An Evaluation of Three Domical Block Replacement Techniques for Streaming Media in Wireless Home Networks*

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

Abstract

Wireless mesh home networks are deployed widely due to their ease of installation and economical prices. A typical network may consist of a handful of devices such as PCs, laptops, wireless consumer electronic devices, and game consoles. Devices may share data by making the state of their caches dependent on one another using a cooperative caching technique such as Domical. This sharing of data at the edges of the network reduces the load on the infrastructure outside of the household, freeing it to service other requests. In this paper, we analyze three local cache replacement techniques designed to enhance average startup latency of streaming media. All three pre-stage a fraction of a clip on a device in anticipation of its future reference in order to display the prefetch portion while streaming its remainder in the background. We use a simulation study of a realistic home network to compare these three techniques with one another when deployed with Domical. Obtained results show show one technique, named urgency-worthiness, is superior to the others when storage is abundant.

1. Introduction

Wireless home networks are deployed widely because they are inexpensive and simple to install. A typical wireless home network may consist of a handful of devices such as PCs, laptops, TV set-top boxes, and wireless interfaces for audio tuners and video games. The latter is an interesting trend with a larger number of consumer electronic devices becoming wireless. An example is Apple TV which can stream and display pay-per-view movies, video clips from popular web sites such as YouTube, and clips that reside on a PC or a laptop in a household. A device might be configured with a mass storage device due to its economical prices[1]. For example, at the time of this writing, one may purchase two different versions of Apple TV with either 40 or 160 Gigabytes of storage.

Devices in a home may share data by streaming clips amongst one another. This sharing of the data at the edges of the network enhances key quality of service metrics such as average startup latency, defined as the average delay incurred from when a device references a clip to the onset of its display. It also reduces the load on the infrastructure outside of the home including remote servers, freeing these resources to service other households and users.

*A shorter version of this paper appeared in *the Proceedings of the IEEE International Symposium on Multimedia (ISM2008)*, Berkeley, California, December 15-17, 2008.

[1]A Terabyte disk drive costs less than $250 at the time of this writing.

One may realize a wireless home network using either an access point or an ad-hoc mode of communication (a mesh network). A mesh deployment provides a higher aggregate network bandwidth because a device may circumvent low network bandwidth connections [24].

Recently, we introduced two cooperative caching techniques, named Cont-Coop [12] and Domical [13], and showed they outperform a deployment that requires devices to employ a greedy caching technique. In [13], we showed Domical is superior to Cont-Coop. In [11], we analyzed Domical with different granularity of data caching, clip-based and block-based, showing the average startup latency improves when clips are cached at the granularity of blocks. If devices do not cooperate, caching at the granularity of clips is superior.

In this study, we focus on caching at the granularity of blocks and analyze the interaction between local replacement techniques employed by a device and Domical cooperative technique. We show the local replacement technique has a significant impact on the average startup latency. We study three techniques targeted to enhance average startup latency: First, urgency-worthiness [11] (UW) replaces blocks based on how urgently their bytes are required relative to the start of a clip's display. Second, partial bandwidth-based [17] (PB) replaces blocks based on their download time from neighboring devices. Third, segment-based technique [30] replaces segments consisting of multiple blocks based on their segment id. While urgency-worthiness [11] was designed for wireless home networks, PB [17] and segment-based [30] were developed for the Internet. To the best of our knowledge, these three replacement techniques are the only ones designed to enhance average startup latency of streaming media. Other techniques enhance metrics such as cache or byte hit rate, distribution of workload across cache servers among others. These alternative metrics do not enhance average startup latency in a wireless home network because, with audio and video clips, one my pre-stage the first few blocks of a clip in order to initiate the display of a clip while retrieving its remainder in the background.

The contributions of this paper are two folds. First, it extends the design of PB to a wireless home network. Second, it compares performance of UW with PB and segment-based in a wireless home setup. Obtained results show that UW outperforms segment-based technique. When compared with PB, UW enhances average startup latency with abundant amount of storage. When storage is scarce, a variant of PB, named Conservative PB, provides a better startup latency.

The rest of this paper is organized as follows. Section 2 surveys research related to different cache replacement techniques. Section 3 provides an overview of Domical. UW, PB and segment-based are detailed in Sections 4.1, 4.3, and 4.2, respectively. The details of our simulation study are presented in Section 5. Section 6 presents our experimental results. Brief conclusions are presented in Section 7.

2. Related Work

This study builds on vast body of work on proxy web servers for streaming media [28, 18, 27, 1, 26, 30, 6, 17, 5, 10, 20, 15, 2, 30], see [19] for a survey. These techniques can be grouped into those that cache at the granularity of either a clip [20, 15, 2] or blocks [28, 18, 27, 1, 26, 30, 6, 17, 5, 10, 30]. Each group can be categorized into either greedy or cooperative techniques. One may taxonomize each sub-category based on the metric enhanced by a caching technique. Examples include a system metric such as cache hit rate or a quality of service metric such as average startup latency. We discard clip-based techniques and its sub-branches because our focus is on block-based techniques. Below, we describe this sub-tree of the taxonomy in detail.

Greedy block-based caching techniques include variants of LFU [23, 21, 10], those based on a utility function [30, 6], HistLRUpick [1], Rainbow [5], Segment-based technique [30], P2P proportional replacement technique of [26], PB, IB and IF techniques of [17], and UW [11]. PB [17], UW [11] and segment-based technique [30] strive to enhance startup latency while others enhance some other metric such as byte and cache hit rates [1, 26, 30, 6, 17, 5, 10]. Enhancing startup latency does not necessarily enhance other metrics [17, 7, 26, 12] and vice versa.

