A Comparison of Block-based and Clip-based Cooperative Caching Techniques for Streaming Media in Wireless Home Networks*

Shahram Ghandeharizadeh, Shahin Shayandeh
Computer Science Department
University of Southern Califronia
Los Angeles, CA 90089
shahram@usc.edu, shayande@usc.edu

Abstract

Wireless home networks are widely deployed due to their low cost, ease of installation, and plug-and-play capabilities with consumer electronic devices. Participating devices may cache continuous media (audio and video clips) in order to reduce the demand for outside-the-home network resources and enhance the average delay incurred from when a user references a clip to the onset of its display (startup latency). In this paper, we focus on a home network consisting of a handful of devices configured with a mass storage device to cache data. A cooperative caching technique may manage the available cache space at the granularity of either a clip or individual blocks of a clip. The primary contribution of this paper is to evaluate these two alternatives using realistic specifications of a wireless home network, identifying factors that enable one to outperform the other.

1 Introduction

Wireless home networks are deployed widely due to their ease of installation and economical prices. A typical wireless home network may consist of a handful of devices such as PCs, laptops, TV set-top boxes such as Apple TV, and consumer electronic devices such as DVRs, audio tuners, video game consoles, and Plasma/LCD TVs. The latter is an interesting convergence of electronic devices with wireless technology to enable consumers to display digital content from Internet sites such as YouTube and Hulu to name a few.

With inexpensive mass storage devices (9 cents per gigabyte of magnetic disk storage), a device might be configured with substantial amount of storage to cache data. Algorithms that control the content of these caches are important for several reasons. First, they enhance startup latency as a key quality of service metric, defined as the delay incurred from when a user references a clip to the onset of its display. If the referenced content is cached on a device, the device may display this clip immediately, minimizing startup latency close to zero. Otherwise, the content must be delivered using the wireless network. With continuous media, audio and video clips, one may stream a clip and overlap its display at a device with its retrieval across the network. While streaming minimizes startup latency, it does not reduce it close to zero as with caching. Second,

*A shorter version of this paper appeared in the Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications (WASA2009), Boston, Massachusetts, August 16-18, 2009.

servicing requests from a local cache minimizes the demand for the infrastructure (network bandwidth and remote servers) outside of the home, freeing these shared resources to service other requests.

The applications used by members of a household define the working set of the household, denoted WS. Examples include user visits to web pages specializing in their area of interest (such as financial) with the same advertisement clips, children shows watched by the younger members of the household over and over again, short video clips such as those found on YouTube and social networking sites, recent wedding and birthday clips watched repeatedly, and others[1].

In this study, we focus on Domical [12], a cooperative caching algorithm for wireless devices in a home network. Domical [12] abstracts the available storage space of a device into three parts named Private, Collaborative, and Elite. The content of the Private portion is dictated by either the user or an application. Domical will not manage or manipulate the Private content. The Collaborative and Elite portions of the storage space constitute the available cache space and are managed by Domical. Domical manages the Elite cache space with the objective to optimize a local metric such as cache hit rate. Candidate algorithms include LRU-K [21], GreedyDual-Size [18] or DYNSimple [10] to name a few. Domical renders the Collaborative cache space of different devices dependent on one another in order to enhance a global metric such as average startup latency. This dependence of collaborative caches does not compromise autonomy of different devices. In particular, if the content watched by the users of different devices is either non-overlapping or have very little overlap, the collaborative cache space of these devices behave as if they are not dependent on one another.

The parameter α [2, 9] dictates what fraction of the available cache space is assigned to the Collaborative and Elite portions. Assuming S_{N_i} dictates the cache space of device N_i, the size of Elite space is $\alpha \times S_{N_i}$. The remainder, $(1-\alpha) \times S_{N_i}$, is the Collaborative space. In [12], we show Domical enhances the average startup latency significantly with $\alpha = 0$ when compared with $\alpha = 1$.

