Avaliando Técnicas de Cache para a Distribuição de Vídeo sob-Demanda na Internet

Marco Schulze, Josilene Aires Moreira
Universidade Federal da Paraíba
Cidade Universitária - João Pessoa - PB - Brasil - CEP: 58051-900
+55 (83) 3216-7093
marco.shulze@di.ufpb.br, josilene@ci.ufpb.br

RESUMO
A distribuição de vídeo na Internet tem crescido significativamente nos últimos anos. Porém os objetos de vídeos são bastante grandes e provocam congestionamentos, atrasos e alto consumo de banda na rede. As técnicas de cache têm sido propostas para minimizar estes impactos, incluindo abordagens de segmentação e de substituição de objetos. Este trabalho modifica técnicas já estabelecidas na literatura, melhorando o desempenho medido em acerto em bytes em até 65% para caches pequenas, enquanto também reduz o percentual de objetos com atraso inicial.

Categorias e Descritores
C.4 [**Measurement Techniques**]

Termos Gerais
Gerenciamento, Medição, Performance.

Palavras-chave.
Gerenciamento de Cache, Vídeo sob-Demanda, Distribuição de Vídeo na Internet

1. INTRODUÇÃO
O uso das aplicações de vídeo na Internet tem aumentado de modo muito significativo nos últimos anos. Particularmente, as aplicações de Vídeo sob-Demanda (VoD) têm apresentado um intenso crescimento: este tipo de aplicação foi responsável por 40% do tráfego na Internet em 2010 e, de acordo com recentes projeções, será responsável por 57% de todo o tráfego na Internet em 2014 [1]. Diversos autores reportam o fenômeno da criação e publicação na Web de vídeos criados pelos próprios usuários (*User-Generated Content - UGC*) como uma das principais causas deste crescimento [3][12]0.

YouTube [2] é o mais bem sucedido serviço de hospedagem e compartilhamento de vídeos UGC desde que foi criado em 2005, com 65.000 vídeos publicados diariamente e com um acervo de mais de 100 milhões de objetos. De modo semelhante, o serviço Daum UCC, o mais popular UGC na Coréia, serve dois milhões de visitantes e atende mais de 35.000 visualizações por semana, sendo conhecido por fornecer vídeos de alta qualidade, transmitidos a uma taxa de 800 Kb/s [3].

Entretanto, a distância dos servidores remotos para os clientes ainda é um obstáculo para a transmissão dos vídeos, que em geral são grandes e ocupam muita banda até chegar ao usuário final. Com a finalidade de reduzir o consumo de banda e também de oferecer uma maior qualidade, diversas técnicas têm sido propostas, podendo-se destacar o uso de Caches. As caches são servidores que replicam o conteúdo, disponibilizando-o em locais mais próximos dos usuários, e consequentemente reduzindo o caminho que os dados precisam percorrer até serem disponibilizados nos clientes [4][5].

As técnicas de cache têm sido amplamente utilizadas no armazenamento de páginas Web tradicionais; no entanto, como os objetos de vídeo têm características bem específicas, é necessário que abordagens mais apropriadas sejam desenvolvidas de modo a prover qualidade adequada e reduzir o consumo de banda. Os algoritmos de cache têm o objetivo de gerenciar o espaço reduzido reservado ao armazenamento dos objetos de modo a manter aqueles que estão sendo mais acessados e assim aumentar a taxa de acertos, consequentemente diminuindo o tráfego na rede e fornecendo melhor qualidade aos usuários finais [6].

Entretanto, esta não é uma tarefa fácil. Os algoritmos de cache trabalham com fatores como popularidade, número total de acessos e probabilidade de acessos futuros, entre outros, a fim de prover o melhor gerenciamento possível do espaço da cache. As estratégias procuram trabalhar com duas métricas principais, o acerto em bytes (*byte hit ratio*), que representa a economia de consumo de banda, e a redução do atraso inicial (*delayed start*). Um dos grandes desafios da área é proporcionar o equilíbrio das duas métricas, pois em geral não é possível melhorá-las ao mesmo tempo [7].

