Using Genetic Algorithms to Optimize Web Caching in Multimedia-Integrated e-Learning Content

Article *in* International Journal of Digital Content Technology and its Applications · August 2011

DOI: 10.4156/jdcta.vol5.issue8.7

4 authors, including:

Cheruiyot Wilson Kipruto
Central South University

Felix Musau
Central South University

Joseph Cosmas Mushi
University of Dar es Salaam

Using Genetic Algorithms to Optimize Web Caching in Multimedia-Integrated e-Learning Content

[1]Cheruiyot W.Kipruto, [1]Guan-Zheng Tan, [1]Felix Musau, [1]Joseph C.Mushi
[1]*School of Information Science and Engineering, Central South University, P.R.China*
wilchery68@gmail.com
doi:10.4156/jdcta.vol5.issue8.7

Abstract

The evolution of e-learning as a new information technology built around the use of internet has originated new possibilities to develop pedagogical methodologies that provide the necessary knowledge and skills in the higher education environment. This has in effect given rise to a large number of learners who desire to access the educational resources provided by the new technology. But for many institutions of higher learning, the users often experiences poor performance when they try to access contents or download e-learning files. The main reasons for such are often performance problems concerning network infrastructure and the fact that many people tend to access the same piece of information repetitively. This paper uses web caching scheme and genetic algorithm optimization to effectively lessen this service bottleneck, diminish the user access latency and reduce the network traffic.

Keywords*: Multimedia Integrated E-learning, Web Cache, Genetic Algorithms*

1. Introduction to e-learning as used in institutions of higher learning

The Information and Communication Technology (ICT) Policy that has been adopted by most countries is aimed at integrating ICT into education and training systems. The objective is to enable the various governments carry out its education reforms by embracing the ICT and creating an e-enabled and knowledge-based society. To build an ICT-literate community, many countries have set up ICT structures in primary, secondary and tertiary institutions. This has led to a tremendous increase in the usage of internet, and for example in Kenya, there are currently millions of connected users. It is because of this connectivity to the internet that many institutions of higher learning are encouraging e-learning as a viable means of enabling large number of learners to access digital educational materials. E-Learning offers great services to all lecturers and students for the teaching and learning process. It presents different modules for learning such as assignment, blog, choice, course, forum, quiz, resource, upload and so on. Each of the users uses dissimilar modules at the same time [1].For a public web-based training content delivered via a web browser over Internet-scale service, which is available to tens of millions of users, individuals can experience extremely rapid and unexpected shifts in workload patterns during seasons of high demand. This results in traffic overload and “data spike” or “volume spike”, whose effects can lead to request processing failures that generally leads to users experiencing poor performance when they try to access the contents or download files. The term “spike," is commonly used to refer to a unexpected sustained increase in aggregate work-load volume [2].Reasons for such are often performance problems which occur directly on the servers, problems concerning the network infrastructure and a surprising fact is that many people tend to access the same piece of information repetitively [2]. To alleviate this problem, one method is to use Failed Request Tracing (FRT) tool [3] to troubleshoot request-processing failures. Alternatively, it is possible to get the generated output of the content page and save it in the cache, so that any subsequent request will be served from the cache only with no need to execute the same code again [3].In this paper we present a Genetic algorithm optimization technique to web caching as one of the methods that can be used to reduce the delay of data being accessed by e-learning users.

2. Multimedia-Integrated e-learning content

E-Learning can be thought of as the learning process created by interaction with digitally delivered content, services and support [4] that encourages the learners to develop the appropriate personal skills essential for independent learning based around ICT. Brandon Hall, a noted e-learning researcher, defines e-learning as “instruction delivered electronically wholly by a web browser, through the Internet or an intranet, or through CD-ROM or DVD multimedia platforms" [5]. Increasingly, the common understanding of e-learning relates exclusively to web-based training or learning content delivered via a web browser over a network.