A group of proxies may cooperate to increase the aggregate cache space, balance load, and improve system scalability [1, 5, 12, 13]. MiddleMan [1] 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 [5] 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 data. Cont-Coop [12] and Domical [13] cache data with the objective to enhance average startup latency by balancing 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.

In this study, we focus on startup latency as a quality of service metric to compare the impact of the known greedy techniques (PB, UW, and segment-based) on the Domical cooperative caching technique. One may implement Domical, PB, UW, and segment-based using the centralized coordinators of MiddleMan [1] and Middleware of [16], and proxy cluster of Silo [5].

3. Overview of Domical

Domical cooperative caching [13] makes the state of caches of different devices dependent on one another. A device N_k may employ Domical as follows. When in radio range of other devices, N_k builds a profile of its network connectivity with its neighboring devices consisting of their identity, and both its observed upload and download bandwidths to them. Different devices may exchange this information with one another in order to construct a graph representing the connectivity between different devices. A node of the graph represents a device. Given a pair of neighboring devices, N_k and N_l, there are two edges representing N_k's download and upload links bandwidths to N_l.

Next, N_k invokes Domical with the connectivity graph to compute dependency groups between the different nodes. Given $\mathcal{N}$ devices, it computes $\mathcal{N}$ dependency groups with each group consisting of one device. These groups are labeled $\{d_0, d_1, ..., d_{\mathcal{N}-1}\}$. The dependency group specifies which device should make the state of its cache dependent on other devices. A device in dependency group d_i makes the state of its cache dependent on those devices that appear in groups d_0, d_1, ..., d_{i-1}. The device in dependency group d_0 employs a greedy caching technique such as LRU-K [22], UW [11], PB [17], DYNSimple [10], etc. Other devices also employ a greedy caching technique, however, they victimize data that resides in the caches of their dependency group first. In [12], we showed this maximizes the number of unique clips in the

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 $R-$Blocks denote those blocks (excluding those of X) that occupy N_k's cache and are unique to the nodes that N_k's cache depends on;
if F is insufficient to accommodate set M then victimize those $O-$Blocks with minimum $\frac{f_i}{d_{ij} \times B_{Display_i}}$ until sufficient free space
    is available to accommodate M;
if F is insufficient to accommodate set M then victimize those $R-$Blocks with minimum $\frac{f_i}{d_{ij} \times B_{Display_i}}$ until sufficient free space
    is available to accommodate M;
Stream and cache those blocks in set M;

Figure 1. Domical block replacement using urgency worthiness

mesh network, enabling a larger number of devices to stream data to one another. This enhances average startup latency when network bandwidth is abundant [12].

Domical constructs dependency groups with the objective to minimize formation of bottleneck links in the mesh network. It realizes this as follows. Each device N_k considers itself and every other device in the network as a candidate server. Considering each candidate server N_i in turn, N_k reserves potential paths that can be used for streaming from N_i to other nodes in the network. Next, N_k computes the standard deviation of bandwidth reservations across the different links. This metric is named bandwidth contention ratio metric [13, 12]. It sorts devices based on this metric and assigns them to the dependency groups starting with the device that has the lowest standard deviation. Different devices may exchange their dependency groups with one another, resolving conflicts by using a mechanism such as voting [12].

A key criterion for Domical is to be general enough to support devices participating in either dependency group d_0 or some other dependency group d_i. This is because when a new device participates in the mesh network, its dependency group is not known in advance. Moreover, it may modify existing assignment of devices to different dependency groups. A replacement policy should enable a device to participate as a member of any dependency group. Techniques described in this paper satisfy this criterion.

In [12], we compared Domical with a deployment where each device employs a greedy caching technique, showing it enhances startup latency by several factor. This is because Domical employs the bandwidth of wireless mesh network to facilitate sharing and exchange of data.

The following two sections present UW and PB local replacement techniques. The pseudo-code of each assumes the presence of dependency groups. For a device N_k, each strategy computes those blocks that are unique to its dependency group, termed *rare blocks*, and those with other replicas, termed *ordinary blocks*. These terms are used extensively in the remainder of the paper.

4. Three Replacement Techniques

4.1. Urgency Worthiness

In this section, we describe a cache replacement policy that swaps out the blocks from local cache of node N_k based on their *urgency worthiness* and assignment of N_k to the dependency group constructed by Domical.

Urgency worthiness of block b_{ij}, denoted by $\chi(b_{ij})$, is defined as how urgently bytes of block b_{ij} are needed once a device initiates the display of clip i. It is a function of the frequency of access to clip i and display time of b_{ij}. It is defined as: $\chi(b_{ij}) = \frac{f_i}{d_{ij} \times B_{Display_i}}$ where $B_{Display_i}$ is the bandwidth required to display clip i. The definition of d_{ij} is as follows. Considering a tolerable startup latency of δ for clip i, $d_{ij} = \delta + t$ where t is the display time of block b_{ij} relative to start of clip i.

