Domical Cooperative Caching: A Novel Caching Technique 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 home networks have become pervasive and widely deployed due to their low cost, ease of installation, and plug-and-play capabilities with consumer electronic devices. Caching of continuous media (audio and video clips) across devices has a significant impact on the average startup latency, defined as the average delay incurred from when a user references a clip to the onset of its display. In this paper, we focus on a home network consisting of a handful of devices and present a novel cooperative caching technique named Domical. Domical controls the placement of clips across devices based on their popularity, minimizing the likelihood of bottleneck links forming in a mesh network. We compare Domical with its predecessor, named Cont-Coop, showing that it enhances average startup latency with those networks that provide a high degree of connectivity among devices.

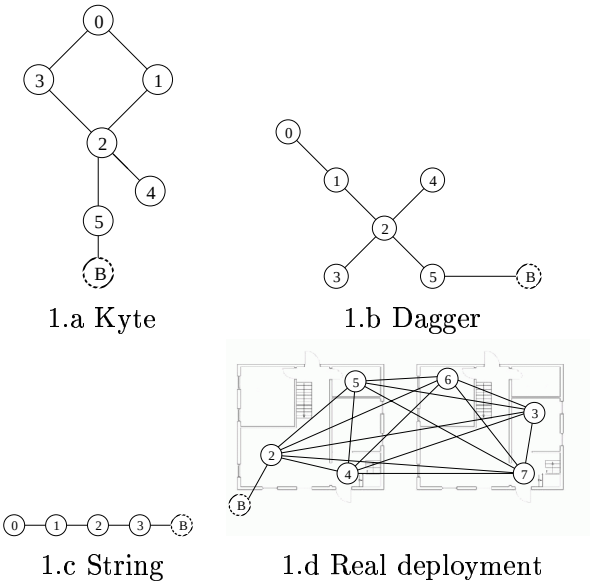
1 Introduction

The Internet has experienced an unprecedented growth in use of continuous media, audio and video clips. It is not uncommon to visit a web page and find a video clip advertising a product. Similarly, a growing number of subscribers to social networking sites such as myspace use a video clip to present their profiles. Service providers have addressed this demand for bandwidth by bringing fiber to either the curb-side or the home itself. Due to the low cost and ease of installation of a WiFi network, wireless home networks have become pervasive and widely deployed. A typical home network may consist of an access point, several PCs and laptops, and one or more electronic devices [22] such as a TV with a wireless set-top box. An example of the latter is Apple TV that synchronizes with a PC using its wireless networking card to stream audio and video clips. Such devices might be configured with mass storage[1] and set aside a fraction of their storage to cache content.

Targeting a wireless home network, in this paper, we present the design of a cooperative caching technique named Domical. Domical is a centralized technique designed to enhance the average startup latency of streaming media with a handful of devices participating in a cooperative group. Average startup latency is defined as the delay incurred from when a user references a clip to the onset of its display. Domical constructs dependencies between the caches of different devices

*A shorter version of this paper appeared in the *Proceedings of the 17th International Conference on Software Engineering and Data Engineering (SEDE)*, Los Angeles, California, June 30 to July 2, 2008.

[1] At the time of this writing, Apple TV might be configured with either 40 or 160 GB of disk storage.



1.a Kyte 1.b Dagger

1.c String 1.d Real deployment

Figure 1: Alternative topologies.

using their available network bandwidth. It constructs these groups with the objective to minimize formation of bottleneck links in the wireless home network. It is appropriate when link bandwidths are higher than the bandwidth required to display a clip [14]. When compared with a deployment that requires each device to use a greedy caching technique, we show that not only does Domical enhance the average startup latency, it also increases the likelihood of devices streaming clips to one another. This sharing of data at the edges of the network minimizes the load on the infrastructure outside the home and the remote servers.

Contributions of this study are two folds. First, we present a novel cooperative caching technique named Domical. Second, we use the topologies of Figure 1 to quantify the tradeoffs associated with Domical and its predecessor named Cont-Coop [14]. Obtained results show Domical enhances startup latency by almost 50% when network connectivity between devices is dense such as the realistic home network of Figure 1.d.

