Cache Replacement 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 improve 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. A device may manage its cache at the granularity of either a clip or individual blocks of a clip by employing either a clip-based or a block-based cache replacement technique. It may employ a technique in either stand-alone mode to enhance a local metric such as cache hit rate or make its state dependent on other devices in the home network to enhance a global metric such as average startup latency. These two variants are named greedy and cooperative, respectively. The primary contribution of this paper is to evaluate these alternatives using realistic specifications of a wireless home network. Our key finding is as follows. Our proposed cooperative block-based technique enhances startup latency when compared with clip-based alternatives. With multiple devices where each device acts greedy, clip-based caching is superior. In addition, we show our proposed block-based replacement policy materializes appropriate fraction of each clip across devices dynamically, eliminating the need to compute and place prefetch portions statically.

1 Introduction

Continuous media, audio and video clips, have become pervasive on the Internet. It is not uncommon to visit a web page with either background music or a video clip advertisement. Similarly, a growing number of subscribers to social networking sites such as myspace use a video clip to introduce their profile. A member may personalize their portals by including an

audio clip to greet their visitors. To reduce the load on the backbone infrastructure, one may employ caching at the edges of the network [38]. In this paper, we focus on wireless home networks because they are widely deployed due to their low cost and ease of installation [29]. A typical home network may consist of an access point, several PCs and laptops, and one or more electronic devices such as Apple TV that synchronizes with a PC using its wireless networking card. Devices might be configured with a mass storage device and set aside a fraction of their storage to cache content. For example, at the time of this writing, Apple TV is configured with either 40 or 160 GB of disk storage and may cache clips.

When a user references a clip X using a device, the device may find X in its local cache. This cache hit enables the device to service the request immediately. Otherwise, the device incurs a cache miss and may stream X from two possible sources: 1) a remote server outside of the home infrastructure, and 2) another device in the home network with a cached copy of X. With the later, a device not only acts as a client to display a clip, it may act as a proxy server to stream a clip from its cache to another device in the household. This localizes network traffic to the home infrastructure, freeing the infrastructure outside of the home to service other requests. This form of cooperation enhances the following quality of service metrics: First, average startup latency defined as the average delay incurred from when a user requests a clip until the start of display. Second, availability of data defined as what fraction of requests are serviced when either the whole or one component of the infrastructure outside the home becomes unavailable. The whole infrastructure might become unavailable due to bad weather conditions that disrupt service to the home.

A replacement technique is a key component of a caching framework. It dictates what objects occupy the cache of a device. Figure 1 shows a taxonomy of replacement techniques. As suggested by their names, clip-based and block-based techniques manage the cache of a device at the granularity of a clip and blocks of different clips, respectively. Each technique might be deployed in either a greedy or a cooperative mode. With a greedy deployment, a device manages its cache independent of other devices, striving to enhance a local metric such as the cache hit rate or its incurred startup latency. When deployed in cooperative mode, a device may render the state of its cache dependent on other devices in the network to enhance a global metric such as the average startup latency incurred by all device.

While the taxonomy of Figure 1 is straightforward, the number of possible ways to design and implement greedy and cooperative versions of clip-based and block-based techniques are many, see Section 2. During the past four years, we have studied these alternatives and proposed new techniques. Some of our work has been analytical [17, 1] while others have been simulation based [14, 15, 16]. These prior publications have focused on different parts of the taxonomy shown in Figure 1. Their culmination is this paper which details the entire taxonomy and characterizes the tradeoffs associated with alternative design decisions. Our **contributions** are as follows. First, we present a simple block-based replacement technique to maximize "urgency-worthiness" of the bytes occupying the caches of different devices. Second, our experimental results show this block-based technique provides the best average startup latency when deployed in a cooperative mode. With multiple devices where each device acts greedy, the clip-based alternative that maximizes byte-hit ratio of each device is superior. The cooperative block-based technique is surprisingly simple. We present alternatives that favor caching prefetch (also named prefix) blocks and show they impact the average startup latency adversely. The explanation for why these intuitive alternatives are not effective is a third contribution of this study.

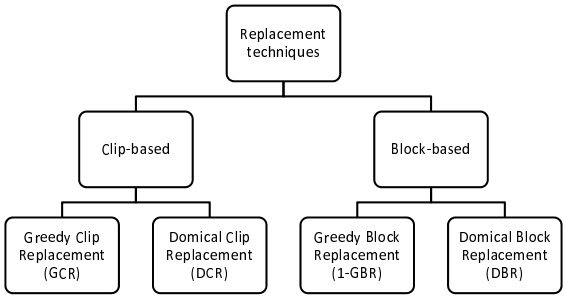


Figure 1. A taxonomy of the caching techniques.

In order to focus on alternative replacement techniques of Figure 1, we make several simplifying assumptions[1] about both the wireless environment and their application usage. We describe these in turn. First, we assume the wireless home consists of a handful of devices and each device is configured with a mass storage device. Second, we assume the cache space of a device is larger than a clip. Third, a device caches an entire clip when a user references it in order to support temporal references for the same clip by the user(s) of that device. Fourth, the bandwidth between nodes is asymmetric [29]. Hence, the bandwidth from Node i to Node j might be different than the bandwidth from Node j to Node i. Fifth, devices may stream a clip amongst one another using a middleware such as [21, 2]. This middleware requires each device to monitor its network bandwidth and delay characteristics. Our cooperative caching techniques employ this information to construct dependency groups, see the description of the Domical and Cont-Coop techniques of Section 3. We assume the coordinator of these middlewares [2, 21] admits requests by reserving link bandwidths between devices, minimizing the likelihood of an active display suffering from jitter when a new display is initiated and the bandwidth of a connection between two or more devices is exhausted. Sixth, an overlay network such as CAN [31] or Chord [34] facilitates discovery of data cached by devices cooperating in a home network [25, 20, 3]. And finally, a home gateway [3], node G in Figure 2, serves as the gateway between the wireless home network and the infrastructure outside the home.

The applications used by members of a household define the working set of the household, denoted WS. (While we focus on streaming media, our techniques apply to other large data items such as images.) 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. The size of this working set is denoted as S_{WS}. We consider different access patterns to WS such as 80% of requests referencing 20% of the clips that constitute WS, and other more skewed and uniform patterns of access. They are implemented using a Zipfian distribution [39].