Figure 1 shows Domical block replacement using UW. It requires each node, N_k, to maintain two disjoint set of blocks in its local cache: 1) Ordinary blocks (O-blocks) defined as those blocks in N_k's cache that also reside in the cache of those devices that N_k depends on, and 2) Rare blocks (R-blocks) defined as the set of blocks that occupy N_k's cache and are unique to the group of nodes that N_k depends on. When N_k caches clip X, it checks if it has enough free space to cache missing blocks of X. If this is the case, there is no need to invoke a replacement technique. Otherwise, N_k swaps out O-blocks and R-blocks using the *urgency worthiness* metric until it has enough free space to accommodate X. It should be noted that O-blocks are victimized first. Once the free space become available, N_k caches X.

4.2. Segment-based Replacement

A similar replacement strategy is segment-based technique of [30]. It is a greedy cache replacement technique designed for web proxies. Segment-based technique groups blocks of a clip into variable-sized, distance-sensitive segments. Within a clip the segment size increases exponentially. This implies that size of segment s_j is $2^j S_b$ where S_b is size of a block. This means that segment s_j consists of blocks $2^j - 1, \ldots, 2^{j+1} - 2$ for $j \geq 0$. The replacement algorithm uses $\frac{f_i}{j}$ to select victim segments. Pseudo-code of Figure 1 can be changed to implement the segment-based technique.

Usage of exponentially-sized segments helps segment-based technique to discard big chunks of a cached clip and quickly adapt to the changes in user access patterns. This reduces overhead of eviction process imposed to the system in comparison with a fixed size segmentation technique, i.e. UW. This overhead is an important factor that should be addressed for a shared environment, e.g. proxy server. But in a distributed environment like a wireless home network that each node acts independently, a fixed size segmentation technique, i.e. UW, can perform well.

4.3. Partial Bandwidth-based (PB) Replacement

Partial bandwidth-based replacement policy [17] was designed and developed for the Internet with the objective to enhance startup latency. It differentiates between prefetch and non-prefetch blocks of a clip. A block b_{ij} is a prefetch block iff:

$$j \leq Max(0, \lceil (1 - \frac{B_{Network_i}}{B_{Display_i}}) \times |i| \rceil) \tag{1}$$

where $|i|$ denotes the number of blocks of clip i. In order to enhance the startup latency, it pre-stages the prefetch portion of a clip onto a device in anticipation of a future reference for the clip. Once the clip is referenced, it displays the prefetch portion and down-loads remainder of the clip in the background.

In order to adapt PB for use with Domical in a home mesh network, this section answers two key design questions. First, how does a device victimize blocks given the dependency imposed by Domical? Second, what is the definition of $B_{Network_i}$

Let F denote the free cache space of N_k;
Let M denote those blocks of X missing from N_k;
Let $ONP-$Blocks denote those *non-prefetch* blocks (excluding those of X) of N_k with replicas in the cache of those nodes that N_k's cache depends on;
Let $OP-$Blocks denote those *prefetch* blocks (excluding those of X) of N_k with replicas in the cache of those nodes that N_k's cache depends on;
Let $RNP-$Blocks denote those *non-prefetch* blocks (excluding those of X) that **do** NOT overlap with the dependency groups that N_k's cache depends on;
Let $RP-$Blocks denote those *prefetch* blocks (excluding those of X) that **do** NOT overlap with the dependency groups that N_k's cache depends on;
if F is insufficient to accommodate set M then victimize those $ONP-$Blocks with minimum f_i until sufficient free space is available to accommodate M;
if F is insufficient to accommodate set M then victimize those $OP-$Blocks with minimum $\frac{f_i}{B_{Network_i}}$ until sufficient free space is available to accommodate M;
if F is insufficient to accommodate set M then victimize those $RNP-$Blocks with minimum f_i until sufficient free space is available to accommodate M;
if F is insufficient to accommodate set M then victimize the $RP-$Block with minimum $\frac{f_i}{B_{Network_i}}$ until sufficient free space is available to accommodate M;
Stream and cache those blocks in set M;

Figure 2. Domical block replacement using PB

in Equation 1? Below, we provide an answer to each question in turn.

Logically, PB maintains 4 disjoint set of blocks in N_k's cache: 1) Ordinary non-prefetch blocks, ONP-blocks, 2) Ordinary prefetch blocks, OP-blocks, 3) Rare non-prefetch blocks, RNP-blocks, and 4) Rare prefetch blocks, RP-blocks, unique to device N_k and its dependency group. PB victimizes blocks from these different disjoint sets based on their numbering order. Figure 2 shows the pseudo code of PB replacement as invoked by a node, N_k, when it has insufficient cache space. Note that the non-prefetch blocks are victimized using their frequency of access (f_i) and the prefetch blocks are swaped out using $\frac{f_i}{B_{Network_i}}$. In the following, we describe alternative ways to specify $B_{Network_i}$.

Given a device N_k in the radio range of several neighboring devices, one may define its download network bandwidth for clip i ($B_{Network_i}$) in several ways. Below, we provide two general classes of techniques and describe several instances of each:

- *Aggregate-PB*: We employ the shared download link bandwidth from neighbors of N_k as the definition of $B_{Network_i}$ using an aggregate such as minimum, average, median, mode, maximum, and others. We introduce Conservative PB by defining $B_{Network_i}$ as the *shared* minimum download link bandwidth from N_K's neighbors; $B_{Network_i} = \frac{\min_{\forall N_l \in neighbor(N_k)} B_{N_l \rightarrow N_k}}{|neighbor(N_k)|}$. To illustrate, if the minimum bandwidth of download link to node N_4 is 1 Mbps and N_4 has 5 neighbors, the minimum shared download link bandwidth is 0.25 Mbps. This is because when all neighbors of N_k transmit simultaneously, they reduce the minimum bandwidth proportionally [3].