The rest of this paper is organized as follows. After describing the related work in Section 2, we present two alternative cooperative caching techniques named Cont-Coop [14] and Domical. Section 4 compares these two alternatives with one another and a deployment that requires devices to employ a greedy caching technique. Brief conclusions and future research directions are presented in Section 5.

2 Related Work

One may categorize alternative caching techniques for streaming media based on whether they are greedy [21, 19, 13] or cooperative [1, 5, 14]. A technique may strive to enhance startup latency [17, 14] or some other metric such as cache hit ratio or distribution of load across the participating cache servers [1, 24, 28, 6, 17, 5, 13], categorizing the techniques into branches. Domical is a cooperative technique designed to enhance average startup latency. It may realize this by increasing the load imposed on a device with abundant network bandwidth. This differentiates Domical from MiddleMan [1] and Silo [5] because they strive to balance the load imposed on participating

devices. It is important to note that one may deploy Domical in the centralized coordinators of MiddleMan [1], Middleware of [16], and proxy cluster of Silo [5].

Domical is an extension of our prior work on Cont-Coop [14]. Both are designed for wireless home networks with the objective to enhance average startup latency. Hence, they share many similarities, see Section 3. A key difference between the two is that Domical imposes greater constraints on placement of data by constructing many dependency groups.

3 Two Cooperative Caching Techniques: Domical and Cont-Coop

This section presents two cooperative caching techniques, Domical and Cont-Coop. With both, when a user references a clip X using a device and X is not resident in the local cache of the device, the device admits X into its cache by ensuring it has sufficient space to store X. This enables the device to service other short term correlated references to X [20, 23] using its local cache. Below, we provide an overview of each technique, outlining their similarities. Section 3.1 describes a component that is common between Cont-Coop and Domical. Subsequently, Sections 3.2 and 3.3 detail Domical and Cont-Coop in turn.

Both caching techniques preserve autonomy of a device while allowing it to cooperate with other devices once it has been in their radio range for a while. This is realized as follows. When a device is isolated and its cache space does not accommodate X then it employs a greedy cache management technique such as DYNSimple [13] or GreedyDual [3] to choose victims to swap out of its cache. These techniques are different than LRU because they utilize the size of clips when choosing victims. Without loss of generality and in order to simplify the discussion, we assume a device employs DYNSimple because it was shown to provide a higher hit ratio than GreedyDual [13]. With this technique, it chooses those clips with the lowest $\frac{f_i}{S_i}$ values where f_i is the frequency of access to a clip and S_i is the clip size.

When in radio range of other devices, a device may build a profile of its network connectivity with its neighboring devices and the bandwidth of these links. Next, it employs either Domical or Cont-Coop to compute a dependency group between itself and the different devices. It may exchange this information with other participating devices and employ a voting mechanism to identify a common dependency group. The dependency group specifies which device should make the state of its cache dependent on other devices. When a device N_i makes the state of its cache dependent on a set of devices N_0, N_1, ..., N_{i-1}, its selection of victim clips is modified as follows. N_i computes which of its clips are identical to those occupying the caches of N_0, N_1, ..., N_{i-1}. This can be implemented in a variety of ways using either a lazy or an eager approach and estimation techniques such as the use of Babb filters, see Section 3.4. Next, it sorts the common clips using DYNSimple and swaps as many of these as possible to accommodate the in-coming clip X. Once all these clips are swapped out, if there is insufficient space to accommodate X then N_i applies DYNSimple to remaining clips (unique to the cooperative group) in its cache until sufficient amount of space is freed up.

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 nodes from becoming a bottleneck. This is important because streaming of 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. A cooperative caching technique that prevents such scenarios, enhances the average startup latency incurred by the cooperative group of devices.