We study the average startup latency under the worst case scenario when all devices in a household are activated to display clips simultaneously.

The rest of this paper is organized as follows. Section 2 surveys research related to our replacement techniques. In Section 3, we present a greedy and a cooperative clip replacement technique, GCR and DCR. Section 4 does the same focusing

[1] Some are trivial to remove.

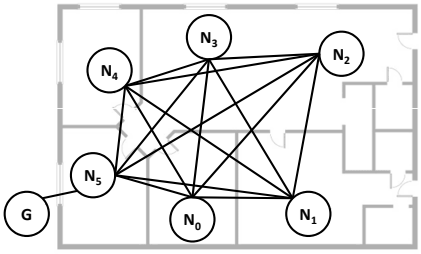


Figure 2. Real Home Network Topology

on block replacement techniques introducing 1-GBR and DBR techniques. We evaluate these alternatives in Section 5 and present four key lessons learnt. Brief conclusions and future research directions are presented in Section 6.

2 Related Works

Our proposed replacement techniques complement the vast body of work on proxy web servers for streaming media [35, 23, 33, 2, 32, 37, 7, 22, 6, 14], see [24] for a survey. Below, we survey replacement techniques found in the prior literature as they constitute the focus of our study.

The greedy replacement techniques include variants of LFU [28, 26, 14], those based on a utility function [37, 7], HistLRUpick [2], Rainbow [6], P2P proportional replacement technique of [32], and PB, IB and IF techniques of [22]. These can be categorized into those that strive to enhance startup latency [22] or some other metric such as byte and cache hit ratios [2, 32, 37, 7, 22, 6, 14]. Enhancing startup latency does not necessarily enhance other metrics [22, 8, 32, 15]. Due to lack of space, we focus on those techniques to enhance average startup latency as it constitutes our focus.

Partial bandwidth based caching (PB) technique [22] victimizes blocks of clips based on their display bandwidth requirement ($B_{Display_i}$) and the available network bandwidth (B_{Net}) in order to enhance average startup latency. Assuming T_i is the display time of a clip i, it caches ($B_{Display_i}$-B_{Net})T_i of those clips with highest $\frac{f_i}{B_{Display_i}}$ value, where f_i is the frequency of access[2] to clip i. This technique is designed for proxy caches deployed on the internet and does not cache those clips whose $B_{Display_i}$ is less than the network bandwidth, B_{Net}. Our proposed block replacement technique employs the concept of urgency worthiness metric to choose victims. This metric is defined as $\frac{f_i}{B_{Display_i} \times d_{ij}}$ where d_{ij} denotes the display time of a block relative to its start. Moreover, since a proxy cache and a client are the same device in our wireless home network, we cache blocks independent of the network bandwidth.

A group of proxies may cooperate to increase the aggregate cache space, balance load, and improve system scalability [2, 6, 15, 16]. MiddleMan [2] segments clips into equi-sized segments and employs a centralized coordinator to balance load evenly across proxies. To minimize the overhead of switching among proxies to construct a media object, Silo data layout [6] partitions a clip into segments of increasing size, storing more copies of popular clips while guaranteeing at least one copy per segment. Its cooperative global replacement technique employs either a Lazy or a Token mechanism to redistribute

[2] Arrival rate of requests for clip i, λ_i, estimates f_i.

data. Cont-Coop [15] and Domical [16] 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 and Silo strive 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 [16]. 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 [2] and Middleware of [21], and proxy cluster of Silo [6].

Cooperative proxy caching for streaming media has been explored in overlay networks [20, 25]. COPACC [20] 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 [25] 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. Moreover, our insights into the tradeoffs between greedy block and clip replacement techniques (GBR and GCR) may shed some light on the discrepancy between the granularity of caching employed by OCS, see Section 6.

Finally, a cooperative replacement technique for home gateways in a neighborhood is detailed in [3]. Its replacement policy, named D-COORD, is at the granularity of a clip and considers both local and remote hits when choosing victims. Our study is different because we compare clip and block replacement techniques. While D-COORD is not designed for a wireless home network, we intend to study its extension to this environment as a future research direction, see Section 6.

3 Clip-based Replacement

With this technique, the granularity of caching at a device is the entire clip. Such a technique must consider the variable size of clips occupying the storage of a device. Candidates include GreedyDual [5] and its variants [10, 14], and dynamic simple (DYNSimple) [14]. With DYNSimple, a device victimizes those clips with the lowest byte hit value, $\frac{f_i}{S_i}$ where f_i is the frequency of access to clip i and S_i is the size of clip i. In [14], we explore these techniques and other alternatives such as LRU-K [27] extended to consider clip size. We showed DYNSimple provides a higher cache hit ratio than the other alternatives and is able to adjust to the changing pattern of access to the data.

With a handful of wireless devices in a home network, different devices may collaborate by making the state of their caches dependent on one another without sacrificing the autonomy of each device. When device i makes its cache state dependent on devices $\{0, ..., k\}$, device i victimizes clips that are common with those stored on devices $\{0, ..., k\}$ prior to victimizing its other clips. A detailed protocol to realize this is detailed in [15, 16]. Cooperative caching algorithms include Cont-Coop [15] and Domical [16]. Both require each device to compute a dependency group that consists of itself and zero or more other devices. Devices in group i make the state of their caches dependent on those devices in groups $\{0, ..., i\}$. If group i consists of only one device then the state of that device is dependent on those devices that appear in groups $\{0, ...,$

$i-1\}$. If group 0 consists of only one device then it will act in a greedy manner because the state of its cache depends on no other device. This flexible paradigm preserves autonomy of a device by allowing to choose to not participate in a group and manage its cache in a greedy manner.

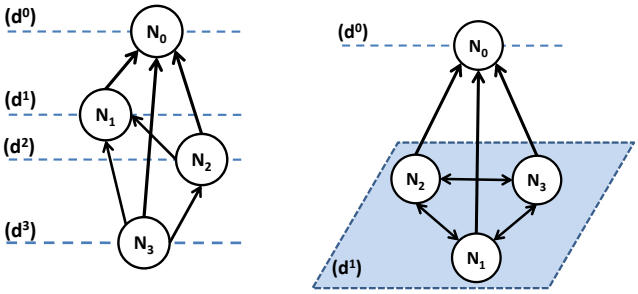
Cont-Coop and Domical differ in their number of dependency groups and assignment of nodes across these groups. Cont-Coop constructs only two groups numbered 0 and 1. Group 0 consists of only one device, named the core node. All other devices are assigned to group 1. This means the core node employs a greedy caching technique while all other devices make the state of their caches dependent on one another and the core node. Domical constructs $\mathcal{N}$ groups where $\mathcal{N}$ is the number of devices in the system. Each group consists of only one device and is numbered from 0 to $\mathcal{N}-1$. The device in group 0 employs a greedy caching technique. The device in group i renders the state of its cache dependent on those devices in groups $\{0, ..., i-1\}$.