  We introduce Optimistic PB by using the maximum shared bandwidth, $B_{Network_i} = \frac{\max_{\forall N_l \in neighbor(N_k)} B_{N_l \rightarrow N_k}}{|neighbor(N_k)|}$. By using the average bandwidth, we would introduce Moderate PB, $B_{Network_i} = \frac{\sum_{\forall N_l \in neighbor(N_k)} B_{N_l \rightarrow N_k}}{|neighbor(N_k)|^2}$. As an example of the latter, if the download link bandwidths from neighbors of N_4 are 11, 1, 11, 11, and 1 Mbps (from nodes N_0, N_1, N_2, N_3, and N_5, respectively). While the average is 7 Mbps, the *shared* average bandwidth is 1.4 Mbps. The rational for this is the same as Conservative PB.

Term	Definition
Startup latency (δ)	Delay from when a user references a clip to the onset of display.
$\mathcal{N}$	Number of nodes in the network
μ	Mean of the distribution of access
S_{WS}	Size of the working set
S_T	Total cache capacity of nodes in the network
f_i	Frequency of access to clip i
S_i	Size of clip i
$B_{Network_{ij}}$	Available bandwidth for downloading b_{ij}
$B_{Network_i}$	Available bandwidth for downloading clip i
$neighbor(N_k)$	Set of neighbors of N_k
$peer(N_k, i)$	Set of neighbors of N_k that have a copy of clip i in their caches
$B_{N_l \rightarrow N_k}$	Available bandwidth from node N_l to N_k

Table 1. Terms and their definitions

- *Peer-PB*: Since Domical assumes a device has knowledge of the blocks contained in its neighboring devices, given a block b_{ij}, one may employ the shared download link bandwidth from those neighbors to compute $B_{Network_{i,j}}$. The complexity of using such a metric impacts the definition of prefetch blocks. To illustrate, consider two consecutive blocks of a clip i, say $b_{i,5}$ and $b_{i,6}$, with copies on two different neighbors of N_k. The download link bandwidth of the neighbors might be such that $b_{i,5}$ becomes a non-prefetch block while $b_{i,6}$ is a prefetch block. In our experiments, this level of complexity did not enhance average startup latency when compared with the following simple variant: Abandon the concept of $B_{Network_{i,j}}$ and compute $B_{Network_i}$ by considering all blocks of clip i, their assignment to different devices in the wireless home network, and the average download link bandwidth from these devices. If none of the neighbors contain a copy of b_{ij} then $B_{Network_{ij}}$ is the available bandwidth from the base station. Let $peer(N_k, b_{ij})$ denotes the neighbors of N_k with a copy of b_{ij}, we define $B_{Network_i}$ as: $B_{Network_i} = \frac{\sum_{\forall b_{ij} \ of \ clip \ i} B_{Network_{ij}}}{|b_{ij}|}$ where $B_{Network_{ij}} = \frac{\sum_{\forall N_l \in peer(N_k, b_{ij})} B_{N_l \rightarrow N_k}}{|peer(N_k,b_{ij})| \times |neighbor(N_k)|}$.

With Aggregate-PB, $B_{Network_i}$ is a constant for all clips referenced by a node N_k. With Peer-PB, $B_{Network_i}$ might be different for each clip i because its value depends on those neighbors of N_k with a copy of blocks of clip i (b_{ij}).

Section 6.1 compares variants of PB with one another. In a nut shell, obtained results show Moderate PB enhances average startup latency with abundant storage while Conservative PB is a better alternative with scarce amount of storage. Peer-PB is the worst because it is adversarial to Domical.

5. Simulation Study

We developed a simulation model to compare UW with PB. 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. While 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 a different replacement technique to control which clips occupy 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 [24]. Node N_i may have g edges with different bandwidths to g different nodes in the system. Moreover, we assume a shared bandwidth transmission model that captures the shared nature

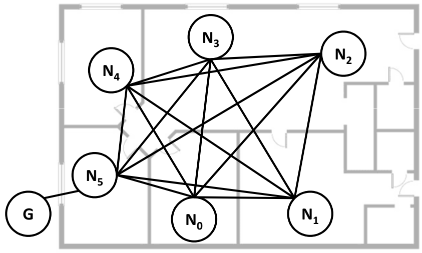


Figure 3. Real Home Network Topology

of the wireless medium [3].

We assume a working set consisting of video clips where each clip is encoded using a constant bit rate encoding technique. We ran experiments using databases consisting of homogeneous and heterogeneous collection of clips. Observed trends are consistent across all databases. Below, we focus on a simple database consisting of 864 equi-sized clips with display time of 30 minutes and bandwidth requirement of 4 Mbps. Each clip is 0.9 Gigabytes in size. The size of working set, S_{WS}, is 0.75 Terabytes. This simple database highlights the tradeoffs associated with UW and PB while removing two experimental variables, namely, clip display time and bandwidth requirement.

We use a Zipf-like distribution [4] to generate requests for different clips. This law has been used to model the distribution of web page requests [14, 4, 29], and sale of movie tickets[2] in the United States [9]. With a Zipf-like distribution, 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.