One key characteristic of e-learning is its capability to integrate different multimedia content and create an instructional material, promoting the reading interests and willingness of the learner [6]. When integrated in e-learning systems, multimedia content offers a variety of media, rich in communication efficiency, with the ability to facilitate shared understanding within a time interval. Many e-learning content development companies are currently working with key subject matter experts to create an immersive and rich user experience content using multimedia so as to capture the imagination and engage with as wide an audience as possible. However, existing techniques for caching text and images are not appropriate for the rapidly growing number of continuous media streams [6]. Because of the high start-up overhead and isochronous requirement, a streaming media request typically is not started by a proxy server until sufficient blocks of data are cached locally. Such delayed starts can frustrate users. To overcome this problem requires any scalable caching solution that should store just a portion of each stream [7]. Such a proxy cache solution should store a fixed set of frames at the beginning of each streaming video (in effect a prefix); instead of storing the entire resource, as is typically the case in text and image caching [8]. Various caching decisions have been re-examined to accommodate the unique characteristics of streaming objects [9], and a prominent one that we have considered for use is the prefix caching technique. Using this method, the proxy only stores part of the initial frames and then uses work-ahead smoothing technique to ensure the high-quality playback. This enables the proxy to reduce client delay, without sacrificing quality. Upon receiving a client request, the proxy immediately initiates transmission to the client from the prefix cache, while simultaneously requesting the remainder of the frames from the server. In forwarding the stream, the proxy can capitalize on any quality-of-service guarantees (such as bounds on throughput and delay) along the path to the client [10].

3. The caching process for web-based multimedia-integrated e-learning

A proxy cache server receives HTTP requests from clients for a web object and if it finds the requested content object in its cache, it returns them to the user without disturbing the upstream network connection or destination server. If it is not available in the cache, the proxy attempts to fetch the object directly from the objects' home server. Finally the originating server which has the object gets it, possibly deposits it and returns it to user [11]. When shared with other users, the proxies serve many clients with cached objects from many servers. Since e-learning web caching is responsible for storing the document in the internal memory, it becomes apparent that if the cache contains more useful information for the user internally, the response time for the user and the network traffic will be reduced [12], hence significantly reducing server loads and improving the system responsiveness to e-learning requests.

Web caching systems can be classified into three types: Browser caches which are located within user browser programs, Surrogate caches typically located near the Web servers and are owned and operated by the Web content providers [12], and Proxy caches located near network gateways to reduce the bandwidth required over expensive dedicated internet connections, and typically configured and operated by ISPs and enterprises operating internal networks that are connected to the Internet. We implemented our web caching within the proxy web caching architecture using Squid running on windows operating system. Squid is used by hundreds of Internet Providers world-wide to provide their users with the best possible web access. It has been adopted widely by many institutions and installed at the main proxy server. Squid optimizes the data flow between client and server to improve performance and caches of frequently-used content to save bandwidth and also route content requests to servers in a wide variety of ways to build cache server hierarchies which optimize network throughput. For this reason, the client receives proper and non-stale data [13].

To preserve a constituent cache population, optimize cacheable objects and results of the e-learning content, our implementation adapts a Genetic Algorithms (GA) evolutionary approach to the cache information objects' update.

4. Genetic algorithms

Genetic Algorithms (GAs) are adaptive heuristic search algorithms premised on the evolutionary ideas of natural selection used in solving various computational problems that demands optimization and adaptation to changing environments. The optimization process is based on evolution, and the ''Survival of the fittest'' concept [14]. As such they represent an intelligent exploitation of a random search within a defined search space to solve a problem [15].For our paper, we adopted the GA iterative procedure [Figure 1] which consists of a constant-size population of individuals each one represented by a finite string of symbols, encoding a possible solution in a given problem space. This space, referred as the search space, comprises of all possible solutions to a given problem [13]. The standard GA generates an initial population of individuals, which is updated at each evolutionary step resulting in a new "generation". The individuals in the current population are decoded and evaluated according to some predefined criterion, called fitness function [13].