Este trabalho analisa variações e combinações de técnicas de cache bem conhecidas na literatura, utilizando traces reais de objetos do YouTube. Para alguns tamanhos de cache, verifica-se que é possível melhorar o acerto em bytes em 65%, modificando a estratégia de segmentação de uma técnica tradicional a fim de liberar espaços maiores na cache a cada objeto retirado. A pesquisa encontra-se em andamento e tem o objetivo maior de propor uma nova técnica que contribua de forma balanceada para aumentar a taxa de acertos em bytes e, concomitantemente, reduzir o atraso inicial.

2. TRABALHOS RELACIONADOS

As estratégias de cache para vídeo são em grande parte baseadas em um trabalho seminal onde Townsley et al. [8] propõem a técnica de particionar os objetos em duas partes, prefixo e restante do objeto, conservando na cache os prefixos dos objetos mais populares (*Prefix Caching*) Na próxima requisição para o mesmo objeto, a parte inicial é transmitida ao usuário enquanto o restante do vídeo é obtido do servidor principal. Grande parte das técnicas usadas até hoje estendem ou modificam a abordagem de prefixo.

Miao e Ortega [9] discutem as vantagens de armazenar outras partes importantes dos objetos, além dos segmentos iniciais, se houver espaço disponível na cache. A seleção dos segmentos considera o tamanho do buffer do usuário e as propriedades dos objetos de vídeo. Na abordagem conhecida por *Exponential Caching ou Pyramid* [10] os autores reservam uma área inicial da cache para os segmentos iniciais dos objetos e aplicam uma segmentação variável ao restante do vídeo. O tamanho do segmento aumenta exponencialmente, e o i-ésimo segmento tem o tamanho de 2^{i-1}. Esta técnica diminui o atraso inicial.

Uma importante abordagem conhecida como *Lazy* foi proposta por Chen et al. [11]. Neste método, todos os objetos são inseridos completamente no cache na primeira requisição, e a decisão sobre o tamanho dos segmentos é adiada para o momento em que um objeto precisa ser removido da cache. *Lazy* utiliza uma função para atribuir importância aos objetos, a qual tenta prever os acessos futuros aos objetos baseada na freqüência e duração dos acessos já realizados. Alcança um bom desempenho em termos de acertos em bytes (*byte hit*) e consequente diminuição do consumo de banda, e é uma das abordagens consideradas importantes para gerenciamento de caches. A função utilizada pelo algoritmo Lazy está descrita na Equação 1.

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

Equação 1 – Função de prioridade da abordagem Lazy

n = Número de acessos do objeto

T_c = Timestamp atual

T_1 = Timestamp do primeiro acesso ao objeto

T_r = Timestamp do último acesso ao objeto

$\frac{n}{(T_c - T_r)}$ = Número médio de acessos no intervalo de tempo

$MIN\left\{1, \frac{\frac{T_r - T_1}{n}}{T_c - T_r}\right\}$ = Probabilidade de acessos futuros

3. METODOLOGIA

Os dados reais usados nas simulações foram obtidos a partir de uma coleta de dados de acessos ao serviço YouTube realizada por 0, em um roteador localizado entre uma universidade e a Internet no período de Junho de 2007 a Maio de 2008. Os traces contém registros das requisições dos clientes e dos objetos de vídeo solicitados.

A fim de corroborar nossos resultados, utilizamos dois traces, o Trace A com um menor número de requisições (18.495) e o trace B com um número bem maior de requisições (60.318). O conjunto de objetos requisitados em cada trace também varia. A Tabela 1 mostra as características dos traces.