Term	Definition
Startup latency (δ)	Delay from when a user references a clip to the onset of display.
Throughput	Number of devices displaying their referenced clips simultaneously.
Cache hit rate (ω)	The percentage of clip requests satisfied using the cache.
i^{th} dependency group (h^i)	Group of nodes that state of their caches depend on state of caches in $h^0, \ldots, h^i$.
Core node	A node whose cache state is independent of other nodes and is a member of h^0.
$\mathcal{N}$	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

Table 1: Terms and their definitions

3.1 Bandwidth contention ratio (ζ)

In this section, we develop the bandwidth contention ratio metric, $\zeta(N_i)$, which estimates the amount of imbalance across the network links when N_i streams a clip to every other node in the network. Intuitively, the node with the smallest ζ value results in the lowest imbalance, avoiding formation of hot spots and bottlenecks. Domical and Cont-Coop use this metric to compute which nodes should cache the popular clips by enforcing dependencies between the caches of different nodes, see Sections 3.2 and 3.3.

The pseudo-code of Figure 2 quantify ζ for each node. It captures two important details. First, a node N_j may stream its clip from the infrastructure outside of the home, denoted as B in Figure 1, when it participates on the path connecting N_j to N_i. Second, the link bandwidths from N_i to its neighboring devices are asymmetric. The later is realized by normalizing the available link bandwidth using $\frac{1}{Bandwidth\ of\ L_k}$ for each link L_k that participates in a path from N_i. The resulting standard deviation quantifies $\zeta(N_i)$. Nodes with smallest ζ values are ideal candidates to cache popular clips because they can stream these clips while minimizing formation of bottleneck links.

3.2 Domical Cooperative caching

Domical Cooperative caching (Domical) uses bandwidth contention ratio to form a hierarchy of cache dependencies. It starts by computing the bandwidth contention ratio for each node, ζ_i. Next, it sorts nodes based on their ζ values such that, $\zeta_i < \zeta_j$ for nodes i and j. Domical constructs dependency groups of nodes with each group consisting of a single node[2]. The node with value ζ_i becomes member of cache dependency group h^i. The node in dependency group h^j makes the state of its cache dependent on those nodes appearing in groups h^0 to h^{j-1}. The node in group h^0 employs a greedy caching strategy, DYNSimple, and the state of its cache is independent of all nodes. The node in group h^1 makes the state of its cache dependent on the node in group h^0.

[2]The concept of a group is useful to compare Domical with Cont-Coop in Section 3.3.

```
ζ(N_i)
{
 Let B denote the infrastructure outside the home;
 L_Max = Link with the highest bandwidth;
 Initialize the weight of all links in the network to zero;
 For each remaining node N_j do
   {
     i) Let P be the set of all possible shortest paths from N_j to N_i that
          does not involve the base station;
     ii) For each link L_k in the network do
     {
          Let q denote the number of paths in P that include L_k;
          Increment link weight of L_k by q/P × 1/(Bandwidth of L_k);
     }
   }
   ζ = standard deviation in the link weights;
   return ζ;
 }
}
```

Figure 2: Bandwidth contention ratio of N_i, $\zeta(N_i)$.

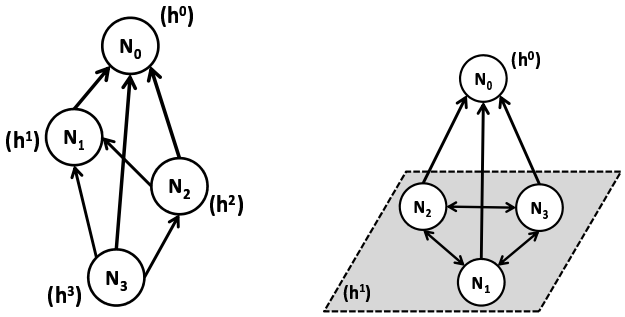
Assuming $\mathcal{N}$ nodes in the network, the node with the largest ζ value appears in the dependency group $h^{\mathcal{N}-1}$ and the state of its cache is dependent on all other nodes.

Figure 3.a shows Domical forming four groups: h^0 to h^3. The dependency group h^0 contains node N_0 because its ζ value is the lowest amongst all the four nodes. To simplify the discussion, in this example, we chose the subscript of each node to be the same as their dependency group. Typically, the identity of the node is independent of the identity of its dependency group, see Section 4.2.