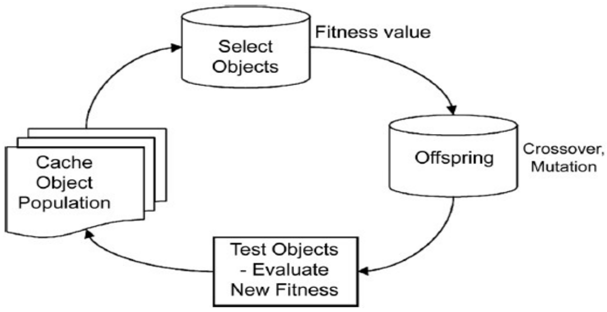

Figure 1. Cycle of GA applied in a space of individuals

GA optimization is adopted because of two main reasons: First, the basic idea of the GAs is based on the evolution of populations by the criterion "survival of the fittest" and the cache should contain the fittest (most fresh) information objects. Second, the GAs are applied to problems demanding optimization out of spaces which are too large to be exhaustively searched. For our e-learning content, the search space is the cache content which consists of millions of stored files.

4.1. The GA cache Model.

To adapt the GA approach to the cache update scheme, the cache is considered as a population of e-learning content individuals. An individual is the actual cached object identified by a distinct number within a filename where it's stored. The objects can be stored in files with the potential solution to the problem being investigated coded in hexadecimal numbering strings. Each individual is assigned with a fitness value derived by a general fitness function declaring the objects' freshness. The cache update scheme is implemented in order to perform cache reform and regeneration at regular time intervals [13].

4.1.1. Encoding and initialization of operators

The e-learning objects are encoded appropriately in order to participate in the GA caching model. Each object within the cache is identified by a distinct number. The cache is considered as a population of cached e-learning object individuals. For implementation purposes, the encoding scheme is chosen such that the potential solution to our problem is represented as a set of parameters, joined together to form the encoded string. This string has a length of *m* bits where *m* is the number of bits for the object identification number. Each individual is assigned with a fitness value derived by a general fitness function. This fitness function is related to a factor declaring the object's access frequency. To initialize the parent population for generation 0, a randomly selected set of cached e-learning objects is populated with a randomly chosen set of the encoded strings. Selection operation directly affects the performance of GA. The idea is to apply the GA process to the cache population in order to optimize cache content by evolving a successive number of cache generations. A pseudo code version of the GA scheme of our cache update implementation is as shown in Figure 2.

```
Initialize// old_pop =initial population
        generation <- 1
        while (generation<=  maxgen)  do
        par1<- selection(popsize, fitness, old_pop)
        par2 <- selection(popsize, fitness, old_pop)
        crossover(par1,par2, old_pop, new_pop, p_cross)
        mutation(new_pop, p_mutate)
        statistical_report(new_pop)
        old_pop <- new_pop
        generation  <- generation  +1
```

Figure2. GA scheme overview of the implementation

where *maxgen* corresponds to the maximum number of successive generation runs, *popsize* is the e-learning cache population size, *fitness* is the cache update factor, *par1* and *par2* are the parents chosen for the reform of each generation, *p_cross*, *p_ mutate* are the probabilities for crossover and mutation respectively. The implementation of crossover in Figure 2 gets old population and results in new population either by preserving the parents or by reproducing a new individual based on parents fitness. Mutation is performed on the new population by affecting few positions in the individuals' string towards a better fitness.

4.1.2. Crossover

As shown in Figure 3, crossover, operates by swapping corresponding segments of a string representation of a couple of chromosomes (called ''parents'') to produce two chromosomes (called ''children'') [16]. The *crossover* genetic function, applied to the selected individual population, after the reproduction [17], is performed between two cache e-learning object individuals ("parents") with some probability. Parents are sampled from the pool in order to identify two new individuals resulting by exchanging parts of parents' strings. The exchanging of "parents" parts using one-point cross-over is performed by cutting each individual at a specific position to produce two head and two tail segments [16]. The tail segments are then swapped over to produce two new full length individual strings. Consequently, fitter parents have a better chance of reproducing children.

```
crossOver( par1, par2 ) {
        prob1 = random();
        if( prob1 < crossRate )
        do {
                xPoint = random();
                for i = 1 to xPoint-1
                        child1[i] = par1[i];
                        child2[i] = par2[i];
                for j = xPoint to numEobjec
                        child1[j] = par2[j];
                        child2[j] = par1[j];
  }
  }
```

Figure 3. The recombination operator algorithm

4.1.3. Mutation

Mutation operates on a single chromosome. One element is chosen at random from the chain of symbols, and the bit string representation is changed with another one [17].The mutation operator to be performed in a chromosome is simply introduced in order to prevent premature convergence to local optima by randomly sampling new points in the search space. Mutation is applied to each "child" individually after crossover. It randomly alters each individual with a (usually) small probability (e.g. 0.001) [13]. The cache population will evolve over successive generations such that the fitness of the best and the average cached e-learning object individual in each cache generation is improved towards the global optimum.