A key intuition behind the design of Domical is to minimize the likelihood of bottleneck links when the available bandwidth is asymmetric. This is applicable to both ad-hoc and infrastructure modes of communication as analyzed by a recent study of six wireless homes in United States and United Kingdom [22]. This study provides the following key insights. First, wireless links in a home are highly asymmetric and heavily influenced by precise node location, transmission power, and encoding rate, rather than physical distance between nodes. It showed many links were unable to utilize the maximum transmission rate of the deployed 802.11 technology. For example, with a US home network deployment operating at 30mW and 11 Mbps rate, all devices observed a bandwidth lower than 5 Mbps. While two devices numbered 5 and 6 could communicate, the bandwidth from device 6 to 5 was almost 3 Mbps while the reverse was 1 Mbps. This asymmetric bandwidth was exaggerated in a few instances where the reverse bandwidth was close to zero. Second, this study shows that coverage and performance is enhanced using a multi-hop topology instead of an infrastructure based deployment, motivating mesh capabilities for consumer electronics for seamless connectivity across the home. Without loss of generality, we assume Domical using a multi-hop, mesh deployment of wireless devices.

Different studies to date have focused on different design decisions for Domical, paying little attention to comparing the physical design and granularity of data placement. For example,the concept of urgency worthiness to manage placement of blocks with domical is studied in [13]. The main contribution of this study is to analyze Domical with two different

[1]While we focus on streaming media, our techniques apply to other large data items such as images.

granularities of data replacement: clip-based and block based. They are named Domical Clip-based Replacement (DCR) and Domical Block-based Replacement (DBR). We quantify their performance tradeoffs showing the following key lessons. First, when the total cache space in the home network is significantly smaller than the working set size of the clips referenced by the users of the home, caching at the granularity of a clip is a superior alternative because it minimizes the likelihood of bottleneck links in the home networks. The working set size is defined as the collection of clips that are repeatedly referenced by a household over a period of time, e.g., shows watched repeatedly by the children. With a small working set size and cooperative nodes (small α values), DBR with caching at the granularity of blocks enhances startup latency. If the members of a household do not watch the same content repeatedly or have different interests then the working set size is either very large or infinite, motivating the use of DCR with clips as the granularity of data placement. Second, with a small working set size and α values approximating 0 (cooperative caching), the system should use DBR and cache data at the granularity of a block. Third, we show these observations hold true with different access distributions to data and a variety of in-home network characteristics.

The rest of this paper is organized as follows. Section 2 surveys research related to our replacement techniques. In Section 3, we briefly describe the Domical clip replacement technique, DCR. Section 3.1 extends modifications to implement the DBR technique. We evaluate these alternatives and present our observations in Section 4. Brief conclusions and future research directions are presented in Section 5.

2 Related Work

Our proposed replacement techniques complement the vast body of work on proxy web servers for streaming media, see [19] for a survey. Below, we survey cooperative replacement techniques found in the prior literature that strive to enhance average startup latency.

A group of proxies may cooperate to increase the aggregate cache space, balance load, and improve system scalability [1, 4, 11, 12]. MiddleMan [1] segments clips into equi-sized segments and employs a centralized coordinator to balance load evenly across proxies. Cont-Coop [11] and Domical [12] cache data with the objective to balance the load of streaming media across the asymmetric wireless connections of a home network. While these techniques impose more work on a device with abundant network bandwidth in order to enhance startup latency, MiddleMan strives to balance the load across proxy servers. Both Cont-Coop and Domical cache data at the granularity of a clip with Domical outperforming Cont-Coop [12]. Our study is novel because we quantify the tradeoffs associated with clip and block replacement techniques with Domical. Moreover, we show the effectiveness of the urgency-worthiness metric when choosing victims. One may implement our proposed techniques using the centralized coordinators of MiddleMan [1] and Middleware of [17], and proxy cluster of Silo [4].

Cooperative proxy caching for streaming media has been explored in overlay networks [16, 20]. COPACC [16] assumes an architecture of a two level cache: at both proxies and clients. It partitions a clip into three segments: Prefix, Prefix-of-suffix, and Suffix. While proxies cache the prefix of a clip, clients cache the Prefix-of-suffix. This layout is shown to minimize the startup latency and facilitate multicast delivery with dynamic clients. OCS [20] assumes a similar architecture as COPACC and presents a caching technique at the granularity of a clip. It specifically notes that caching at the granularity of a block provides inferior average startup latency than clip-based caching (without either presenting comprehensive results or elaborating on the reasons). Our study is different because, in our architecture, the proxy and client caches are the same and one.