Domical causes those clips with highest probability of access to be cached in group h^0 and the clips with lowest probability of access to be cached in group h^{n-1}. Results of Section 4 show this provides a lower startup latency with the realistic home network of Figure 1.d by minimizing bandwidth contention when streaming clips due to cache misses.

3.3 Cont-Coop

Cont-Coop [14] constructs two dependency groups h^0 and h^1, see Figure 3.b. Similar to Domical, it computes ζ for the different nodes. The node with the minimum ζ value is termed the core node and assigned to h^0. It employs a greedy cache replacement policy, DYNSimple. Remaining $\mathcal{N}-1$ nodes are assigned to h^1. They cooperate by making the state of their caches dependent on one another and the core node. Figure 3.b shows the dependency groups constructed for the four node network using Cont-Coop. N_1's cache depends on the state of the caches of all three nodes in the network. This is in sharp contrast with Domical where N_1's cache depends on the state of N_0's cache only.



3.a Domical 3.b Cont-Coop

Figure 3: Different cache dependency groups.

3.4 Implementation in a distributed manner

In a wireless home network consisting of a handful of wireless devices, the bandwidth contention ratio can be calculated by each individual node. A node may constantly monitor its available bandwidths to its neighboring nodes using a packet dispersion [10] based technique, such as Cap-Probe [18]. It may periodically broadcast its information to all the other nodes in the network, enabling each node to construct an approximate topology of the network. Next, each node N_i calculates its $\zeta(N_i)$. Then a master-slave scheme [12] can be used to force nodes to form the cache dependencies based on the cooperative caching policy used in the cooperative group.

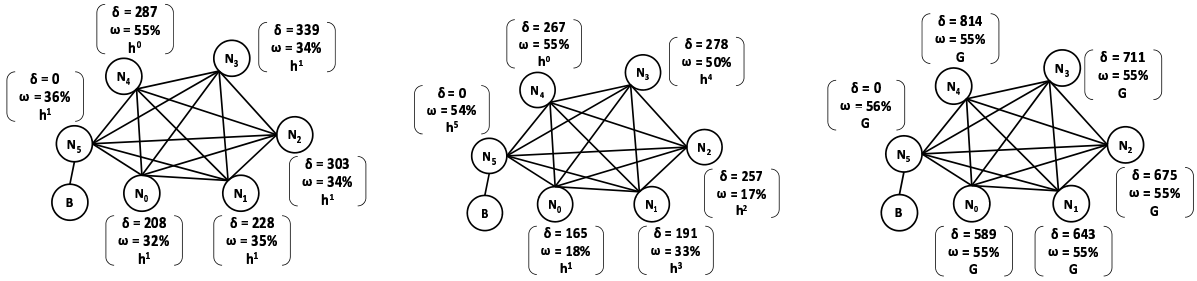
To synchronize the state of its cache with other nodes, a node must discover clips residing in other nodes of the cooperative group. The initial discovery process might be implemented flooding technique of an unstructured P2P network [7], or one of the discovery techniques employed by cooperative proxy caching techniques [25, 26]. Subsequently, the node may monitor clip referenced by its neighboring devices to incrementally update its profile information about the content of other devices. In order to further refine this information, a node may double check its profile of other nodes as follows. It may apply a hash function to the ID of a clip building a Babb filter of bits for a node. It may send this filter to the target node which repeats the process. If the Babb filters do not match then the node sends the identity of its clips to the requesting node.

4 Comparison

In this section, we compare Domical with Cont-Coop using a simulation study. We start with a brief overview of the simulation model. Subsequently, Section 4.2 presents obtained results using the characteristics of a British home network described in [22].

4.1 Simulation Model

We developed a simulation model to compare Domical with Cont-Coop. 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. In this study, we use Dynamic Simple (DYNSimple).



