IEEE Access, 6, 27010-27017.

[33] Moruz, G., Negoescu, A., Neumann, C., & Weichert, V. (2015). Engineering efficient paging algorithms. Journal of Experimental Algorithmics (JEA), 19, 1-19.