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 [8] to reserve link bandwidths. When there are multiple paths available, N_{admit} chooses the path to minimize startup latency.

The simulator conducts rounds that are repeated tens of thousands of times. A round selects nodes one at a time. The order in which nodes are selected is shifted by one in each iteration, 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_i's local storage then its display incurs a zero startup latency. Otherwise, N_i 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_i 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.

Section 6 considers the realistic home network of Figure 3 corresponding to a deployment of six nodes employing 802.11a networking cards in a British household [24] with asymmetric link bandwidths.

6. Experimental Results

In Section 6.1, we compare the PB replacement technique with alternative definitions of $B_{Network_{ij}}$ detailed in Section 4.3. These results show Moderate PB is superior to the other alternatives. In Section 6.2, we compare Moderate PB with urgency-

[2]In [9], 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 [9], we analyze 0.73 as a possible value for μ in Section 6.

$\frac{S_T}{S_{WS}}$	Conservative	Moderate	Peer-PB
0.05	344.98	388.11	516.85
0.1	209.64	264.66	407.19
0.25	86.50	110.96	243.79
0.5	25.47	24.11	69.43
0.75	11.29	7.41	16.51
1	3.87	2.22	2.55
1.25	1.52	0.86	0.99
1.5	0.83	0.67	0.67
2	0.47	0.37	0.40

Table 2. Average startup latency (Seconds), $\mu = 0.73$**.**

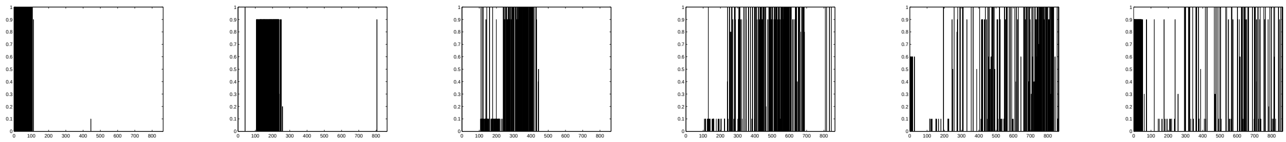
worthiness (UW) replacement. Obtained results show UW provide a better average startup latency when compared with Moderate PB. Finally, we compare urgency-worthiness (UW) with segment-based technique in Section 6.3. Obtained results show superiority of UW over segment-based technique.

6.1. A Comparison of Alternative Partial Bandwidth-based (PB) Replacement Techniques

Table 2 shows the average startup latency observed with the three alternative PB techniques. It shows Moderate PB is superior to both Conservative PB and Peer-PB when storage is abundant, i.e., high $\frac{S_T}{S_{WS}}$ ratios. With scarce amount of storage, low $\frac{S_T}{S_{WS}}$ ratios, Conservative PB provides a better average startup latency. Consider these observations in turn.

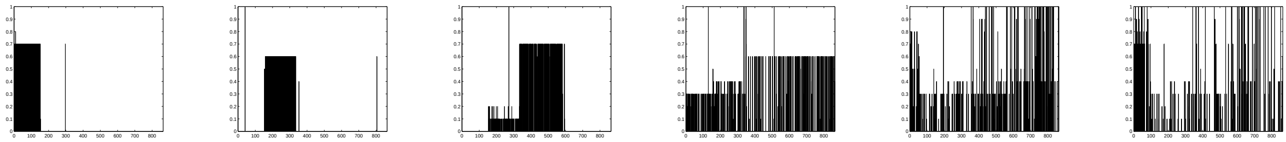
With high values of $\frac{S_T}{S_{WS}}$, Moderate PB outperforms Conservative PB because Conservative constructs prefetch portions that are extremes, consisting of either zero blocks or almost all blocks that constitute a clip. To elaborate, the lowest in-coming link bandwidth of different devices is as follows: $\{N_0 : 2\ Mbps, N_1 : 1.5\ Mbps, N_2 : 1\ Mbps, N_3 : 9\ Mbps, N_4 : 1\ Mbps, N_5 : 2\ Mbps\}$. With a homogeneous repository of clips where each clip consists of 10 blocks, Conservative computes the number of blocks per prefetch portion of each clip for each node to be: $\{N_0 : 9\ blocks, N_1 : 10\ blocks, N_2 : 10\ blocks, N_3 : 6\ blocks, N_4 : 10\ blocks, N_5 : 9\ blocks\}$. Using the whole clip as the prefetch portion of a clip does not enhance startup latency with streaming media [25, 17]. By using the average in-coming link bandwidth of network connections to a device, Moderate PB ensures the prefetch portion of each clip consists of a few blocks, $\{N_0 : 6\ blocks, N_1 : 6\ blocks, N_2 : 7\ blocks, N_3 : 5\ blocks, N_4 : 7\ blocks, N_5 : 7\ blocks\}$. This enables Moderate PB to provide an enhanced average startup latency.