4.a Cont-Coop ($\bar{\delta} = 228$) 4.b Domical ($\bar{\delta} = 194$) 4.c Greedy ($\bar{\delta} = 573$)

Figure 4: A real home network, $\frac{S_T}{S_{WS}} = 1$, $\mu = 0.73$.

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 [22].

Node N_i may have g edges with different bandwidths to g 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 g 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 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. This study assumes Shared-BW transmission model.

We assume a working set consisting of video clips where each clip is encoded using a constant bit rate encoding technique. The display bandwidth requirement of each clip is 4 Mbps. We assume number of clips is 864 clips grouped into 3 categories with display times of 2 hours, 60 minutes, and 30 minutes. The size of these clips are 3.5 Gigabytes (GB), 1.8 GB, and 0.9 GB, respectively. The size of working set, S_{WS}, is 1.85 Terabytes.

We use a Zipf-like distribution [2] to generate requests for different clips. To elaborate, Zipf's law [29] 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 $\chi = \frac{1}{\sum_{i=1}^{C} \frac{1}{i^{\mu}}}$. This law has been used to model the distribution of web page requests [15, 2, 27], and sale of movie tickets[3] 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

[3]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 4.

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_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.

In each iteration, we measure the following parameters for each node: incurred startup latency, cache hit rate. Each experiment is based on ten thousand iteration of each round. 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.

We investigated alternative topologies shown in Figure 1 with link bandwidths ranging from 2 to 16 Mbps. Section 4.2 considers a realistic network corresponding to a deployment of six nodes employing 802.11a networking cards in a British household [22] with asymmetric link bandwidths.

4.2 A realistic home network: Domical outperforms Cont-Coop

We employed the characteristics of one of the British households reported in [22] to compare Domical with Cont-Coop. This network consists of six devices as shown in Figure 1.d. We assume the sum of the cache sizes, S_T, equals the working set size, S_{WS}. Moreover, the distribution of access to the clips that constitute the working set is skewed, $\mu = 0.73$. (Figure 5 shows the results with different $\frac{S_T}{S_{WS}}$ ratios and various distributions of access (μ values); see below for a discussion.) The average startup latency with Cont-Coop is 228 seconds, and Domical is 194 seconds. Domical provides an improvement of 15%. If each device used a greedy replacement policy, the incurred startup latency would have been 573 seconds. This is almost a 3 fold increase relative to Domical and more than a two fold increase relative to Cont-Coop. Domical outperforms Cont-Coop for two reasons. First, by constructing a hierarchy of nodes, it controls placement of popular clips across devices to minimize formation of bottleneck links. Second, the high connectivity between nodes enables Domical to benefit from its constructed hierarchies. In Section 4.3, we show that if the connectivity between the nodes is minimized, Domical might become inferior to Cont-Coop.

Figure 4 shows the average startup latency (δ) for each node and its hit ratio (ω) with the alternative caching strategies. Below these two values, we show the dependency group of a node. We start by discussing the incurred startup latency. With Greedy, the average startup latency (δ) is in excess of 500 Seconds with all nodes except N_5 because we assume the wireless connection between N_5 and the base station is very high. Both Cont-Coop and Domical enhance the average startup latency of each node by: a) maintaining a larger number of unique clips in the home network, and b) utilizing the high bandwidth of the wireless network connections to stream clips among the participating devices. This results in the following phenomena. For all nodes except N_5, the average number of requests issued by a device that are serviced by its neighbors are as follows: 1) with Domical, it is 46%, 2) with Cont-Coop, it is 43%, and 3) with Greedy, it is 3%.

Thus, Greedy forces wireless network connections to remain idle, wasting their bandwidth.