Finally, a cooperative replacement technique for home gateways in a neighborhood is detailed in [2]. Its replacement policy, named DCOORD, is at the granularity of a clip and considers both local and remote hits when choosing victims. In [9], we show Domical enhances startup latency for the home network when compared with DCOORD. DCOORD is appropriate for outside the home network because it is decentralized and can scale to a large number of nodes.

3 Domical Caching framework

Domical [12] constructs dependencies between the caches of different devices with the objective to prevent the possibility of a wireless link from becoming a bottleneck. A device may contribute a fraction of its available cache space for cooperation with other devices. This is the collaborative space and data items occupying it might be either clips or blocks of different clips. To simplify the discussion and without loss of generality, in the following we assume the granularity of cached data is a clip. In Section 3.1, we describe modifications to support block caching.

The cache space of N_i is made dependent[2] on the state of caches managed by devices N_0, N_1, ..., N_{i-1}. This impacts how N_i victimizes clips as follows. N_i constructs a list of those clips with a replica in the cache space of N_0, N_1, ..., N_{i-1}. These are named common clips. The remaining clips are named rare clips. Domical victims the common clips with the lowest byte-hit rate first. If this does not release sufficient space for the incoming clip X and there are no common clips left, then Domical victimizes the rare clips starting with those that have the lowest byte-hit rate.

One may implement Domical in different ways. Assuming devices in the home network are single user devices with abundant amount of processing capability, a simple implementation would be as follows. When a clip X is admitted into the cache and Domical has insufficient space to store X, it victimizes existing clips as follows. First, it sorts[3] those clips occupying N_i's cache in descending order using their byte-hit rate, $\frac{f_j}{S_j}$, where f_j is the frequency of access to object j and S_j is the size of that object. Next, it marks the first k clips with a total size exceeding the size of the Elite space of the cache. These clips maximize the byte-hit rate of the Elite space. The remaining clips occupy the Collaborative space. Domical identifies the common and rare clips in the Collaborative space, removing as many common clips with a low byte-hit rate to provide space for the incoming clip X. Once common clips are exhausted, it deletes the rare clips starting with the ones that have the lowest byte-hit rate.

N_i may identify common and rare clips by listening on the traffic generated by different devices referencing different clips. Moreover, it may periodically exchange the identity of clips occupying its cache with other devices.

Domical computes dependencies between shared caches of different nodes using the bandwidth contention ratio metric [12]. This metric estimates the amount of imbalance across the network links when N_i streams a clip to every other node in the network. Intuitively, the node with the smallest bandwidth contention value (N_{min}) results in the lowest imbalance, avoiding formation of hot spots and bottleneck links. N_{min} manages its entire cache space (both Elite and Collaborative)

[2]Towards the end of this section, we describe how Domical computes dependencies between different nodes.
[3]Sorting is not appropriate for devices with multiple simultaneous users due to its computational overhead.

using a greedy caching technique. The state of Collaborative space of every other node in the cooperative group depends on N_{min}. Moreover, the remaining nodes are sorted in ascending order using their bandwidth contention ratio metric. The state of the Collaborative space of a node i is dependent on the cache of every other node (both Elite and Collaborative space) before it and N_{min}.

3.1 Domical Block-based Replacement (DBR)

To change the granularity of data placement from a clip to a block, we modify the local replacement policy used by each device to manage its Elite and Collaborative cache space. Instead of byte hit rate, each device employs the urgency worthiness metric [13] to sort its list of blocks to choose victims. This metric is defined as $\frac{f_i}{B_{Display_i} \times d_{i,j}}$ where $B_{Display_i}$ is the bandwidth required to display clip i and $d_{i,j}$ denotes the display time of j^{th} block of clip i relative to its start. The intuition behind this metric is to cache sufficient number of the first few of blocks of a clip (termed prefetch portion) to enable a client to display these blocks while streaming the remaining blocks of a clip without the client starving for data. This complements streaming to minimize observed startup latency close to zero, similar to a clip being cached in its entirety. The dependencies between Collaborative cache of different devices minimizes the likelihood of duplicate prefetch portions cached across multiple devices.