Peer-PB is significantly worse than both Moderate and Conservative PB due to its adverse interaction with Domical. The explanation for this is as follows. Domical strives to balance the load across the wireless connections by streaming data from dependency group zero, N_4. Peer-PB prevents this because, every time a neighboring devices with a high amount of bandwidth (say N_3) references and caches one of the popular clips that resides on N_4, it triggers N_4 to deletes its copies of popular clips. Once this happens, Domical can no longer realize its objective to prevent formation of bottleneck links by streaming clips from N_4 to the other devices in the mesh network. This results in a high average startup latency with Peer-PB. To illustrate, Figure 4 shows the content of each node with Conservative, Moderate, and Peer-PB. The dependency groups are as follows: N_4 in d_0, N_0 in d_1, N_2 in d_2, N_1 in d_3, N_3 in d_4, and N_5 in d_5. The x-axis of this figure is the identity of clips in descending order of popularity. The y-axis is the percentage of clips resident across each device. With Moderate and Conservative PB, N_4 (the node assigned to d_0) contains the frequently accessed clips. With Peer-PB, the popular clips are



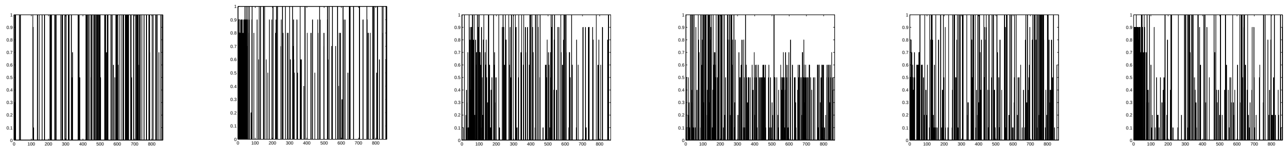
Node 4 Node 0 Node 2 Node 1 Node 3 Node 5

a. Conservative PB



Node 4 Node 0 Node 2 Node 1 Node 3 Node 5

b. Moderate PB



Node 4 Node 0 Node 2 Node 1 Node 3 Node 5

c. Peer-PB

Figure 4. Distribution of blocks across devices with Domical-PB, $\mu = 0.73$, $\frac{S_T}{S_{WS}} = 0.75$.

scattered across other devices.

With a scarce amount of disk storage ($\frac{S_T}{S_{WS}} \leq 0.25$), Conservative outperforms Moderate PB because the average cache hit rate of a device is very low, less than 10%. By caching the whole clip on each device, Conservative minimizes the likelihood of those devices that observe a cache hit from competing for the available wireless network bandwidth. With Moderate, since each device contains a fraction of a clip, even when a device observes a cache hit, it must stream non-prefetch blocks of that clip from other devices. This exhausts the available network bandwidth, resulting in a high average startup latency.

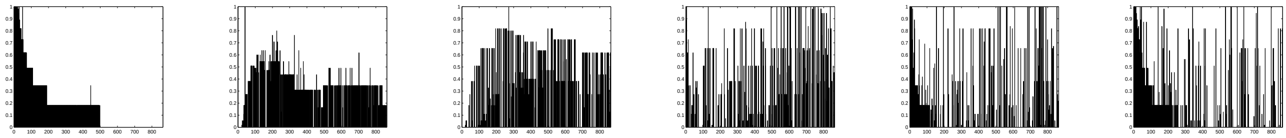
We also analyzed maximum and median as definitions of $B_{Network_{ij}}$ with PB, see Section 4.3. Maximum and median link bandwidths are 11 and 5.5 Mbps, respectively. Both values cause Equation 1 to compute zero as the number of prefetch blocks, causing the technique to resemble a clip-based LFU cache replacement policy. One may adjust Equation 1 to compute the minimum number of prefetch blocks to equal one. When network bandwidth is scarce (as in our assumed environment), this causes every node to stream remaining blocks of its referenced clip and exhaust the available bandwidth. This is undesirable because it results in a high startup latency. However, if the network bandwidth is abundant such that it can accommodate streaming of blocks to all devices simultaneously, startup latency is minimized when the prefetch portion consists of at least one block.

6.2. A Comparison of PB With UW

Table 3 shows the average startup latency with urgency-worthiness (UW) and PB with three different distributions of access to clips: Skewed with 80% of requests referencing 20% of clips (μ=0.973), moderately skewed with 60% of requests referencing 20% of clips (μ=0.73), and uniform with 30% of requests referencing 20% of clips (μ=0.25). For PB, we chose

	$\mu = 0.973$		$\mu = 0.73$		$\mu = 0.25$	
$\frac{S_T}{S_{WS}}$	UW	PB	UW	PB	UW	PB
0.05	59.08	115.28	327.11	344.98	700.18	669.4
0.1	33.45	53.85	213.67	209.65	611.29	558.54
0.25	10.33	18.91	75.3	86.5	376.21	335.65
0.5	2.75	6.59	16.37	24.12	74.78	95.21
0.75	1.04	1.73	4.67	7.41	16.57	23.36
1	0.5	0.63	1.58	2.22	4.96	6.58
1.25	0.31	0.31	0.96	0.86	2.84	4.89
1.5	0.21	0.22	0.75	0.67	2.04	1.88
2	0.12	0.1	0.44	0.37	1.29	1.13

Table 3. Average startup latency (Seconds) of Domical-UW and Domical-PB



Node 4 Node 0 Node 2 Node 1 Node 3 Node 5

Figure 5. Distribution of blocks across devices with Domical-UW, $\mu = 0.73$, $\frac{S_T}{S_{WS}} = 0.75$.

the variant that provides the best average startup latency: Conservative with $\frac{S_T}{S_{WS}} \leq 0.25$ and Moderate with $\frac{S_T}{S_{WS}} > 0.25$. Obtained results show two key observations. First, UW outperforms PB with a skewed distribution of access. For example, when μ=0.973, UW outperforms PB with $\frac{S_T}{S_{WS}} = 0.5$. Second, with a uniform distribution of access (μ=0.25) and a scarce amount of storage ($\frac{S_T}{S_{WS}} \leq 0.25$), Conservative PB is superior to UW. For example, with μ=0.25, PB is 12% better when compared with UW. We describe each in turn.