4.1.4. The objective function

Each chromosome solution is evaluated by a predefined objective function. The value, derived from the objective function, is the fitness value, which evaluates the fitness value of an individual [19], representing the strength, or quality, of a solution. A set of individuals that have high scores in the fitness function of the cached e-learning objects are selected to reproduce itself. Such a selective process results in the best-performing chromosomes in the population to occupy an increasingly larger proportion of the population over time [17].Since fitness function drives the evolution of the GA population, it is important to reward the improved cache content individuals. Therefore, the metric characterizing cache e-learning objects freshness will be the best choice for the GA caching scheme. It is noted that all proxy caches relate their object's refresh policy with timing object's last modification period, therefore, in our GA caching, each individual object's fitness is evaluated by a factor corresponding to the ratio of object's ages, since retrieval and its last modification [13].We note that e-learning cache generations are evolved based on specific fitness function related to the object's freshness/staleness factor. In order to identify this factor, Squid's log files fields are used for each object's fitness function. Therefore, we use the following expression to derive the fitness function, F_o :

$$F_o = \frac{A_1}{A_2}$$

where A_1 corresponds to the time that passed since the object's retrieval and A_2 is the age of the object at the time of its retrieval. Fields of store.log, one of the four default Squids' log files, are used to evaluate each of the stated parameters.
By using these tags, the evaluation of the numerator results in:
A_1 = *now- datehdr*, and the denominator in A_2 = *datehdr-lastmod*,
where *datehdr* is the HTTP Date-reply header (of the Squid store.log fields) of each individual e-learning object.

5. Experiments Results

The experiments were tested on a simulator that accepts Squid proxy logs as input traces together with their corresponding logfiles. We collected and presented snapshot workload characteristics of the web proxy cache from a case of one university in Kenya, JKUAT, located at http://www.jkuat.ac.ke/index.php and used them to analyze the web user access usage patterns for the e-learning resources. We used Squidalyser, a popular squid traffic analyzer which is designed to allow per-user scrutiny and analysis of squid logfiles. Squidalyser also generate numerical and text information based on original server log files covering different aspects of the users' access log records.

Figure 4 shows the user requested web traffic trace patterns (requests) together with the number of bytes served from the web proxy server (received) that should normally occur when there is there is no web caching applied to the web infrastructure. In Figure 5 we show the web traffic pattern changes that respectively occur to the requested and received number of bytes as a result of applying the GA caching scheme. The traces are for the period starting from 14-Nov-2010, 00:13 A.M to 20-Nov-2010, 12:22 A.M.

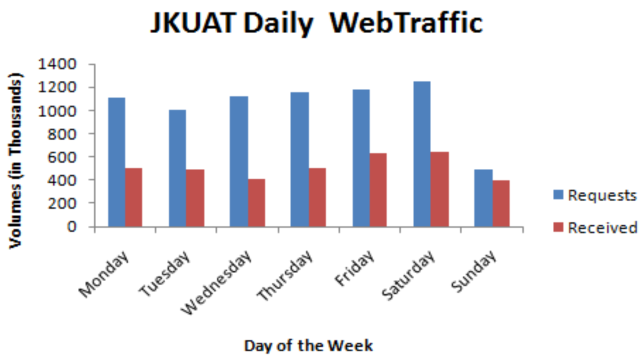

Figure 4. Requests-received web traffic trace patterns with no web caching application