The use of frequency of access enables many (if not all) blocks of the most popular clip to evict the first few blocks of the least popular clips. This is shown in Section 4.2, see the discussion of Figure 1.

Section 4 compares Domical variations of clip and block replacement techniques with one another.

4 Comparison

In this section, we present the simulation model used to compare the alternative techniques. Subsequently, we present the key observations one at a time.

4.1 Simulation Model

We developed a simulation model to compare the alternative block-based and clip-based caching techniques. This model represents a device as a node. A node is configured with a fixed amount of storage and sets aside a portion of it as cache. Let S_T denote the sum of the size of Elite and Collaborative caches contributed by $\mathcal{N}$ nodes in a cooperative group, and S_{WS} to denote the size of the working set of our target application. A node may employ either DCR or DBR to manage the available cache space.

We assume a repository of constant bit rate (CBR) video clips. The display bandwidth requirement of each clip is 4 Mbps. We assume the working set (WS) consists of 864 clips. 864 is an arbitrary number of clips used to generate the test database. We have examined other values and have observed the results of Section 4.2 to hold true for different $\frac{S_T}{S_{WS}}$ values. We examined a variety of media and clip mixes that result in different bandwidth requirements. In all cases, We observed the same lessons. To simplify discussion and without loss of generality, for the rest of this section, we assume all clips have a display time of 30 minutes and each clip is 0.9 GB in size.

We use a Zipf-like distribution [3] with mean of μ to generate requests for different clips. A larger value for exponent μ makes the distribution more skewed by increasing the frequency of access to a few popular clips. A small μ value makes the distribution more uniform. Zipf has been used to model the distribution of web page requests [14, 3, 23], and sale of movie tickets[4] in the United States [7].

One node in the system is designated to admit requests in the network by reserving link bandwidth on behalf of a stream. This node, denoted N_{admit}, implements the Ford-Fulkerson algorithm [6] to reserve link bandwidths. When there are multiple paths available, N_{admit} chooses the path to minimize startup latency.

The simulator conducts ten thousand rounds. In each round, we select nodes one at a time in a round-robbin manner, ensuring that every node has a chance to be the first to stream a clip in the network. A node (say N_1) references a clip using a random number generator conditioned by the assumed Zipf-like distribution. With DCR, if this clip resides in N_1's local storage then its display incurs a zero startup latency. Otherwise, N_1 identifies those nodes containing its referenced clips, termed candidate servers. Next, it contacts N_{admit} to reserve a path from one of the candidate servers. N_{admit} provides N_1 with the amount of reserved bandwidth, the paths it must utilize, and how long it must wait prior to streaming the clip. This delay is the incurred startup latency.

With DBR, when N_1 references a clip X, the startup latency is zero if all blocks of X are in N_1's cache. Otherwise, N_1 identifies the missing blocks of X, identifies the candidate servers for each blocks, contacts N_{admit} to reserve a path from one of these servers for each block. N_{admit} provides N_1 with the amount of reserved bandwidth, the path it must utilize, and how long it must wait for each block. N_1 uses this information along with the identity of its cached blocks to compute the incurred startup latency.

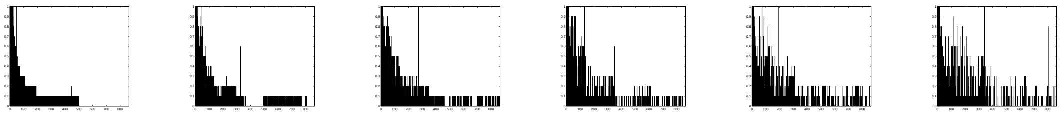
In each iteration, we measure the following parameters local to a node: startup latency, byte-hit and cache hit rates, and unutilized cache space. In addition, we measure the following global parameters: average startup latency, average hop distance to stream a clip, and average amount of bytes transmitted across the network. To minimize the impact of a cold-start on the observed parameters, the most popular clips are stored in the cache of every device at the start of each experiment. The number of such clips changes for different $\frac{S_T}{S_{WS}}$ values. Section 4.2 shows how this starting state evolves with DCR and DBR.