As an example, Figure 3 shows the assignment of four nodes, numbered N_0 to N_3, with each technique. Domical constructs four groups numbered $\{d^0, d^1, d^2, d^3\}$ and assigns one node to each group. Cont-Coop constructs two groups with one node in d^0 and all other nodes in d^1. Nodes N_0 and N_3 behave the same way with both techniques: N_0 employs a greedy caching technique because it is assigned to d^0. Similarly, N_3 makes the state of its cache dependent on every other node. With Cont-Coop, nodes N_1 and N_2 act the same as N_3. With Domical, while the state of N_1's cache depends only on N_0, N_2 depends on both N_0 and N_1.

Both Cont-Coop and Domical construct the dependency groups with the objective to minimize the likelihood of a wireless network connection between two or more devices from becoming a bottleneck. This is important because streaming a clip requires reservation of wireless link bandwidths to ensure a display free from disruptions and delays. When the bandwidth of a link is exhausted, a new request that requires the use of that link must wait for an in progress display to complete, resulting in a high startup latency. To minimize the likelihood of this event, in [16], we develop the bandwidth contention metric to estimate the amount of imbalance across the network links when a node N_i streams a clip to every other node in the network. Intuitively, the node with the smallest contention value results in the lowest imbalance, avoiding the formation of bottleneck links. Cont-Coop computes this metric for every node and assigns the node with the smallest value to group 0 and all the other nodes to group 1. Domical computes the same metric for every node, sorts them based on this metric, and assigns them to the groups in ascending order: node with the smallest value is assigned to group 0 while the node with the highest value is assigned to group $\mathcal{N}-1$.

With a high degree of network connection between wireless devices (such as those reported in [29]), Domical enhances startup latency when compared with Cont-Coop [15]. When the network bandwidth exceeds the bandwidth required to display a clip, both techniques outperform a deployment where each device employs a greedy replacement (GCR and 1-GBR).

We use Domical to develop our cooperative block-based technique, see Section 4. Section 5 compares Domical variations of clip and block replacement techniques with one another.



3.a Domical 3.b Cont-Coop

Figure 3. Different cache dependency groups.

4 Block-based Replacement

This section describes a greedy and a cooperative replacement technique for a device that manages its cache at the granularity of a block. They are named 1 Node Greedy Block Replacement (1-GBR) and Domical Block Replacement (DBR), respectively. As implied by its name, 1-GBR is appropriate for an environment consisting of a device with connectivity to a gateway. Both 1-GBR and DBR strive to enhance the incurred average startup latency. We conceptualize a clip as a sequence of blocks and introduce the concept of *urgency-worthiness* of a block defined as how urgently the bytes of that block are needed once the clip's display has been initiated. While DBR applies this concept across all blocks uniformly, 1-GBR applies the concept to non-prefetch blocks only and favors caching of prefetch blocks. Section 4.2 explains this design difference. DBR employs the Domical dependency groups of Section 3 and is surprisingly simple. Section 4.3 describes intuitive extensions of the DBR that favor caching prefetch blocks of a clip always. Lesson 2 of Section 5 shows these alternatives are inferior to the simple technique.

4.1 1-GBR: Greedy Block Replacement

One Node Greedy Block Replacement, 1-GBR, is appropriate for an environment consisting of a single device. Figure 4 shows the pseudo-code of 1-GBR as invoked by a node, N_k. To explain its details, assume N_k is in the range of a wireless home gateway and references clip X with some of its blocks missing from N_k's local cache. Set M contains these missing blocks. If N_k has sufficient free cache space to store members of set M then N_k streams and stores them in its cache. If N_k has the first few blocks of X such that their display masks the time required to download the blocks in set M then N_k initiates the display of X. This form of overlapping the display of a clip with downloading its missing blocks is named *progressive* [26] display. The number of required blocks is a function of the bandwidth of the wireless connection from the home gateway to the device ($B_{Network}$) and the bandwidth required to display the clip X ($B_{Display_x}$). It is defined as $p_X = Max(0, \lceil (1 - \frac{B_{Display_x}}{B_{Network}}) \times |X| \rceil)$ where $|X|$ denotes the number of blocks of clip X. Assuming blocks of X are numbered from 0 to $|X| - 1$, those blocks with ids smaller than p_X constitute the prefetch portion of X. They are named the prefetch blocks, P-blocks of clip X. Other blocks of X are named non-prefetch, NP-blocks of X.

Let F denote the free cache space of N_k;
Let M denote those blocks of X missing from N_k;
Let P denote prefetch blocks of different clips
    (excluding those of X) in N_k's cache;
Let NP denote the non$-$prefetch blocks of different clips
    (excluding those of X) in N_k's cache;
While ($F < Sizeof(M)$) {
  if (NP is nonempty)
        Victimize those NP blocks with minimum $\frac{f_i}{d_{ij} \times B_{Display_i}}$;
  else
        Victimize those P blocks with minimum $\frac{f_i}{S_{P_i}}$;
}
Stream and cache those blocks in set M;

Figure 4. 1-GBR: Block-based greedy caching algorithm

When N_k's cache is full such that it cannot store blocks in set M, then N_k frees space by selecting victim blocks and swapping them out. To select victims, 1-GBR maintain 2 disjoint sets of blocks residing in its local cache: $\{$P-blocks$\}$, and $\{$NP-blocks$\}$. It victimizes NP-blocks first using their urgency-worthiness, defined as $\frac{f_i}{d_{ij} \times B_{Display_i}}$. The variable d_{ij} defines how urgently block j of clip i is needed for display. Assuming the display of clip i may tolerate δ startup latency and the first byte of block j is displayed at time t relative to the start of the clip display time, d_{ij}=t+δ.

