Selecting the most suited cache strategy for specific streaming media workloads

Marcio Neves, Moises Rodrigues, Ernani Azevêdo, Djamel Sadok
GPRT – UFPE
Recife, Brazil

Arthur Callado
Quixadá Campus – UFC
Quixadá, Brazil

Josilene Moreira
Department of Informatics – UFPB
João Pessoa, Brazil

Victor Souza
Ericsson Research
Kista, Sweden

Abstract — **Streaming media usage on the Web has recently increased by over 100% a year, and Video-on-Demand applications represent 40% of the Internet traffic. To improve the quality of streaming media delivery, a proxy server using cache strategies placed closer to end users is one of the main solutions used by service providers. The problem faced by cache managers is the difficulty decide, given workload characteristics such as number of objects, popularity distribution, object size etc, which strategy should be used to achieve the best performance. This work presents a deep evaluation of how traditional approaches behave over a wide range of scenarios and suggests which cache strategy should be used, given a specific workload; furthermore, it also introduces two new strategies that perform better than the others in the majority of evaluated scenarios.**

Keywords-component; cache strategy, workload, streaming

I. INTRODUCTION

Streaming media on the Web has increased by 100% a year. By 2014 all forms of video (TV, on-demand, Internet video and peer-to-peer) will exceed 91% of global traffic and Internet video will account for 57%, from 40% in 2010. As the popularity of User Generated Content (UGC), video news, movies and TV series grow, broadband consumption increases, impacting Internet traffic. Unlike traditional TV, congestion and jitter are unpredictable in the internet, resulting in low Quality of Experience (QoE) for users, with long start-up delays and losses of playback continuity.

A key solution is the use of caches to store content closer to the end user, reducing delays and bandwidth consumption [1]. New techniques are developed every day and it is difficult to choose one. Each new technique claims benefits though often in specific conditions while using synthetic workloads or real collected traces. What is perceived is that no single cache strategy achieves the best overall performance under any kind of workload. The goal of this paper is to evaluate a number of cache strategies available under different kinds of workloads and classify them according to suitability for use within specific network behavior, which may be used by a cache administrator when choosing the best strategy. We focus on streaming, which is often heavier and more complex to accommodate due to dynamic user sessions and flash crowds.

II. RELATED WORK

Multimedia objects are larger than text and image ones. Web pages are typically small and present a highly dynamic update behavior, while services such as VoD present less dynamic changes. Storing large multimedia objects entirely can exhaust the capacity of a proxy cache. Therefore, distinct strategies have been proposed to deal with cache replacement of large objects, such as partial caching [2][3] and [4].

In [2] the authors propose a segment-based algorithm to partition an object into exponentially variable-sized segments based on their distance from the beginning. This reduces the cache-replacement granularity and achieves better byte-hit ratio. However, it has some parameters, such as the number of segments (K_{min}) comprising the initial segment, called prefix, and the capacity of the cache reserved for them (C_{init}), that must be pre-configured according to workload's characteristics.

An algorithm is proposed in [3] for capturing changes in object popularity considering both recency and frequency of requests. This scheme is based on the exponential segment-based approach presented in [3], but introduces a new policy of caching replacement by considering LRU and LFU selection modes. It outperforms the segment-based approach in terms of byte-hit ratio and fraction of requests with delayed start up but has the same drawbacks of pre-configured parameters.

Reference [4] proposes an adaptive caching strategy which adapts to real time access and segments the object only after assessing user's behavior. It is composed of an aggressive admission policy, a lazy segmentation approach and a two-phase iterative replacement policy. It captures the object popularity through a complex caching utility function. Although it achieves much better byte-hit ratio, it does not prevent initial start-up latency well.

A. *Problem Statement*

Despite the analysis presented in these studies that introduce a new caching strategy, [5][6] and [11], it is very difficult to choose the best technique suited for a specific kind of video streaming service. This is mainly because there are no strong references offering a clear comparison or guideline for selecting among several caching strategies present in the literature considering simple workload aspects such as popularity distribution, the objects sizes, storage capacity.

To address these issues, we propose a new method to evaluate and classify multimedia caching strategies considering simple workload parameters that can be input by the service provider in order to improve specific metrics and to assist service operators selecting the suitable algorithm for one specific scenario. Furthermore, we propose two novel proxy-

978-3-901882-50-0

cache schemes with specific behavior and objectives. The first scheme called PCMA (Proxy Cache Moderate Approach) aims at balancing both byte-hit ratio and the fraction of requests with delayed start, improving QoE for both metrics. The second one, named PCAA (Proxy Cache Aggressive Approach), is more aggressive and is targeted at increasing the byte-hit ratio over the fraction of requests with delayed start.