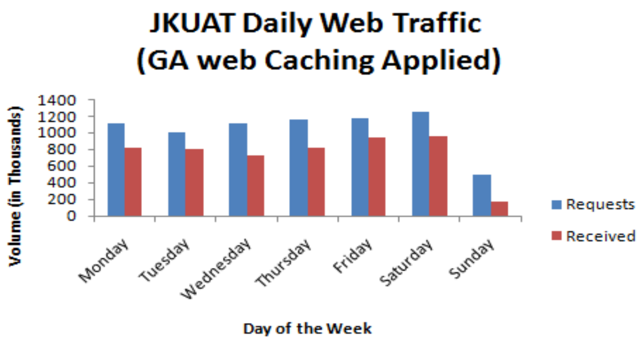

Figure 5. Requests-received web traffic trace patterns with GA web cache optimization application

GA scheme was applied to the initial cache population produced by the squid proxy. Figure 6 presents the average and maximum fitness values for a cache population being reproduced by generations number 1 to 10 of intervals 10, 20,…, 100. The Crossover probability used is 0.6 whereas mutation probability is 0.0333. These values have been suggested as a representative trial set for most GA optimizations.

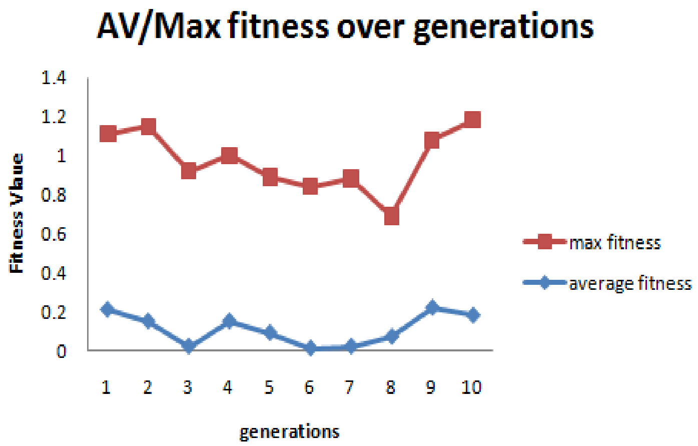

Figure 6. Av/Max fitness over generations

It is observed that the populations' average fitness value decreases generally for the 10 generations' reproduction whereas the maximum fitness remains at almost the same rate, with a slight decrease at generation number 8. Decreasing average objects fitness is important since cache will consist of more fresh members. Figure 7 refers to the KBytes length variation at each cache generation. This length is the actual length and not the expected of the store/log headers.

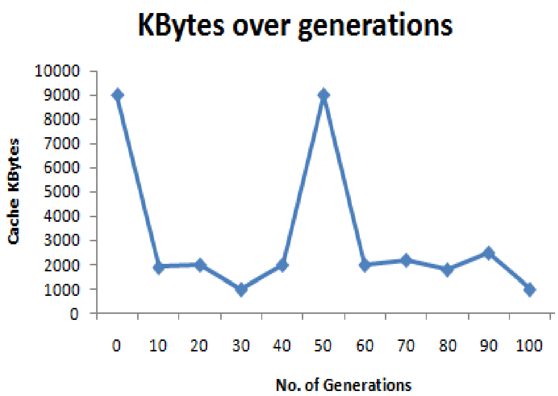

Figure 7. Kbytes over generations

It is observed that this length is reduced significantly for larger number of generations (>50) and depends on the types of files that e-learning cache objects contains. For example, the gif, jpeg and mpeg files occupy more space.

6. Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (50275150) and the National Research Foundation for the Doctoral Program of Higher Education of China (20040533035, 20070533131).

7. Conclusions and future research directions

This paper presented a study that applies the genetic algorithm**s** idea to web caching of multimedia integrated e-learning data objects to improve web content access. The simulation process included most of the necessary parameters to study the model under real Squid cache traces. The technique is shown to save network bandwidth and server processing workload, and the GA scheme has been proven quite effective since cache population evolved over the simulation time for an increasing number of generations. More research is needed to further experiment the present scheme for better configuration, scaling and under different fitness selection policies. Other evolving computation schemes like simulated annealing, threshold acceptance could be adopted in the e-learning web caching in order to study their effect on cache consistency.

8. References

[1] Sarina Sulaiman, “Intelligent Web Caching for E-learning Log Data’, International Journal of Computer Mathematics, UK, vol.76 no.2, pp 297-305, 2007.