Once NP-blocks are exhausted, 1-GBR victimizes P-blocks using their byte-hit ratio, $\frac{f_i}{S_{P_i}}$ where S_{P_i} denotes the size of P-blocks. This is because the urgency of all P-blocks in node N_k blocks are the same. The byte-hit metric enhances the likelihood of a future request finding the prefetch portion of its referenced clip in the cache.

We ran experiments to compare 1-GBR with DYNSimple [14] of Section 3. Our target environment consists of one device with S_T as its cache space. The WS consists of 576 clips with a bandwidth requirement of 4 Mbps. All clips have a display time of 30 minutes. This last assumption renders DYNSimple to be LFU. 1-GBR outperforms DYNSimple in all our experiments including those with heterogenous mix of media types. Figure 5 shows the results for different $\frac{S_T}{S_{WS}}$ ratios with a skewed distribution of access. This figure shows the percentage improvement obtained by the block-based when compared with clip-based caching for wireless connections with bandwidth of 1, 2, and 3 Mbps. The block-based greedy caching technique outperforms DYNSimple because it maintains the prefetch portion of different clips cache resident. DYNSimple maximizes the byte-hit ratio and does not differentiate between the different blocks of a clip and how urgently they are needed.

4.2 DBR: Domical Block Replacement

The cooperative version of block replacement technique constructs the dependency groups between different devices using the Domical technique, see Section 3. A device N_i maintains two lists of block occupying its cache: 1) those that occupy the cache of devices that N_i depends on, overlapping blocks, labeled O-blocks, 2) all other blocks, non-overlapping blocks, labeled NO-blocks. DBR victimizes the O-blocks prior to swapping out NO-blocks. It chooses victims with the objective to

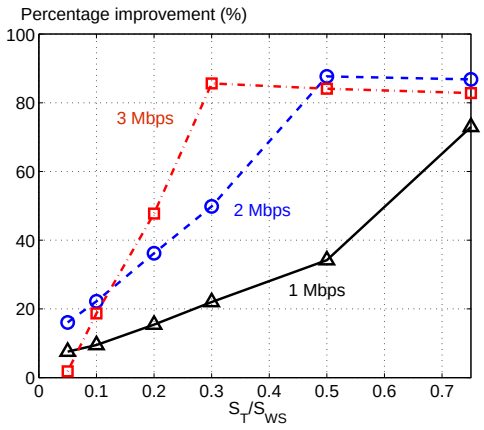


Figure 5. 1-GBR Vs. DYNSimple.

Let F denote the free cache space of N_k;
Let M denote those blocks of X missing from N_k;
Let $O-$Blocks denote those blocks (excluding those of X)
    that are in common with the dependency groups that N_k's
    cache depends on;
Let $NO-$Blocks denote those blocks (excluding those of X)
    that **do** NOT overlap with the dependency groups that N_k's
    cache depends on;
While ($F < Sizeof(M)$) {
  if ($O-$Blocks is nonempty)
        Victimize those $O-$Blocks with minimum $\frac{f_i}{d_{ij} \times B_{Display_i}}$;
  else
        Victimize those $NO-$Blocks with minimum $\frac{f_i}{d_{ij} \times B_{Display_i}}$;
}
Stream and cache those blocks in set M;

Figure 6. DBR: Domical Block Replacement

maximize the urgency-worthiness of the bytes occupying the different caches in the cooperative group. This simple technique is extremely effective, see Lessons 1 and 2 of Section 5.

DBR does not differentiate between P-blocks and NP-blocks of different clips occupying its cache. While appropriate with an environment consisting of multiple devices, DBR is not a substitute for 1-GBR in an environment consisting of a single device. For example, with a 3 Mbps network connection from the home gateway to a device and $\frac{S_T}{S_{WS}}$=0.3, startup latency with DBR is three times worse than 1-GBR. This is because DBR does not cache the prefetch portion of as many clips as possible. As detailed in Lesson 1 of Section 5, the urgency-worthiness metric causes a device to cache all blocks of a few popular clips by swapping out the prefetch portion of the least popular clip.

4.3 IntPrefix and FragPrefix: Prefix variations of DBR

This section presents two intuitive extensions to the simple DBR technique of Section 4.2. The first alternative, Integral Prefix (IntPrefix), strives to maintain the prefetch portion of clips cache resident. Each device measures its available downlink bandwidth from its one hop neighbors in order to compute the prefetch portion of each clip, see Section 4.1. It favors P-blocks of different clips by victimizing blocks[3] as follows: First, it identifies the O-blocks and the NO blocks of different clips. Next, it separates these into P-blocks and NP-blocks. This results in four sets of blocks: 1) O-blocks:NP-blocks, 2) NO-blocks:NP-blocks, 3) O-blocks:P-blocks, and 4) NO-blocks:P-blocks. It swaps out blocks starting with those in set 1. It does not swap out blocks of the next set unless the current set becomes empty. Moreover, once it starts to delete the blocks of a prefetch portion of a clip X, it does not victimize prefetch portion of another clip until all blocks of X's prefetch portion are swapped out. In essence, it allows at most one partial prefetch portion of a clip to occupy the cache of a device.

FragPrefix modifies IntPrefix in two ways. First, it switches the order of sets 2 and 3 in order to victimize the overlap blocks first always. Second, blocks that constitute the prefetch portion of different clips compete based on their urgency-worthiness metrics. Hence, partial prefetch portions of different clips may compete for the cache of a device.

Lesson 2 of Section 5 shows FragPrefix is superior to IntPrefix. DBR outperforms both by a wide margin.

5 Comparison

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

5.1 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. S_T denotes the sum of the size of caches contributed by $\mathcal{N}$ nodes in a cooperative group, S_{WS} is the size of the working set. A node may employ one of the presented techniques to control the content of its cache.

An edge from node N_i to node N_j means N_j is in the radio range of N_i to receive data from it. There must exist an edge from N_j to N_i in order for N_i to receive data from N_j. Each edge is assigned a pre-specified bandwidth, providing the asymmetric transmission rates between nodes as described in [29].

Node N_i may have e edges with different bandwidths to e different nodes in the system. We consider two different transmission models. The first, termed kp-card, assumes the bandwidths correspond to the transmission rates observed by N_i's k cards using their p channels to communicate with the e nodes. In essence, N_i's total out-going bandwidth is the sum of the bandwidths specified on its out-going edges. The second, termed Shared-BW, assumes the bandwidths are shared and N_i may not transmit at the full rates specified on each of its out-going edges simultaneously. As an example, assume N_i has two out-going edges to nodes N_j and N_k with bandwidths of 10 and 12 Mbps, respectively. With the kp-card model, N_i

[3]This conceptual description might be implemented efficiently in several ways.