4.2 Performance Results

We focus on a realistic wireless home network corresponding to a deployment of six nodes employing 802.11a networking cards in a British household [22] with asymmetric link bandwidths. Domical constructs the same dependencies between the nodes with both DCR and DBR. Let d^i indicate the i^{th} dependency group. Assignment of node N_j to d^i indicates that state of cache in N_j depends on the state of caches in dependency groups d^0 to d^{i-1}. Both DCR and DBR assign Node 4 to d^0. The remaining nodes are assigned as follows: Nodes 0, 2, 1, 3, and 5 to d^1, d^2, d^3, d^4, d^5, respectively. The key difference between the two techniques is the realized placement of data across devices.

Figure 2 shows the percentage improvement in average startup latency provided by DBR when compared with DCR for

[4]In [7], a Zipf-like distribution is defined as $\frac{X}{i^{(1-\omega)}}$ where ω is 0.27. In this paper, μ equals $1 - \omega$. To be consistent with [7], we analyze 0.73 as a possible value for μ in Section 4.



a) Node 4 b) Node 0 c) Node 2 d) Node 1 e) Node 3 f) Node 5

Figure 1. Distribution of blocks across devices in the network, DBR, $\alpha = 0.7$**,** $\mu = 0.73$, $\frac{S_T}{S_{WS}} = 0.75$

different α values and $\frac{S_T}{S_{WS}}$ ratios. Assuming δ is the average startup latency observed with each technique, percentage improvement is defined as $100 \times \frac{\delta(DCR)-\delta(DBR)}{\delta(DCR)}$. This is shown on the y-axis of Figure 2 as a function of different α values. This figure highlights two key trends. First, with high $\frac{S_T}{S_{WS}}$ values, small working set size, the percentage improvement observed by DBR when compared with DCR diminishes as α approaches 1. Second, the percentage improvement observed by DBR diminishes with lower $\frac{S_T}{S_{WS}}$ values. With $\frac{S_T}{S_{WS}} = 0.1$, DCR is superior to DBR. We explain each trend in turn.

With α=1, all cache space is Elite and DCR outperforms DBR with alternative $\frac{S_T}{S_{WS}}$ ratios. The explanation for this is as follows. With DBR, each node caches the first few blocks of the most popular clips, causing the placement of blocks across different nodes to be identical to Node 4 in Figure 1.a. The x-axis of this figure is the clip id in descending order of popularity. The y-axis shows the fraction of each cached clip. When a node references a clip with missing blocks, it must stream the remaining blocks of its referenced clip from the infrastructure outside the home. With all nodes referencing a clip, this results in bottleneck links and increased response time. DCR, on the other hand, caches the most popular clip in its entirety onto each node. Now, when a node observes a cache hit for its referenced clip, it services this clip from its local storage, freeing the home network to stream a clip for another node that may have observed a cache miss.

With $\alpha \leq 0.7$ and $\frac{S_T}{S_{WS}} \geq 0.75$, DBR outperforms DCR because different nodes in the network can cache the first few blocks of different clips, see Figure 1. This enables different nodes to act as servers for blocks referenced by another node enhancing startup latency when compared with DCR.

As we reduce the value of $\frac{S_T}{S_{WS}}$ from 1 to 0.1, see Figure 2, the percentage improvement observed by DBR diminishes. When the total cache space of the home network is 10% of the repository ($\frac{S_T}{S_{WS}} = 0.1$), DCR is superior to DBR regardless of the degree of cooperation between different nodes, i.e., all α values shown on the the x-axis of Figure 2. With a small amount of cache, the first few blocks of popular clips compete for the available cache space of different nodes, displacing the remaining blocks of these and other clips. Similar to prior discussion, DBR is now forced to stream the missing blocks of a referenced clip from the home gateway. When all nodes service their requests in this manner, this results in formation of hot spots and bottlenecks. While DCR must also stream a larger number of its requests using the home gateway, it benefits from having a clip in its entirety, enabling a device to observe a low response time every time it observes a cache hit.