With Greedy, each node observes a 55% hit ratio because each materializes the popular clips in its cache. With both Domical and Cont-Coop, only N_4 observes a 55% hit ratio. This is because this node belongs to dependency group h^0 and employs DYNSimple to materialize the popular clips. An interesting result in Figure 4.b is the hit ratio of nodes with Domical. They are dramatically reduced to 18% for nodes in h^1 (N_0) and h^2 (N_2). It increases to 34% for the node in h^3 (N_3) and continues to increase in excess of 50% with nodes in h^4 and h^5. The explanation for this is as follows. Conceptualize the clips sorted based on their $\frac{f_i}{S_i}$ values. Partition this sorted list into five consecutive segments: Segment one consists of the most popular clips, the second contains the second group of most popular clips, so on and so forth. N_4 caches the most popular clips (segment 1) because it employs DYNSimple as its greedy caching strategy. The state of node N_0 depends on N_4 because it is in h^1. Every time N_0 references a clip in Segment 1, it streams it from N_4 and caches it. At the same time, N_0 may reference a clips in other segments with 0.4 probability. Every time this happens, if N_0 must choose a victim clip then it swaps out one of the popular clips in Segment 1 because its state depends on N_4 and N_4 will almost certainly have this clip in its cache. The same happens with node N_2 which belongs to dependency group h^2. However, the impact of this phenomena is minimized with h^3, h^4, and h^5. To illustrate, consider N_5 which belongs to h^5 and the state of its cache depends on all nodes. This node will access clips in Segments 1 to 4 very frequently, streaming them from devices in h^0 to h^4, and caching them. It rarely references clips in Segment 5 because they are very unpopular with 7% probability. When it does, Domical will force this node to swap out those clips with lowest $\frac{f_i}{S_i}$ that also reside in the cache of those devices in h^0 to h^4. Most likely, clips in Segments 4 and 3 are swapped out because they have a lower $\frac{f_i}{S_i}$ value when compared with the other Segments. Since these are not as popular as clips in Segment 1, N_5 references them with a low probability, allowing N_5 to observe a cache hit for the Segment 1 clips residing in its cache.

It is worthwhile to note that Domical outperforms Cont-Coop with a larger cache and **not** a more skewed distribution of access. This is shown in Figure 5 where we plot the percentage improvement offered by Domical when compared with Cont-Coop ($\frac{\delta_{Cont-Coop}-\delta_{Domical}}{\delta_{Cont-Coop}}$) for different $\frac{S_T}{S_{WS}}$ values. The different curves pertain to different distributions of access to clips, very skewed (μ=0.973) to more uniform (μ=0.001). The percentage improvement with Domical increases with very large cache sizes. With smaller cache sizes, there is little improvement offered by Domical because many of clip references must be streamed from the base station with both Domical and Cont-Coop. In this case, both Domical and Cont-Coop outperform a deployment that employs a greedy caching strategy by 30%.

4.3 Network Topology and Connectivity

The degree of network connectivity between devices has a significant impact on the performance of Domical. Figures 6 and 7 show the average startup latency observed with the Kyte and dagger topologies, see Figure 1. The Kyte topology has less connectivity than the realistic topology of Figure 1.d, closing the large gap between the cooperative techniques and the Greedy caching strategy. This gap becomes even smaller with the Dagger topology which has only one less connectivity than the Kyte topology, see Figure 1.

In addition, while Domical is superior to Cont-Coop with the Kyte topology, it becomes inferior to Cont-Coop with the Dagger topology. The explanation for this is as follows. The topological difference between the Dagger and Kyte is the network connection between N_0 and N_3, causing a different node to be chosen for h^0. The assignment of nodes to different dependency groups with