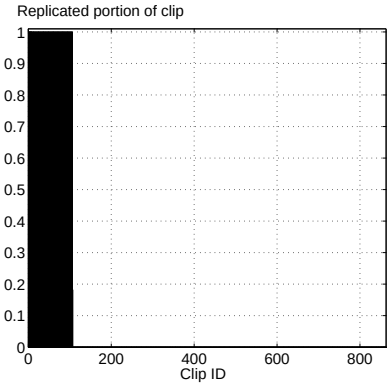


Figure 7. Starting Clip distribution $\frac{S_T}{S_{DB}} = 0.75$

may transmit to N_j at a rate of 10 Mbps while transmitting to N_k at a rate of 12 Mbps simultaneously. With the Shared-BW model, if N_i transmits data to N_j at a rate of 10 Mbps then it may transmit data to N_k at a rate of 2 Mbps only. With both models, N_i's minimum data rate is 10 Mbps. However, N_i's maximum data rate is 22 Mbps and 12 Mbps with kp-card and Shared-BW, respectively. For the rest of this paper, we assume Shared-BW transmission model.

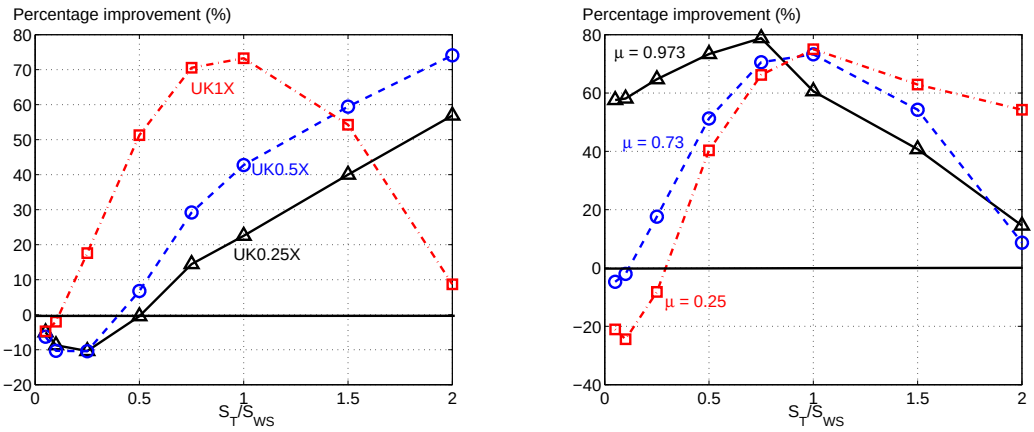
We assume a repository of constant bit rate (CBR) video clips. The display bandwidth requirement of each clip is 4 Mbps. We assume the working set (WS) consists of 864 clips. We examined a variety of media and clip mixes that result in different bandwidth requirements. In all cases, We observed the same lessons. To simplify discussion and without loss of generality, for the rest of this paper, 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 [4] to generate requests for different clips. To elaborate, Zipf's law [39] defines the relative popularity of a request for the i'th most popular clip is proportional to $\frac{\chi}{i}$ where χ is a normalizing constant. Different studies provide different definitions for this law. Assuming C clips are rank ordered based on their popularity (1, 2, ..., C), a general definition named Zipf-like distribution is as follows. The probability of access for clip i is: $\frac{\chi}{i^\mu}$. The exponent μ ($0 \leq \mu \leq 1$) controls the mean of the distribution and $\chi = \frac{1}{\sum_{i=1}^{C} \frac{1}{i^\mu}}$. A larger value for exponent μ makes the distribution more skewed by increasing the frequency of access to a few popular clips. On the other hand, a smaller value of μ makes the distribution more uniform. Zipf has been used to model the distribution of web page requests [18, 4, 36], and sale of movie tickets[4] in the United States [12].

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 [11] 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. 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

[4] In [12], a Zipf-like distribution is defined as $\frac{\chi}{i^{(1-\omega)}}$ where ω is 0.27. In this paper, μ equals $1 - \omega$. To be consistent with [12], we analyze 0.73 as a possible value for μ in Section 5.



8.a $\mu = 0.73$ 8.b $UK1X$

Figure 8. Percentage improvement with DBR caching when compared with cooperative clip-based caching.

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. One may introduce an arbitrary delay (termed a think time) between different iterations. However, the observed values will not change because: 1) all devices are activated in each round, and 2) we measure the average startup latency incurred in each round.

In each iteration, we measure the following parameters local to a node: startup latency, byte-hit and cache hit ratios, 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 first 108 popular clips are stored in the cache of device, see Figure 7. The number of cached clips changes for different $\frac{S_T}{S_{WS}}$ values. Lesson 1 shows how this starting state evolves with the alternative caching techniques.

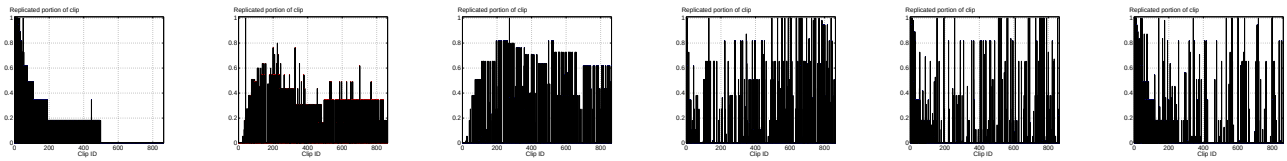
We focus on a realistic wireless home network corresponding to a deployment of six nodes employing 802.11a networking cards in a British household [29] with asymmetric link bandwidths. This environment is labeled $UK1X$. We analyze scaling of the link bandwidth of this environment by 0.5 and 0.25. These environments are labeled $UK0.5X$ and $UK0.25X$, respectively. We have studied alternative synthetic home network topologies with link bandwidths ranging from 2 to 16 Mbps. Our key observations hold true for these topologies as well. Due to lack of space, we do not present these results.

5.2 Performance Results

This section presents four key lessons observed from our experiments.