III. PROPOSED ALGORITHMS AND COMPARISON MODEL

The proposed schemes were developed based on several approaches described in earlier works. They improve cache efficiency by creating new techniques that intelligently combine the strengths of existing strategies from the literature.

A. *Media Segmentation*

Prefix size is the smallest initial portion of media object contained in the cache and directly impacts start-up latency. In other works, prefix size was fixed independently of the video size [2]. In our approach, the prefix is set at 10% of the object size; the next segments each double the size of the previous. This strategy creates a dynamic exponential segmentation that consists of four pieces (10%, 20%, 40% and the remaining 30%) and helps the cache algorithm quickly discard a big portion of cached media object without the overhead of traditional exponential segmentation. Fig. 1 compares the Exponential and our media segmentation strategies.

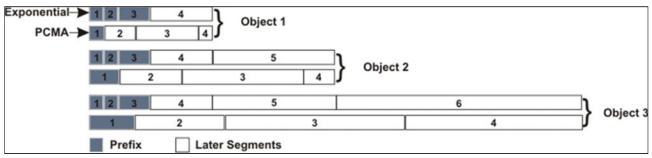

Figure 1. PCMA Segmentation versus Exponential Segmentation

B. *Admission Policy*

When there is sufficient free space to cache a media object, the entire object is cached. From the moment in which there is not enough space, the cache applies the admission process. In this phase, we consider two approaches: PCMA and PCAA.

In PCMA, if it can't cache a new video entirely, only the prefix will be cached until video requests appear. It performs aggressively when there is space available and selectively otherwise, choosing only the most popular segments, which reduces the start-up delay perceived by the user.

The PCAA is more aggressive and any requested video will be completely cached. Without free space, it also selects the least popular content from the cache do delete. This proposal increases the byte-hit ratio over the start-up latency's decrease.

C. *Replacement Policy*

When there is no free space, a priority function is computed for every object stored in cache which is not being streamed. The object with the smaller priority is chosen for eviction. The last segment of the object is removed and this procedure is repeated until there is space to store a requested segment.

The caching priority function (CP) considers several aspects of the multimedia object like the access frequency in a period of time, the number of accesses and the probability of future access to compute its value. Properties are maintained in a data structure for each entry in the cache. It is shown in (1).

$$CP = \frac{n}{(T_c - T_r)} * MIN\left\{1, \frac{\frac{T_r - T_1}{n}}{T_c - T_r}\right\} \quad (1)$$

n is the number of accesses for the entry, *Tc* is the current time stamp, *T1* is the object first access time, *Tr* is the object last access time. $\frac{n}{(T_c - T_r)}$ is the average number of accesses in the period [3] to combine the LRU and LFU in a single score. $MIN\left\{1, \frac{\frac{T_r - T_1}{n}}{T_c - T_r}\right\}$ is the probability of future accesses [4].

D. *Performance Metrics*

To the best of or knowledge, there is no other comparison model present in the literature that goes beyond the evaluation of two metrics: the byte-hit ratio (BHR) and delayed start request ratio (DS) [4]. Byte Hit Ratio measures how many bytes are delivered directly to the client from the proxy cache, normalized by the total bytes requested. Delayed start request ratio is also a popular metric, which measures how many requests had startup latency among the total requests.

In our study, we introduce a balanced factor (BF). This metric combines BHR and DS in a single score, to show which technique is able to achieve the best performance for BHR and for DS at the same time ($BF = \frac{BHR}{DS}$).

E. *Strategies Classification*

We classify algorithms according to prefix special treatment, admission policy and behavior (aggressive caches whole videos, conservative has strict prefix policy and moderate changes behavior according to space available). TABLE I summarizes the strategies regarding these aspects.

TABLE I. STRATEGIES CLASSIFICATION

	Prefix special treatment	Admission policy	Algorithm Behavior
Exp.	Yes	Always cache the prefix	Conservative
LRLFU	Yes	Always cache the prefix	Conservative
Lazy	No	Always cache the entire video	Aggressive
PCMA	Yes	Cache entire video if there is free space otherwise cache prefix	Moderate
PCAA	No	Always cache the entire video	Aggressive

IV. PERFORMANCE EVALUATION

Parameters that directly affect caching performance such as number of requests, number of videos, mean video length, popularity distribution, cacheable objects ratio and user session behavior have to be taken into account when designing and comparing strategies [7]. We use several synthetic scenarios built by combining a group of specific characteristics that influence the performance of cache algorithms. Though impossible to create all parameter combination possibilities, we have built a representative model based on [9][10].