With a skewed distribution of access, Domical minimizes the amount of overlapping blocks between nodes 4, 0 and 2, see [11] for an explanation. UW caches the first few blocks of many clips onto these nodes when compared with PB, compare Figure 5 with Figures 4.a and b. For example, while these three nodes cache the first few blocks of 495 unique clips with UW, they have blocks of 108 and 154 unique clips with Conservative and Moderate PB, respectively. Moreover, nodes 4, 0 and 2 have the complementary blocks that are missing from each other's cache. This helps nodes 4, 0 and 2 to observe a lower startup latency when compared with Conservative and Moderate PB. It should be noted that same discussion hold true for rest of the nodes in the network and this enables UW to enhance average startup latency when compared with different alternatives of PB.

With a more uniform distribution of access (μ=0.25) and scarce amount of storage ($\frac{S_T}{S_{WS}} \leq 0.25$), caching data at the granularity of a clip enhances average startup latency because cache hit rate of each node is very low (less than 10%). If data is cached at the granularity of a block, even those nodes that observe a cache hit compete for the network bandwidth to retrieve their non-prefetch blocks, resulting in a high average startup latency. This explains why Conservative PB is superior to UW with μ=0.25 and $\frac{S_T}{S_{WS}} \leq 0.25$.

$\frac{S_T}{S_{WS}}$	Segment-based	UW
0.05	568.14	327.11
0.1	448.45	213.67
0.25	224.63	75.3
0.5	57.03	16.37
0.75	14.56	4.67
1	4.55	1.58
1.25	1.77	0.96
1.5	0.94	0.75

Table 4. Average startup latency (Seconds), $\mu = 0.73$**.**

6.3. A Comparison of UW With Segment-based Replacement

Table 4 summarizes the simulation results comparing performance of UW with segment-based technique using the homogeneous database of Section 5. Obtained results show that UW outperforms segment-based technique. This is because segment-based technique provides a lower utilization of disk space by evicting segments that consist of many blocks, potentially half the blocks that constitute a clip, to accommodate a segment consisting of a few blocks. This results in smaller number of unique clips in the cooperative group. For example, when $\frac{S_T}{S_{WS}} = 0.5$, the percentage of clips cached in the network is 44% with segment-based technique and 59% with UW. This enables UW to outperform segment-based technique.

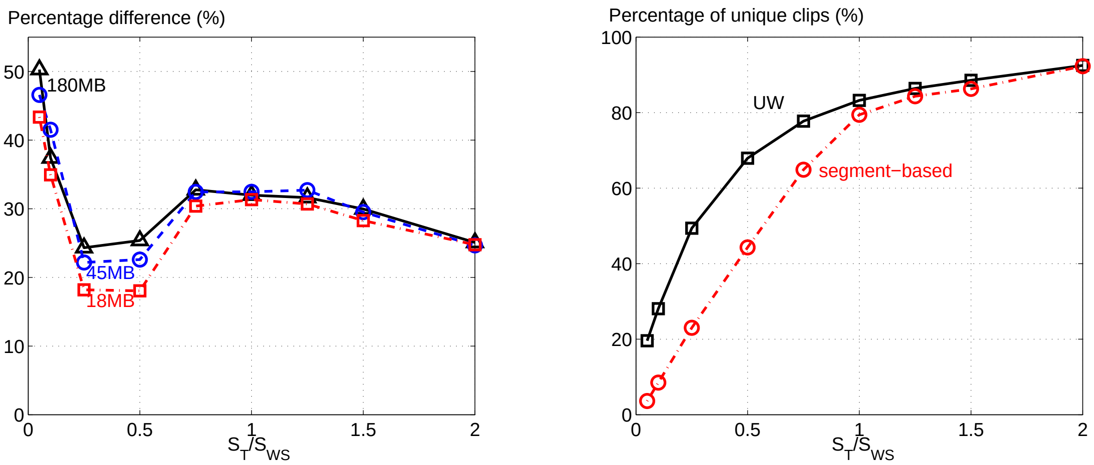
In addition, we used a heterogeneous database to compare performance of UW with segment-based technique. This database consists of 864 clips. Half are low resolution with display bandwidth requirement of 2 Mbps. The other half are high resolution with display bandwidth requirement of 4 Mbps. Clips that constitute each media type have a display time of 30, 60, and 120 minutes. Thus, the repository consists of six different clip types. The same number of clips (144) are available for each type in the repository. We analyzed UW and segment-based with three different block sizes: 18, 45, and 180 MB.

Table 5 shows UW outperforms segment-based by a large margin with a scarce amount of storage, $\frac{S_T}{S_{WS}}$=0.05, with a block size of 180 MB[3]. This difference is reduced with larger values of $\frac{S_T}{S_{WS}}$. Consider each in turn.