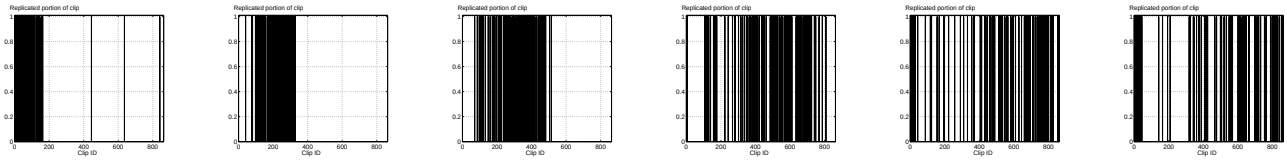
Lesson 1: Block replacement technique (DBR) outperforms clip replacement technique (DCR) when nodes cooperate.

Figure 8 shows the percentage improvement observed with Domical Block Replacement (DBR) when compared with



9.a Node 4 9.b Node 0 9.c Node 2 9.d Node 1 9.e Node 3 9.f Node 5

Figure 9. Distribution of blocks across devices in the network $\mu = 0.73$, $\frac{S_T}{S_{WS}} = 0.75$



10.a Node 4 10.b Node 0 10.c Node 2 10.d Node 1 10.e Node 3 10.f Node 5

Figure 10. Distribution of clips across devices in the network $\mu = 0.73$, $\frac{S_T}{S_{WS}} = 0.75$

Domical Clip Replacement (DCR). 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 8 as a function of different $\frac{S_T}{S_{WS}}$ values. Figure 8.a shows this improvement for a skewed distribution of access to clips ($\mu = 0.73$) and different scaling of the bandwidths reported for the British household of [29]. Figure 8.b focuses on $UK1X$ and analyzes alternative distributions of access to clips: very skewed with μ=0.973 to a more uniform distribution with μ=0.25. These results show DBR outperforms DCR by a significant margin, enhancing average startup latency.

The bandwidth contention metric of the Domical cooperative technique constructs the same Domical dependencies between the nodes with both clip and block replacement techniques. Both assign Node 4 to d^0, and assign 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. We describe this starting with the block-based technique.

Figure 9 shows placement of data across different nodes at the end of the simulation when μ=0.73 and $\frac{S_T}{S_{WS}}$=0.75. The x-axis of this figure is the id of clips in descending order of popularity. The y-axis shows the fraction of each cached clip. Figure 9 is evolution of the original assignment shown in Figure 7. It shows the following three observations. First, DBR enables Node 4 to cache the (first 23) popular clips of the WS in their entirety. It caches a decreasing fraction of the remaining clips based on their popularity: 50-90% of clips 24 to 44, 20-50% of clips 45 to 110, and 10% of clips 111 to 495. The last category of 384 clips consists of one block per clip because each clip consists of 10 blocks. Node 4 exhibits this behavior because (a) the state of its cache is **in**dependent of other devices, and (b) blocks compete for the available cache space using their urgency worthiness metric.

Second, Node N_5 with highest dependency on other nodes contains more popular blocks than other nodes with fewer dependencies on other devices such as N_0. Figure 9 shows this as follows. While Nodes 0 and 2 have no blocks of the (first 23) popular clips, an increasing number of such blocks occupies the caches of Nodes 1, 3, and 5 with Node 5 containing the highest number. This is counter intuitive because Node 0's cache depends only on Node 4 while Node 5's cache depends on

all nodes. To explain this, one must separate the logical placement suggested by Domical from its realized placement[5]. Its logical placement suggests five groups of blocks sorted in descending order based on their urgency-worthiness. Logically, these should be assigned to Nodes 4, 0, 2, 1, 3, and 5, respectively, because of their dependency order. However, in practice, a block occupies the cache of a device only when it is referenced. If an unpopular block is not referenced then the popular blocks compete for its logical cache space. This is the case with Node 5: Its logically assigned blocks are so unpopular that Node 5 does not reference them. This enables the blocks of a popular clip referenced by Node 5 to occupy its cache permanently. Moreover, when a node in the dependency group of Node 5 (say Node 0) references an unpopular clip (that should belong to Node 5 logically) coincidentally, Node 0 caches blocks of this clip. This prevents Node 5 from caching the blocks of this clip because the state of its cache depends on Node 0.

Nodes 0 and 2 service their frequent references for the popular clips by streaming them from Node 4. This is desirable because Node 4 provides the highest network bandwidth and balances the load across the wireless network connection evenly. The bandwidth contention ratio metric of Domical [16] is responsible for computing this node and assigning it to d_0.

Third, Nodes 0 and 2 contain mostly partial clips while other nodes have tens of clips in their entirety. In the experiments of Figure 9, Nodes 4, 0, 2, 1, 3, and 5 have 23, 1, 1, 24, 27, and 25 complete clips in their caches, respectively. The explanation for this is as follows. Nodes 0 and 2 are forced to contain the prefetch blocks of many different clips because: 1) they are dependent on Node 4, and 2) they reference the blocks of clips logically assigned them. Nodes 1, 3, and 5 do not reference all their logically assigned clips. Moreover, Nodes 4, 0, and 2 have a very small fraction of overlapping blocks ($< 0.1\%$) while the other nodes have significantly higher number of overlapping blocks.

These 3 observations imply the following. When Nodes 4, 1, 3 and 5 reference the most popular clip, they observe a cache hit for this clip and do not need network bandwidth for its transmission. When Nodes 0 and 2 reference this clip, they retrieve almost all blocks of this clip from their neighboring devices. They have multiple candidate server nodes to stream these clips from because these blocks have multiple (potentially four) replicas. Domical chooses Node 4 strategically because it has the highest amount of network bandwidth and streams blocks to Nodes 0 and 2 without causing bottleneck links.

Figure 10 shows the distribution of clips across the nodes with Domical Clip Replacement, DCR. The greatest concentration of popular clips is on Node 4 because it employs DYNSimple in a greedy manner. Node 0 contains the 2nd most popular group of clips because its state depends on Node 4 and it victimizes those clips that are cached by Node 4 first. While Nodes 2, 1, 3, and 0 contain most of the unpopular clips, it is interesting to note that Nodes 3 and 5 contain the most popular clips. The explanation for this is the same as DBR where a node, say Node 5, must reference an unpopular clip in order to cache it. If Node 5 does not reference its logically assigned unpopular clips then popular clips start to occupy its cache space.