[2] Peter Bodik, Armando Fox, Michael Franklin, Michael Jordan, and David Patterson, “Characterizing, modeling and generating workload spikes for stateful services”, In First ACM Symposium on Cloud Computing (SOCC 2010), Indianapolis, IN, pp.241-252, 2010.

[3] Mostafa E Saleh,A.Abdel Nabi, A.Baith Mohamed, “A Design and Implementation Model for Web caching Using Server “URL Rewriting”, World Academy of Science, Engineering and Technology, issue 60, pp.303-307, 2009.

[4] Riad A.M, Ei-Ghareeb, “Service Oriented Architecture to integrate mobile assessment in learning management systems”, Turkish Online Journal of Distance Education-TOJDE”, vol. 9 no.2, pp.1302-6488, 2008.

[5] Codone, S.,”AN E-learning Primer”, Raytheon Interactive, 2001(n.d).Retrieved from http://faculty.mercer.edu/codone_s/elearningprimer.pdf

[6] Vichuda K.,.Ramamurthy,K., Haseman, W.D.,“User attitude as a mediator of learning performance improvement in an interactive multimedia environment”, International journal of Human-Computer Studies, vol.54,no.4, pp.541-583,2001.
[7] J.Liu, X.Chu, and J.XU, “Fine-grained scalable video caching for Heterogeneous clients,’ IEEE Transactions on Multimedia, vol.8, no.5, pp.1011-1020, 2006.
[8] R.Rejaie, M.Handly, H.Yu, and D.Estrin, “Proxy caching mechanism for multimedia playback streams in the internet”, In Proceedings of the Fourth Int. Web Caching Workshop, March 1999.
[9] Feng Wang, “Storage Management in Large Distributed Object-Based storage systems”, December 2006.
[10] S.Sen, J.Rexford, and D.Towlsey, “Proxy prefix caching for multimedia streams,” IEEE INFOCOM,New York, pp1310-1319, Mar. 1999.
[11] Kayvan Tirdad, Fatemeh Pakzad, Dr. Abdolreza Abhari, “Cache Replacement Solutions by Evolutionary Computing Technique”, In proceedings of the 2009 Spring simulation Multiconference, San Diego, California, Society for Computer Simulation International, pp.1-4, 2009.
[12] D. Zeng,F.Wang, and M.Liu, “Efficient Web content delivery using proxy caching techniques”, IEE transactions on systems, MAN, and Cybernetics-Part c: Applications and reviews, vol. 34, no. 3, pp.270-280, 2004.
[13] Athena Vakali, **“**Evolutionary Techniques for Web Caching”, Distributed and Parallel Databases, vol.11, no.1, pp. 93–116, 2002.
[14] Cheruiyot Wilson, Guan-Zheng Tan, Joseph Cosmas Mushi, Felix Musau, "Genetic Algorithm-Enhanced Retrieval Process for Multimedia Data", IJACT, Vol. 3, No. 3, pp. 153 ~ 167, 2011.
[15] Abeysundara S., Giritharan B., Kodithuwakku S, “A Genetic Algorithm Approach to Solve the Shortest Path Problem for Road Maps”, In Proceedings of the International Conference on Information and Automation, pp. 272-275,2005.
[16] Cheruiyot Wilson,Guan-Zheng Tan, Felix Musau, Cosmas Mushi, (in press), “Query quality refinement in singular value decomposition to improve genetic algorithms for multimedia data retrieval”’Springerlink journal.
[17] Massimo Orazio Spata, Salvatore Rinaudo, "Merging Nash Equilibrium Solution with Genetic Algorithm : The Game Genetic Algorithm", JCIT, Vol. 5, No. 9, pp. 9 ~ 15, 2010.
[18] Mu-Jung Huang, Hwa-Shan, Mu-Yen Chen, “Constructing a personalized e-learning system based on genetic algorithm and case-based reasoning approach”, Expert Systems with Applications vol.33, no.3, pp.551-564, 2007.
[19] Huijuan Lu, Wutao Chen, Xiaoping Ma, Mingyi Wang, Jinwei Zhang , "Model-free Gene Selection Using Genetic Algorithms ", JDCTA, Vol. 5, No. 1, pp. 195 ~ 203, 2011.