A typical Web movie is 700MB while an YouTube video averages 10MB. Our synthetic workloads use two file sizes [10]: "Small video", with an average size of 10MB (standard deviation 2), and "large video", with average 600MB (s.d. 60).

Based on "impatient audience" [9] and interrupted sessions statistics, our workloads have short sessions (80% of users watch 20% of objects and 20% watch it all) and long sessions (25% watch 25%, 25% watch 50%, 50% watch all).

Studies in [4][8] and [7] show that the popularity influences cache performance and VoD popularity can be modeled by a Zipf distribution. We model the file popularity in our synthetic workloads using Zipf $\alpha = 0.4$ for "spread" popularity and $\alpha = 0.8$ for "skewed". We consider two values for the number of objects [8]: "low", where the number of objects corresponds to 10% of the total requests, and "high", where it accounts for 40%. The percentage of cacheable objects can be "high" (50%) or "low" (30%) [4].

A group of synthetic scenarios (S) was generated through ProWGen [7]. Each scenario is a set of values for one type of workload characteristics referenced in TABLE II.

TABLE II. FACTORS AND LEVELS OF SYNTHETIC WORKLOADS

Characteristic	Values	Comments
Requests	10,000	Low
	50,000	High
Objects	10% of requests	Low
	40% of requests	High
Video Size	10 MB	Small
	600 MB	Large
Popularity distribution	Zipf(α=0.4)	Spread
	Zipf(α=0.8)	Skewed
Cacheable objects	30%	Low
	50%	High
User Session	25% watch 25% of the media, 25% watch 50% and 50% watch completely	Long
	80% watch 20%, 20% watch all	Short

32 Scenarios (S1-S32) were created representing possible workload configurations (see Fig. 2). For the 50,000 requests experiment, 32 Scenarios (S33-S64) were created as well (same setup, not shown).

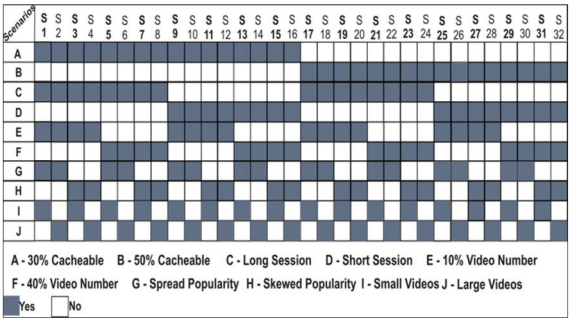

Figure 2. Synthetic scenarios evaluation model

A. Simulation Tool

Our Simulator, P2PCDNSim [12][13], provides a clear separation between different overlay "operational modes" or paradigms and the common underlying support strategies and functionalities which include caching strategies, placement algorithms, the simulator's event scheduler, metric collection functions and the plotting of graphs. This was a good choice as we could expand our scenarios to some initially unforeseen ones such as the study of CDNs with and without virtualization support, Hybrid CDNs and P2P systems and multiple cooperating CDNs within a single topology. The reader is invited to test it at cdn1.gprt.ufpe.br through guided scenarios.

B. Simulation Results

We considered that the strategy that achieves the highest value in the case of byte-hit ratio (BHR) and balance factor (BF) or the lowest value for delayed start (DS) is considered the most appropriate to be used in such scenario. We also considered that a strategy with 98% or more of the value of the best one is equally suited for the scenario. Concerning the strategy configuration, PCMA and PCAA do not need any parameters. The Lazy strategy was configured according to the guidelines in [4]. The C_{init} and K_{min} LRLFU and Exponential parameters were set to 10% and 6, according to [2].

1) Byte-hit Ratio 10,000 Requisition

TABLE III. BYTE-HIT RATIO FOR 5% OF CACHE SIZE

Scenarios	Most suited
S5, S6, S7, S8, S13, S14, S15, S16, S17, S18, S21, S22, S23, S24, S25, S26, S29, S30, S31, S32	PCMA
S1, S2, S3, S4, S9, S10, S11, S12, S19, S20, S27, S28	PCMA, PCAA

When cache size is very small (around 5% of videos), TABLE III shows that moderate strategies like PCMA achieve highest byte-hit ratio.

When cache size is 20%, behavior is similar (TABLE IV).