Tabela 1. Traces do YouTube usados nos experimentos

Dados YouTube	Trace A 012908.24	Trace B 031208
Número de requisições	18495	60318
Número de vídeos	13325	37245

Para analisar a efetividade dos algoritmos, foi escrito um simulador em Python. O simulador registra, a cada requisição, a quantidade de objetos atualmente no cache, e a quantidade total de espaço utilizada para guardar esses objetos, além de atributos dos objetos como a frequencia e o timestamp das requisições. A Figura 1 mostra a configuração utilizada nas simulações.

Considera-se que o servidor de cache (*Proxy cache*) encontra-se na mesma rede local que os clientes e, dessa forma, o acesso do clientes à cache é rápido e sujeito a poucas variações. Dessa

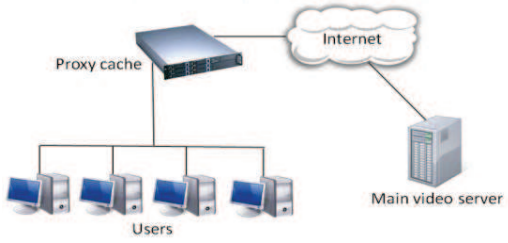

Figura 1 - Estrutura do simulador

forma, o atraso deste acesso local é desprezado.

Nas simulações considera-se que o servidor principal tem capacidade de atender a todas as requisições recebidas. Por outro lado, entre o servidor de cache e o servidor principal (*Main video server*) considera-se que existe uma latência considerável que pode gerar atrasos na transmissão, provocando perda na qualidade percebida pelos usuários. Assim, quando uma requisição chega ao servidor de cache e o vídeo não encontra-se armazenado ainda, contabiliza-se uma ocorrência de atraso inicial, pois será necessário que o usuário espere enquanto o conteúdo é buscado pela cache e armazenado no servidor principal.

Utilizam-se duas métricas principais para medir a eficiência das estratégias de cache, que são a taxa de acertos em bytes (*byte hit ratio*) e o atraso inicial (*start up delay*). A taxa de acertos em bytes é a razão da quantidade de bytes requisitada pela quantidade encontrada na cache. O atraso inicial contabiliza a quantidade de objetos que não foram encontrados na cache. Enquanto a taxa de acerto em bytes deve ser maximizada, consequentemente reduzindo o consumo de banda na rede, o atraso inicial precisa ser minimizado, melhorando a qualidade de experiência do usuário (QoE). Em geral, as técnicas de cache ou diminuem o consumo de banda ou melhoram a experiência do usuário, mas dificilmente atingem os dois objetivos simultaneamente.

As abordagens de cache são basicamente divididas em (a) estratégia de segmentação e (b) estratégia de substituição dos objetos. Nossos experimentos combinam as estratégias de (a) e (b) de diferentes abordagens visando obter melhores resultados. As estratégias de cache avaliadas nos nossos experimentos incluem

modificações e combinações das técnicas Lazy e Exponencial, descritas na seção 2.

Os experimentos medem a eficiência da técnica em função do tamanho da cache. O tamanho da cache é uma variação do tamanho total dos objetos presentes no trace, isto é, o tamanho total dos objetos que poderiam ser requisitados. Esta técnica de avaliação é frequentemente utilizada em experimentos de cache [7][8][9][10].

A técnica Lazy é bem sucedida em reduzir o consumo de banda, enquanto que a abordagem Exponencial reduz a quantidade de vídeos com atraso inicial Por questões de espaço, mostramos apenas as avaliações com a técnica Lazy, nas quais obtivemos os melhores resultados até o momento.

4. RESULTADOS PRELIMINARES

4.1 Avaliação do Acerto em Bytes

A Figura 2 mostra a taxa de acerto em bytes para o trace A. É apresentado o desempenho da estratégia de segmentação Lazy combinada com duas abordagens para a substituição de objetos na cache, a própria Lazy e a variação LRU.