With fully cooperative nodes ($\alpha = 0$) and small working set size ($\frac{S_T}{S_{WS}} \geq 0.5$), DBR outperforms DCR for three reasons. First, some devices contain the prefetch portion of a clip, enabling them to overlap the display of this portion with the retrieval of the rest of the clips, minimizing the incurred startup latency. Second, a device may retrieve the blocks of a clip in different order from different nodes, maximizing the utilization of network bandwidth. Third, block replacement has a higher percentage of the repository (as measured in blocks, see below for a discussion of unique clips) cached across

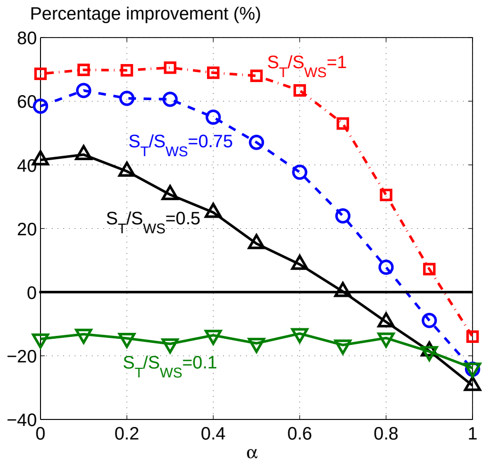


Figure 2. DBR Vs. DCR, $\mu = 0.73$.

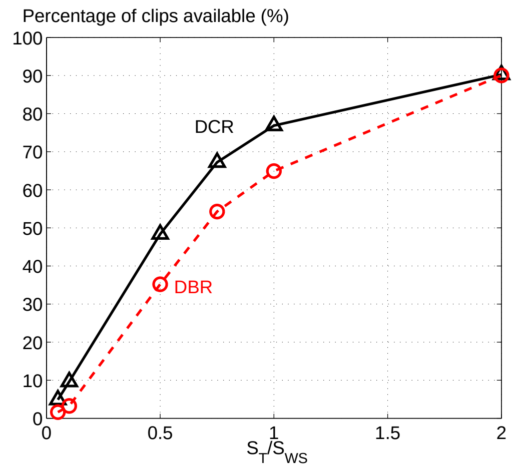


Figure 3. Data availability, $\mu = 0.73$.

devices; typically 10% higher. This enhances the ability of different nodes to stream blocks amongst each other, minimizing dependency of the network on the wireless connection of N_5 as the intermediatory to the home gateway and the outside infrastructure.

While the percentage of cached data is higher with DBR, a clip may not cache its entirety across the caches. These partially cached clips may not be displayed when the home network is disconnected from the outside infrastructure. Figure 3 shows the percentage of clips available with DCR and DBR. In general, DCR provides 10 to 15% higher clip availability when compared with DBR.

Same trends and observations hold true when we vary mean of the distribution of access and available bandwidth of the wireless network.

5 Conclusion and future research directions

The primary contribution of this study is to analyze different granularity of data placement with the Domical cooperative caching technique in the context of a wireless home network. Block-based caching is appropriate when the working set size of an application is relatively small, enabling different devices to service blocks missing from each other's cache. If the working set size of an application is anticipated to be more than ten times the total cache size of the home network, caching at the granularity of a clip is a better alternative because it enhances the average startup latency.

We intend to extend this study in several ways. First, we plan to investigate impact of efficient data delivery algorithms such as patching [15, 5] and stream merging [8] with the clip and block replacement techniques. Second, we are exploring a dynamic version of Domical that switches from one granularity of data placement (a block) to another (a clip) based on the observations reported in Section 4.2. To elaborate, these observations identify a few parameters such as the working set size and the available cache space as the basis for deciding the data caching granularity. Assuming a fixed physical block size, a device may switch to clip-based caching by managing its cache using a logical block size, and increasing the size of a logical block to equal the total size of the blocks that constitute a clip. A device may monitor the key parameters and use the average startup latency to decide the size of a logical block. Third, we will conduct a performance analysis of Domical under Dynamic network situations such as nodes joining and leaving the cooperative group due to either mobility or being turned off.

References

[1] S. Acharya and B. Smith. MiddleMan: A Video Caching Proxy Server. In *Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, pages 162–172, June 2000.

[2] H. Bahn. A Shared Cache Solution for the Home Internet Gateway. *IEEE Transactions on Consumer Electronics*, 50(1):168–172, February 2004.

[3] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web Caching and Zipf-like Distributions: Evidence and Implications. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, pages 126–134, 1999.