TABLE IV. BYTE-HIT RATIO FOR 20% OF CACHE SIZE

Scenarios	Most suited strategy
S5, S6, S21, S22, S29, S30	PCMA
S1, S2, S3, S4, S7, S8, S9, S10, S11, S12, S13, S14, S17, S18, S19, S20, S23, S24, S25, S26, S27, S28, S31, S32	PCMA, PCAA
S15, S16	PCMA, PCAA, Lazy

When cache size increases to 40% and the percentage of cacheable traffic reaches 30% under short sessions, Lazy strategy improves its performance (TABLE V). PCAA is the most adequate with long session and high video number (50%).

TABLE V. BYTE-HIT RATIO FOR 40% OF CACHE SIZE

Scenarios	Most suited strategy
S5, S6, S7, S8, S21, S22, S23, S24	PCAA
S9, S10, S11, S12, S13, S14	Lazy
S17, S18, S25, S29, S31, S32	PCMA, PCAA
S1, S2, S3, S4, S15, S16, S19, S20, S26 S27 S28 S30	PCMA, PCAA, Lazy

When the cache size reaches 60% (TABLE VI) Lazy strategy is the correct choice in most scenarios. However, when we have the users watching a good portion of the media and a high number of objects, PCMA and PCAA outperform Lazy.

TABLE VI. BYTE-HIT RATIO FOR 60% OF CACHE SIZE

Scenarios	Most suited strategy
S9, S10, S11, S12, S13, S14, S15, S16, S25, S26, S27, S28, S29, S30	Lazy
S21, S22	PCAA
S5, S6, S7, S8, S23, S24	PCMA, PCAA
S1, S2, S3, S4, S17, S18, S19, S20, S31, S32	PCMA, PCAA, Lazy

Clearly, cache size has a great impact over byte-hit ratio.

2) Delayed start 10,000 Requisitions

With 10,000 requests LRLFU and Exponential show less delayed start for larger media objects. Otherwise, PCMA is the most suited (TABLE VII).

TABLE VII. DELAYED START FOR 5% OF CACHE SIZE

Scenarios	Most suited strategy
S1, S5, S7, S9, S3 S11, S13, S15, S17, S19,	PCMA

S21, S23, S25, S27, S29, S31	
S2, S4, S6, S8 S10, S12, S14, S16, S18, S20, S22, S24, S26, S28, S30, S32	LRLFU, Exponential

When LRLFU and Exponential have the parameters correctly configured they achieve the smallest delayed start but if parameters are not well set, strategies fails and bad results are experienced.

Increasing cache size up to 40%, LRLFU and Exponential maintain their performance for large video sizes while PCMA and PCAA improve their performance (TABLE VIII).

TABLE VIII. DELAYED START FOR 40% OF CACHE SIZE

Scenarios	Most suited strategy
S1, S17, S19, S21, S23, S3, S5, S7	PCMA
S18, S20, S22, S24, S26, S28	LRLFU, Exponential
S11, S13, S15, S25, S27, S29, S31, S9	PCMA, PCAA
S2, S4, S6, S8	PCMA, LRLFU, Exponential
S10, S12, S14, S16, S30, S32	PCMA, PCAA, LRLFU, Exponential

3) *Balanced factor 10,000 Requisitions*

For 5% cache size, PCMA performs best in most scenarios. LRLFU and Exponential also achieve best results in some scenarios mainly because of delayed start (TABLE IX, Fig. 4).

TABLE IX. BLANCE FACTOR FOR 5% OF CACHE SIZE

Scenarios	Most suited strategy
S1, S3, S5, S6, S7, S9, S11, S13, S14, S15, S16, S17, S19, S21, S22, S23, S25, S27, S29, S30, S31	PCMA
S2, S4, S8, S10, S12, S18, S20, S24, S26 S28, S32	LRLFU, Exponential

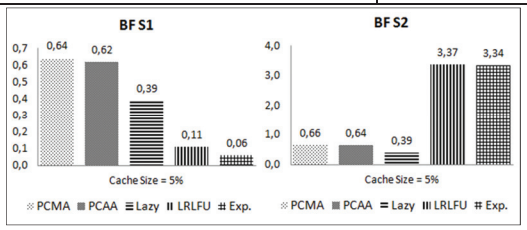

Figure 3. Balance factor for 5% of cache in S1 and S2

When cache size is 40% we can see that PCMA and PCAA are most suited. Results are summarized data in TABLE X.