A abordagem Lazy utiliza a fórmula mostrada em (1) para atribuir importância aos objetos que estão no cache, e os objetos menos importantes são excluídos primeiro. A abordagem LRU retira do cache os objetos referenciados há mais tempo, conservando na cache aqueles objetos requisitados mais recentemente.

É possível perceber que, para caches pequenas (até 10% do tamanho total dos objetos) a abordagem que utiliza LRU consegue melhorar o acerto em bytes. Quando a cache é bem pequena (2,5%), verifica-se que a segmentação Lazy combinada com a substituição LRU é capaz de melhorar o acerto em bytes em 36%. Isto representa uma grande economia de banda quando o volume de tráfego é alto, como no caso do tráfego de vídeos de serviços como o YouTube.

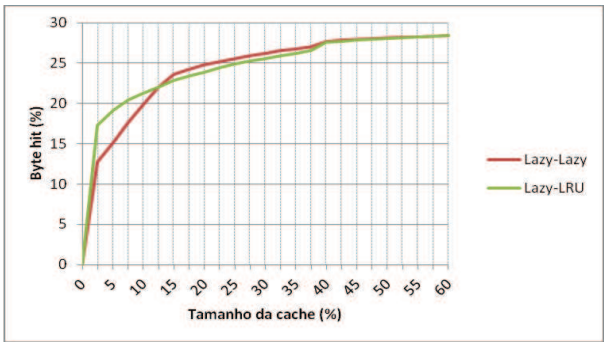

Figura 2 - Acerto em bytes trace A

Outro importante aspecto a destacar é a questão do melhor desempenho quando a cache é pequena. A variação do tamanho da cache nos experimentos é feita em relação ao tamanho total de objetos únicos do trace (metodologia adotada em inúmeros estudos de cache para vídeo). Sabe-se que a quantidade de objetos únicos nos traces do YouTube é muito grande, e que apenas cerca de 25% destes objetos são requisitados mais de uma vez. Dessa forma, uma cache pequena corresponde na maioria das vezes à realidade do tamanho das caches usadas na prática, já que é impossível variar o tamanho da cache em função do número de objetos de um serviço do YouTube. O dimensionamento inicial pode até ser feito com base no número de objetos, mas o crescimento da cache em geral não acompanha o crescimento do número de vídeos disponíveis no sistema. Assim, obter bons resultados para caches pequenas torna-se importante.

A Figura 3 mostra a mesma avaliação usando um trace com um número muito maior de requisições, o trace B (60.000). Fica claro que, quando o número de requisições aumenta, a diferença de desempenho entre as técnicas se torna maior. Para uma cache de tamanho 2.5% a técnica Lazy combinada com a LRU é capaz de obter um desempenho 65% melhor do que a técnica Lazy original.

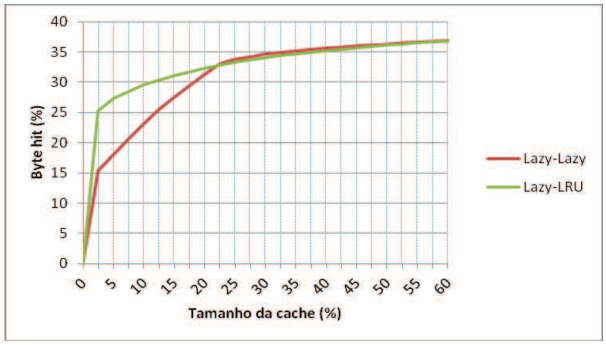

Figura 3 - Acerto em bytes trace B

4.2 Avaliação do Atraso Inicial

Como relatado em pesquisas anteriores, a técnica Lazy consegue obter um ótimo desempenho para o acerto em bytes; entretanto, esta técnica não é bem sucedida na redução do atraso inicial. Os nossos experimentos mostraram que a variação que combina a estratégia de segmentação Lazy combinada com a estratégia de substituição de objetos LRU é capaz de reduzir o percentual de objetos com atraso inicial para caches pequenas.