While the impact of Domical on both clip and block replacement techniques is identical, the block-based caching has three advantages. First, some devices contains 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 cached across 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

[5]The realized placement is intentional while the suggested logical placement is undesirable.

$\frac{S_T}{S_{WS}}$	DBR	IntPrefix	FragPrefix
0.05	3165	4043 (-21.71%)	3701 (-14.47%)
0.1	2676	3734 (-28.32%)	3301 (-18.93%)
0.5	1334	2753 (-51.53%)	1675 (-20.34%)
0.75	870	2419 (-64.05%)	1120 (-22.35%)
1	546	2155 (-74.69%)	725 (-24.77%)
2	137	1290 (-89.37%)	241 (-43.08%)

Table 1. Average startup latency with DBR, IntPrefix and FragPrefix.

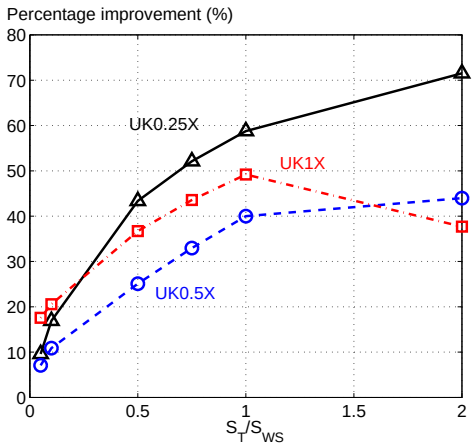


Figure 11. Percentage improvement in startup latency by GCR when compared with GBR. $\mu = 0.73$

intermediatory to the home gateway and the infrastructure outside of the home.

Lesson 2: IntPrefix and FragPrefix are inferior to DBR.

Table 1 shows DBR enhances average startup latency when compared with its two variants IntPrefix and FragPrefix described in Section 4.3. The numbers in parentheses show the percentage degradation in startup latency observed with IntPrefix and FragPrefix when compared with DBR. IntPrefix is inferior to both DBR and FragPrefix because it materializes the prefetch portion of as many clips as possible in the cache of each device. When a device references a clip, its NP-blocks must be streamed using the home gateway. This exhausts N_5's wireless bandwidth connection, see Figure 2, causing devices to wait for one another. FragPrefix victimizes overlapping P-blocks first (prior to non-overlapping NP-blocks), reducing the degree of replication for the P-blocks. However, it continues to favor replication of P-blocks because it victimizes NP-blocks that are overlapping first (independent of their urgency-worthiness). In contrast, DBR forces overlapping P and NP blocks to compete for the available cache space using their urgency-worthiness metric, outperforming FragPrefix.

Lesson 3: Greedy Clip Replacement (GCR) outperforms Greedy Block Replacement.

One may deploy the block replacement technique of Figure 6 in a greedy manner by making the state of each node independent of the others. This causes the state of each device to evolve to become identical to the one shown in Figure 9.a. This is better than 1-GBR because 1-GBR materializes the prefetch portion of each clip onto each device. When different devices reference different clips, they must stream their remaining blocks via the home gateway and N_5, exhausting N_5's

bandwidth and resulting in a high startup latency.

Greedy Clip Replacement (GCR) is superior to both block replacement techniques because it caches the popular clips in their entirety on each device. When a node references a popular clip that resides in its cache, it no longer requires the network bandwidth. This frees the network bandwidth to service those nodes that observe a cache miss for a clip.

Figure 11 shows the percentage improvement observed with GCR when compared with the best of greedy block replacement techniques for a variety of $\frac{S_T}{S_{WS}}$ thresholds and scaling of the network bandwidths observed with the British household.

Focusing on $UK1X$, the percentage improvement observed with clip-based caching increases as a function of $\frac{S_T}{S_{WS}}$, reaches its maximum when $\frac{S_T}{S_{WS}}$ equals one, and then starts to decrease. The explanation for this is as follows. With small values of $\frac{S_T}{S_{WS}}$, clip-based caching also observes many misses that result in bandwidth contention and delays (similar to block-based). As we increase $\frac{S_T}{S_{WS}}$, each node observes a higher cache hit with the clip-based caching, reducing the bandwidth contention. DBR also observes a higher hit ratio for the prefetch-portion of clips. However, it must retrieve the non-prefetch blocks using N_5 and continues to observe delays because N_5's bandwidth is exhausted. Beyond $\frac{S_T}{S_{WS}}$ of one, the percentage improvement drops because block-based caches a larger number of clips in their entirety and starts to exhibit a behavior similar to GCR.

With $UK0.5X$ and $UK0.25X$, the same general curve repeats itself with one difference: Their peaks are higher than $UK1X$. This means GCR will outperform block replacement by a wider margin for those wireless homes with lower network bandwidths.

Lesson 4: DBR provides a lower availability of clips than DCR.

When the home network is disconnected from the infrastructure outside of the home, a device may display a clip that is 100% resident across the caches in the home network. DCR guarantees this because a clip is either cached in its entirety or none at all. With DBR, those clips with missing blocks cannot be displayed. Figure 12 shows the percentage of clips available with DCR and DBR. In general, DCR provides 10 to 15% higher clip availability when compared with DBR. Note that DBR materializes a higher ($\simeq 10\%$) percentage of WS across the caches when compared DCR. However, these blocks correspond to clips with blocks missing from the home network.

6 Conclusion and future research directions

In this study, we investigate a wireless home network and show a cooperative block replacement technique (DBR) enhances startup latency when compared with a cooperative clip replacement technique (CBR). If devices do not cooperate, the clip replacement technique is a better alternative. The concept of urgency-worthiness to choose victims is novel and central to the design of the block replacement technique.