With a scarce amount of storage, $\frac{S_T}{S_{WS}}$=0.05, UW outperforms segment-based for three reasons. First, UW enables each node to observe a higher cache hit rate than segment-based. We measured the cache hit rate of each node and averaged it across all the nodes with both UW and segment-based. Figure 6.a shows the percentage increase in this value offered by UW. When $\frac{S_T}{S_{WS}}$=0.05, UW improves the cache hit rate by more than 40% with different block sizes. Second, UW caches a larger number of unique clips across devices because it evicts victims at the granularity of blocks (not segments). This is shown in Figure 6.b where the percentage difference between UW and segment-based is largest for the smallest value of the x-axis, $\frac{S_T}{S_{WS}}$=0.05. Third, UW evicts blocks of different clips based on the amount of data that should be streamed before display of each block. This is not the same as the metric used by the segment-based technique. The segment-based technique employs the segment-id to replace blocks.

To illustrate the third explanation, consider two clips with display time of 30 minutes. Assume these two clips have the same frequency of access and belong to two different media types. With a block size of 18 MB, the high resolution clip

[3]With a 180 MB block size, the high resolution clips with display times of 30, 60, and 120 minutes consist of 5, 10 and 20 blocks respectively. The low resolution clips consist of 3, 5, and 10 blocks.



6.a Percentage difference in cache hit rates 6.b Percentage of unique clips in the network

Figure 6. UW Vs. segment-based

consists of 50 blocks. The low resolution clip consists of 30 blocks. UW treats the blocks of these two clips in the same manner, i.e., a given block id of each clip has the same likelihood of being replaced because its bytes are needed with the same urgency. Segment-based does not do the same. It constructs 4 segments with the low resolution media type and 5 segments with the high resolution media type. It evicts the last segment of the high resolution media type (consisting of 20 blocks) before any other blocks of the low resolution media type is evicted. The number of such blocks increases as we increase the clip display time. It increases to 38 and 74 with clip display times of 60 and 120 minutes, respectively.

As we increase $\frac{S_T}{S_{WS}}$ from 0.05 to 0.25, the factor of improvement in the average startup latency observed with UW when compared with segment-based increases from a factor of 4 to 7, see Table 5. With $\frac{S_T}{S_{WS}}$=0.25, this factor of improvement starts to diminish. Beyond $\frac{S_T}{S_{WS}}$=1, UW and segment-based start to provide comparable average startup latency. There are two explanations for this. First, both techniques start to cache the same fraction of the database in the mesh network, see Figure 6.b. Second, Domical employs the bandwidth of the mesh network to compensate effectively for the lower cache hit rate observed by the segment-based technique, see Figure 6.a.

With the cache hit rate, there is a sharp drop in the percentage difference as we increase $\frac{S_T}{S_{WS}}$ from 0.05 to 0.25 because segment-based technique caches segments of most popular clips with more storage. The percentage difference between UW and segment-based increases from $\frac{S_T}{S_{WS}} = 0.5$ to $\frac{S_T}{S_{WS}} = 0.75$, see Figure 6.a, because the cache hit rate observed by the nodes located at lower levels of the hierarchy formed by Domical, i.e. Nodes 3 and 5, is lower with segment-based. While UW caches blocks of most frequently accessed clips in nodes 3 and 5, the segment-based technique caches large segments of least frequently accessed clips onto these nodes, leaving them with small amount of available storage to cache blocks of most popular clips. This causes a smaller increase in the cache hit rate of nodes 3 and 5 with segment-based in comparison with UW when we increase the available storage.

$\frac{S_T}{S_{WS}}$	Segment-based	UW
0.05	214.42	47.45
0.1	131.87	21.76
0.25	36.81	4.64
0.5	5.92	1.13
0.75	1.76	0.50
1	0.74	0.41

Table 5. Average startup latency (Seconds), Heterogeneous mix of media, $\mu = 0.73$.

7. Conclusions

Urgency-Worthiness (UW) is a simple local replacement policy for a cooperative group of devices that share their caches in a mesh network. Its objective is to minimize the average startup latency of streaming media while facilitating sharing of data using the Domical cooperative caching technique. This sharing at the edges of the network minimizes the load imposed on the shared infrastructure outside of a home, freeing resources to service other requests. When UW is compared with the Conservative Partial Bandwidth-based (PB) replacement, UW is superior when storage is abundant. With scarce amount of storage, Conservative PB is a superior alternative.

References

[1] S. Acharya and B. Smith. MiddleMan: A Video Caching Proxy Server. In *Proceedings of NOSSDAV*, 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] S. Bararia, S. Ghandeharizadeh, and S. Kapadia. Evaluation of 802.11a for Streaming Data in Ad-hoc Networks. In *ASWN, Boston, MA*, 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] 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.

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

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

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

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

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

[11] S. Ghandeharizadeh and S. Shayandeh. Cache Replacement Techniques for Streaming Media in Wireless Home Networks. USC Database Lab Technical Report 2008-05, University of Southern California, 2008.

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

[13] S. Ghandeharizadeh and S. Shayandeh. Domical Cooperative Caching: A Novel Caching Technique for Wireless Home Networks. In *ISCA SEDE, Los Angeles, CA*, June 2008.

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

[15] 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.

[16] 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.

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

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

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

[22] E. J. O'Neil, P. E. O'Neil, and G. Weikum. The LRU-K Page Replacement Algorithm for Database Disk Buffering. In *ACM SIGMOD*, pages 413–417, 1993.

[23] 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.

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

[25] R. Rejaie, M. Handley, H. Yu, and D. Estrin. Proxy Caching Mechanism for Multimedia Playback Streams in the Internet. In *Proceedings of the 4th International Web Caching Workshop*, 1999.

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

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

[28] 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.

[29] 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.

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