TABLE X. BLANCE FACTOR FOR 40% OF CACHE SIZE

Scenarios	Most suited strategy
S17, S18, S19, S20, S21, S23, S24	PCMA
S1, S10, S11, S12, S13, S14, S15, S16, S2, S22, S25, S26, S27, S28, S3, S4, S5, S7, S8, S9, S29, S30, S31, S32, S6	PCMA, PCAA

4) *50,000 Requisitions*

Using 50,000 requisitions, the results observed for byte-hit ratio and delayed start were very similar to those in the previous experiments (not shown).

For balance factor, again PCMA was the best metric in most scenarios, but unlike the experiments with 10,000 requisitions, here LRLFU and Exponential have the best results in some scenarios with has large videos and low video number.

V. CONCLUSION AND FUTURE WORK

Our study showed that no single strategy provided the best results on a variety of scenarios: difficulty relies on determining which one to use in each scenario. In this work we show that cache size and workload behavior have great impact in strategies and demonstrate that the number of requests provide small impact if the overall behavior is maintained.

PCMA, a moderate technique, achieves the highest byte-hit ratio when having small cache sizes. The more aggressive ones, like Lazy and PCAA, have their best results when plenty of cache space is available. The conservative ones have the worst performance in reducing the bandwidth usage mainly because of the space reserved for prefix, compromising later segments. Regarding the delayed start metric, LRLFU and Exponential are the ones to be chosen if their manual parameter (C_{init} and K_{min}) could be well set. If this is a difficult task to be performed, PCMA should be used. To reduce the delayed start the aggressive ones are not advisable in most of scenarios even with plenty of cache space available.

The balance factor introduced in this work showed that it is difficult to reduce bandwidth consumption and at the same time try to provide a better experience to users. Most strategies available have a good performance in one metric in detriment of the others. The strategy which achieves the best performance concerning this metric is PCMA, developed to provide a good equilibrium between the QoE and network costs.

As future work, we aim at the development of an adaptive algorithm that selects the right strategy on-line given a specific workload to dynamically improve cache server performance.

REFERENCES

[1] M. Hofmann, L.R. Beaumont, Content Networking: Architecture, Protocols, and Practice, Morgan Kaufmann Publishers, USA, 2005.

[2] K. Wu, P.S. Yu, J.L. Wolf., Segmentation of multimedia streams for proxy caching, IEEE Transactions on Multimedia, IEEE, 2004.

[3] A. Satsiou, M. Paterakis, Efficient caching of video content to an architecture of proxies according to a frequency-based cache management policy, Proc. of the 2nd International Workshop on Advanced. Arch. and Algorithms for Internet Delivery, ACM, 2006.

[4] S. Chen, H. Wang, X. Zhang, B. Shen, S.Wee, Segment-Based Proxy Caching for Internet Streaming Media Delivery, IEEE Journal of MultiMedia, IEEE, 2005.

[5] J. Li, Z. Chen, Sliding-window caching algorithm for streaming media server, Proc. of the 2nd International Conf. on Interaction Sciences: Information Technology, Culture and Human (ICIS '09), ACM, 2009.

[6] Z. Zeng, B. Veeravalli, K. Li, A novel server-side proxy caching strategy for large-scale multimedia applications, Journal of Parallel and Distributed Computing, 2011.

[7] M. Busari, C. Wiliamson, ProWGen: A Synthetic Workload Generation Tool for Simulation evaluation of Web Proxy Caches, Journal of Computer Networks, 2002.

[8] R. Buyya, M. Pathan, A. Vakali, Content Delivery Networks, Springer, 2008.

[9] H. Yu, D. Zheng, B.Y. Zhao, W. Zheng, 2006. Understanding user behavior in large-scale video-on-demand systems, Proc. of the 1st ACM SIGOPS/EuroSys, 2006.

[10] X. Cheng, C. Dale, J. Liu, Statistics and Social Network of YouTube Videos, IWQoS, 2008

[11] F. Li; J. Li; Z. Hu, J. Zhou, Common Caching Replacement Algorithm for Video-on-Demand System, WISM 2009.

[12] M. Rodrigues, M. Neves, J. Moreira, D. Sadok, A. Callado, P. Karlsson, V. Souza. On traffic locality and QoE in hybrid CDN-P2P networks. Proceedings of the 44th Annual Simulation Symposium. Boston, 2011.

[13] A. Moreira, J. Moreira, D. Sadok, A. Callado, M. Rodrigues, M. Neves, V. Souze and P. Karlsson. A Case for Virtualization of Content Delivery Networks. In: IEEE Winter Simulation Conference. Phoenix, 2011.