[4] Y. Chae, K. Guo, M. Buddhikot, S. Suri, and E. Zegura. Silo, Rainbow, and Caching Token: Schemes for Scalable, Fault Tolerant Stream Caching. *IEEE Journal on Selected Areas in Communications*, 20(7):1328–1344, Sep 2002.

[5] S. Chen, B. Shen, Y. Yan, and X. Zhang. Buffer Sharing for Proxy Caching of Streaming Sessions. In *Proceedings of the International World Wide Web Conference*, May 2003.

[6] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, editors. *Introduction to Algorithms*, chapter 26.2. MIT Press, 2001.

[7] A. Dan, D. Sitaram, and P. Shahabuddin. Scheduling Policies for an On-Demand Video Server with Batching. In *Proceedings of the ACM Multimedia Conference*, pages 15–23, October 1994.

[8] D. Eager, M. Vernon, and J. Zahorjan. Optimal and Efficient Merging Schedules for Video-on-Demand Servers. In *Proceedings of the ACM Multimedia Conference*, pages 199–202, October 1999.

[9] S. Ghandeharizadeh and S. Shayandeh. An Analysis of Two Cooperative Caching Techniques for Streaming Media in Residential Neighborhoods. In *Submitted for consideration*.

[10] S. Ghandeharizadeh and S. Shayandeh. Greedy cache management techniques for mobile devices. In *Proceedings of the IEEE International Conference on Data Engineering*, pages 39–48, April 2007.

[11] S. Ghandeharizadeh and S. Shayandeh. Cooperative caching techniques for continuous media in wireless home networks. In *Proceedings of the international conference on Ambient media and systems (Ambi-Sys)*, pages 1–8. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), February 2008.

[12] S. Ghandeharizadeh and S. Shayandeh. Domical cooperative caching: A novel caching technique for streaming media in wireless home networks. In *Proceedings of International Conference on Software Engineering and Data Engineering (SEDE)*, pages 274–279, June 2008.

[13] S. Ghandeharizadeh and S. Shayandeh. An evaluation of two domical block replacement techniques for streaming media in wireless home networks. In *Proceedings of the IEEE International Symposium on Multimedia (ISM)*, pages 372–377, Dec. 2008.

[14] S. Glassman. A caching relay for the world wide web. *Computer Networks and ISDN Systems*, 27(2):165–173, 1994.

[15] K. Hua, Y. Cai, and S. Sheu. Patching: A Multicast Technique for True Video-on-Demand Services. In *Proceedings of the ACM Multimedia Conference*, pages 191–200, September 1998.

[16] A. T. S. Ip, J. Liu, and J. C.-S. Lui. COPACC: An Architecture of Cooperative Proxy-Client Caching System for On-Demand Media Streaming. *IEEE Trans. Parallel Distrib. Syst.*, 18(1):70–83, 2007.

[17] W. J. Jeon and K. Nahrstedt. QoS-aware Middleware Support for Collaborative Multimedia Streaming and Caching Service. *Microprocessors and Microsystems*, 27(2):65–72, 2003.

[18] S. Jin and A. Bestavros. Popularity-Aware GreedyDual-Size Web Proxy Caching Algorithms. In *Proceedings of the 20th International Conference on Distributed Computing Systems (ICDCS)*, pages 254–261, April 2000.

[19] J. Liu and J. Xu. Proxy caching for media streaming over the internet. *Communications Magazine, IEEE*, 42(8):88–94, Aug. 2004.

[20] T. M, W. Tavanapong, and W. Putthividhya. Ocs: An effective caching scheme for video streaming on overlay networks. *Multimedia Tools Appl.*, 34(1):25–56, 2007.

[21] E. J. O'Neil, P. E. O'Neil, and G. Weikum. The LRU-K Page Replacement Algorithm for Database Disk Buffering. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 413–417, 1993.

[22] K. Papagiannaki, M. Yarvis, and W. S. Conner. Experimental characterization of home wireless networks and design implications. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–13, April 2006.

[23] A. Wolman, M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy. On the Scale and Performance of Cooperative Web Proxy Caching. *SIGOPS Opereting Systems Review*, 33(5):16–31, April 1999.