A Figura 4 mostra o desempenho medido pelo atraso inicial para as mesmas abordagens da seção anterior, para o trace A. Ressalta-se que o Eixo Y representa o percentual de vídeos com atraso inicial (aqueles que não foram encontrados na cache no momento da requisição), enquanto o Eixo X mostra a variação do tamanho da cache (Figuras 4 e 5).

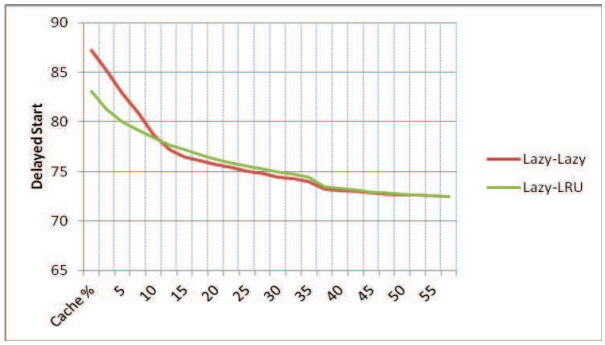

Figura 4 - Atraso inicial trace A

Observa-se que para tamanhos de cache de 2,5% e de 5%, a técnica combinada melhora o desempenho da técnica Lazy em 5%. Considerando-se que esta técnica é bastante complexa e foi muito trabalhada pelos autores, estimamos que este é um bom resultado, significando que um número menor de requisições irão sofrer atraso inicial.

A Figura 5 mostra o resultado da análise do atraso inicial para o trace B. É possível perceber que a técnica combinada Lazy-LRU é capaz de melhorar o desempenho da técnica Lazy para tamanhos

menores de cache. Verifica-se que para um número de requisições maior a diferença de desempenho entre as técnicas também é maior. Para um tamanho de cache de 2.5% a técnica combinada consegue obter um desempenho 9% melhor do que a técnica Lazy.

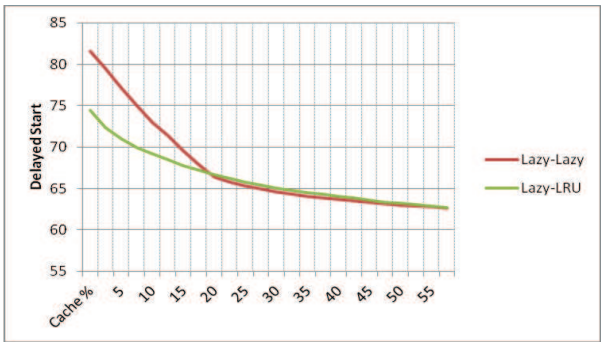

Figura 5 - Atraso inicial trace B

5. CONCLUSÕES

Uma dos principais objetivos do uso de caches é a redução do consumo de banda e do congestionamento na rede, através da colocação de servidores com réplicas de conteúdo em localizações estratégicas da rede. Os algoritmos de cache são extremamente importantes e estão diretamente relacionados com o desempenho das caches tanto em termos de consumo de banda como na redução dos atrasos.

O cenário deste trabalho é a utilização de um servidor de cache (*proxy cache*) na rede local a fim de interceptar as requisições de vídeo para serviços de distribuição de vídeo na Internet como o YouTube. Proxy caches já são utilizados na Internet; entretanto, ainda existe um número reduzido de análises específicas para a distribuição de vídeo.

Este trabalho analisa variações de técnicas de cache conhecidas que apresentam bom desempenho, a fim de melhorar ainda mais os seus resultados. A nossa principal contribuição é conseguir alterar a técnica Lazy, tradicionalmente com ótimo desempenho para o aumento do acerto em bytes, tornando-a ainda melhor para caches pequenas. Além de aumentar o acerto em bytes e, consequentemente, diminuir o consumo de banda, a variação que utiliza LRU como estratégia de substituição na cache é capaz também de reduzir o percentual de vídeos com atraso inicial, proporcionando uma melhoria da qualidade de experiência dos usuários (QoE).