We intend to extend this study in several ways. First, we intend to analyze our cooperative techniques with alternatives such as D-COORD [3] that require a device to consider remote references for a clip when choosing victims. Second, we plan to investigate impact of efficient data delivery algorithms such as patching [19, 9] and stream merging [13] with the clip and block replacement techniques. Third, we intend to explore the use of urgency-worthiness in support of proxy cache servers for both the Internet and overlay networks. This requires a comparison of our technique with alternatives such as IF, PB and

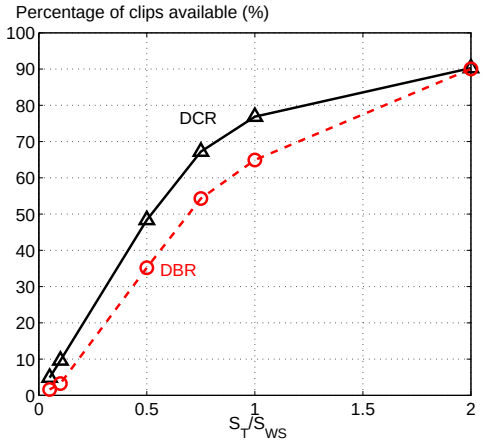


Figure 12. Data availability $\mu = 0.73$

IB of [22]. This will almost certainly expand the taxonomy of Figure 1, motivating the need for a more comprehensive survey of alternative replacement technique including the utility based techniques of [37, 7]. This survey might be similar to [30] and specific to streaming media [24].

References

[1] A. Aazami, S. Ghandeharizadeh, and T. Helmi. Near optimal number of replicas for continuous media in ad-hoc networks of wireless devices. In *MIS*, pages 40–49, 2004.

[2] S. Acharya and B. Smith. MiddleMan: A Video Caching Proxy Server. In *Proceedings of NOSSDAV*, June 2000.

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

[4] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web Caching and Zipf-like Distributions: Evidence and Implications. In *Proceedings of Infocom*, pages 126–134, 1999.

[5] P. Cao and S. Irani. Cost-Aware WWW Proxy Caching Algorithms. In *1997 Usenix Symposium on Internet Technologies and Systems*, 1997.

[6] 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.

[7] S. Chen, B. Shen, S. Wee, and X. Zhang. Adaptive and Lazy Segmentation Based Proxy Caching for Streaming Media Delivery. In *NOSSDAV*, 2003.

[8] S. Chen, B. Shen, S. Wee, and X. Zhang. Investigating Performance Insights of Segment-Based Proxy Caching of Streaming Media Strategies. In *MMCN*, 2004.

[9] S. Chen, B. Shen, Y. Yan, and X. Zhang. Buffer Sharing for Proxy Caching of Streaming Sessions. In *WWW*, 2003.

[10] L. Cherkasova and G. Ciardo. Role of Aging, Frequency, and Size in Web Cache Replacement Policies. In *Proceedings of the High-Performance Computing and Networking*, December 2001.

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

[12] A. Dan, D. Sitaram, and P. Shahabuddin. Scheduling Policies for an On-Demand Video Server with Batching. In *2nd ACM Multimedia Conference*, October 1994.

[13] D. Eager, M. Vernon, and J. Zahorjan. Optimal and Efficient Merging Schedules for Video-on-Demand Servers. In *ACM Multimedia*, 1999.

[14] S. Ghandeharizadeh and S. Shayandeh. Greedy Cache Management Techniques for Mobile Devices. In *AIMS*, 2007.

[15] S. Ghandeharizadeh and S. Shayandeh. Cooperative Caching Techniques for Continuous Media in Wireless Home Networks. In *Ambi-sys*, February 2008.

[16] S. Ghandeharizadeh and S. Shayandeh. Hierarchical cooperative caching: A novel caching technique for wireless home networks. In *Submitted for publication, working draft is available*, 2008.

[17] S. Ghandeharizadeh, S. Shayandeh, and T. Helmi. To share or not to share storage in mesh networks: A system throughput perspective. *AINAW 2008.*, pages 862–867, 25-28 March 2008.

[18] S. Glassman. A Caching Relay for the World Wide Web. In *WWW*, May 1994.

[19] K. Hua, Y. Cai, and S. Sheu. Patching: A Multicast Technique for True Video-on-Demand Services. In *ACM Multimedia*, September 1998.

[20] 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.

[21] 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.

[22] S. Jin, A. Bestavros, and A. Iyengar. Network-Aware Partial Caching for Internet Streaming Media. *Multimedia Systems*, 2003.

[23] J. Liu and J. Xu. Proxy Caching for Media Streaming over the Internet. *IEEE Communications Magazie*, 42(8):88–94, August 2004.

[24] J. Liu and J. Xu. Proxy Caching for Media Streaming over the Internet. *IEEE Communications*, August 2004.

[25] 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.

[26] W. Ma and D. H. C. Du. Design a Progressive Video Caching Policy for Video Proxy Servers. *IEEE Transactions on Multimedia*, 6(4):599–610, August 2004.

[27] 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.

[28] S. Paknikar, M. Kankanhalli, K. R. Ramakrishnan, S. H. Srinivasan, and L. H. Ngoh. A Caching and Streaming Framework for Mulitmedia. In *Proceedings of the eighth ACM international conference on Multimedia*, pages 13–20, New York, NY, USA, 2000. ACM.

[29] K. Papagiannaki, M. Yarvis, and W. S. Conner. Experimental Characterization of Home Wireless Networks and Design Implications. In *IEEE Infocom*, April 2006.

[30] S. Podlipnig and L. Boszormenyi. A Survey of Web Cache Replacement Strategies. *ACM Computing Surveys*, 35(4):374–398, December 2003.

[31] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 161–172, Aug. 2001.

[32] O. Saleh and M. Hefeeda. Modeling and Caching of Peer-to-Peer Traffic. In *ICNP*, pages 249–258, 2006.

[33] S. Sen, J. Rexford, and D.Towsley. Proxy Prefix Caching for Multimedia Streams. In *Proceedings of IEEE INFOCOM*, 1999.

[34] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, pages 149–160, San Diego, California, Aug. 2001.

[35] Y. Wang, Z. Zhang, D. Du, and D. Su. A Network Conscious Approach to End-to-End Video Delivery over Wide Area Networks Using Proxy Servers. In *Proceedings of IEEE INFOCOM*, 1998.

[36] 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 Oper. Syst. Rev.*, 33(5):16–31, April 1999.

[37] K. Wu, P. S. Yu, and J. L. Wolf. Segment-based Proxy Caching of Multimedia Streams. In *WWW*, 2001.

[38] M. Zink, K. Suh, Y. Gu, and J. Kurose. Watch global, cache local: Youtube network traffic at a campus network: measurements and implications. volume 6818, page 681805. SPIE, 2008.

[39] G. K. Zipf. Relative Frequency as a Determinant of Phonetic Change. Harvard Studies in Classified Philiology, Volume XL, 1929, 1929.