O algoritmo modificado apresenta melhor desempenho para caches menores. Este achado é importante, visto que, na prática, torna-se difícil, redimensionar constantemente a cache em relação ao número de objetos disponível no sistema. Em geral, as caches permanecem com um tamanho fixo por um determinado período de tempo e, quando comparadas ao número de objetos disponíveis no YouTube, são quase sempre pequenas. Sendo assim, concluímos que a técnica pode contribuir significativamente para diminuir o consumo de banda na rede, em serviços como o YouTube, além de reduzir o atraso inicial.

Uma série de estudos estão sendo conduzidos como continuação deste trabalho. Ressalta-se os experimentos com traces sintéticos modelados para avaliar diversas condições de tráfego, nos quais se variam importantes parâmetros da carga de dados como a distribuição de popularidade, o tamanho dos objetos, a quantidade de requisições, a duração das sessões de usuários e o percentual de objetos *cacheable*. Além disso, pretende-se realizar experimentos com tamanho absoluto de cache, a fim de compreender melhor o comportamento dos algoritmos.

6. AGRADECIMENTOS

Nossos agradecimentos ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), por financiar este trabalho através de bolsa de iniciação científica (PIBIC).

7. REFERÊNCIAS

[1] CISCO, “Cisco Visual Networking Index: Forecast and Methodology, 2009-2014”. 2010, June. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf. Acessado em Novembro, 2010.

[2] YouTube. http://youtube.com/ Acessado in January, 2012.

[3] M. Cha , H. Kwak , P Rodriguez , Yong-Yeol Ahn , Sue Moon, “I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System”, Proc. of the 7th IMC 2007, San Diego, California, USA, October 2007.

[4] Hofmann, M., and Beaumont, L. R., “Content Networking: Architecture, Protocols, and Practice”. Morgan Kaufmann Publishers, San Francisco, CA, USA, 2005.

[5] Pathan, A. K., Buya, R., “A Taxonomy of CDNs", Content Delivery Networks, R. Buyya, M. Pathan, and A. Vakali (Eds.), Springer-Verlag, Germany, 2008.

[6] Jayarekha, P., P. C. Air, T. R. Gopalakrishnan, “A Rank Based Replacement Policy for Multimedia Server Cache Using Zipf-Like Law”, Journal of Computing Vol 2, Issue 3, March 2010.

[7] Romano, Sam, ElAarag, H. "Comparison of function based web proxy cache replacement strategies". In Proc. of 12th Intern. Conf. on Performance Eval. of Computer & Telecom. Systems (SPECTS'09), IEEE Press, NJ, USA, 2009.

[8] Sen, S., Rexford, J., and Towsley, D. "Proxy prefix caching for multimedia streams". In Proceedings of IEEE INFOCOM'99. IEEE Computer Society, 1999.

[9] Miao, Z. and Ortega, A. 1999. "Proxy Caching for Efficient Video Services over the Internet". In Proceedings of the 9th International Packet Video Workshop (PVW'99).

[10] Wu, K., Yu, P. S., and Wolf, J. L. “Segment-based proxy caching of multimedia streams”. In Proc. of the 10th Intern. Conference on WWW, Hong Kong, 2001.

[11] Chen, S., Wang, H., Zhang, X., Shen, B., and Wee, S., “Segment-Based Proxy Caching for Internet Streaming Media Delivery”. IEEE MultiMedia 12, July 2005.

[12] Zhou, R., Khemmarat, S., Gao,L., "The Impact of YouTube Recommendation System on Video Views", in Proc. of Internet Measurement Conference (IMC), 2010.

Zink, M., Suh, K., Gu, Y. and Kurose, J., "Watch Global Cache Local: YouTube Network Traces at a Campus Network - Measurements and Implications", IEEE MMCN, 2008.