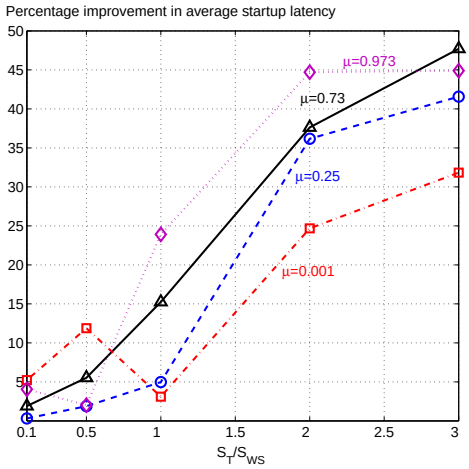
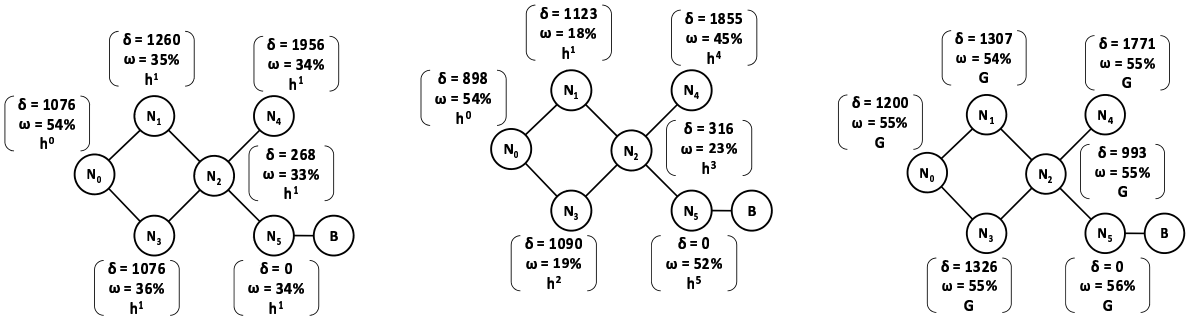
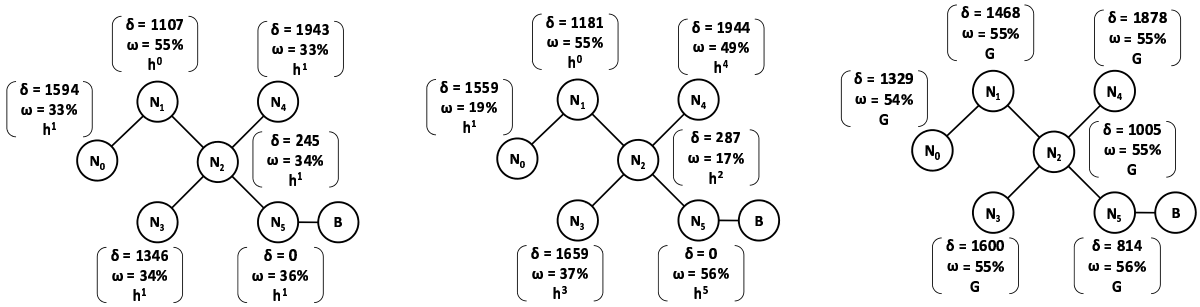


Figure 5: Percentage improvement in startup latency ($\frac{\delta_{Cont-Coop}-\delta_{Domical}}{\delta_{Cont-Coop}}$), real deployment topology.



4.a Cont-Coop ($\bar{\delta} = 939$) 4.b Domical ($\bar{\delta} = 880$) 4.c Greedy ($\bar{\delta} = 1100$)

Figure 6: Kyte topology, $\frac{S_T}{S_{WS}} = 1$, $\mu = 0.73$.



4.a Cont-Coop ($\bar{\delta} = 1039$) 4.b Domical ($\bar{\delta} = 1105$) 4.c Greedy ($\bar{\delta} = 1213$)

Figure 7: Dagger topology, $\frac{S_T}{S_{WS}} = 1$, $\mu = 0.73$.

Domical for each topology is as follows:

- Kyte topology: $N_0 \in h^0, N_1 \in h^1, N_3 \in h^2, N_2 \in h^3, N_4 \in h^4, N_5 \in h^5$.
- Dagger topology: $N_1 \in h^0, N_0 \in h^1, N_2 \in h^2, N_3 \in h^3, N_4 \in h^4, N_5 \in h^5$.

For each topology, the node assigned to h^0 by Domical is also chosen by Cont-Coop as the core node. Cont-Coop assigns the remaining nodes to h^1.

With the Kyte topology, the node assigned to h^0 (N_0) observes a better average startup latency with Domical when compared with Cont-Coop. This is because its neighboring devices (N_1 and N_3) service a larger fraction of its cache misses by caching the 2nd and 3rd group of most popular clips (caused by their assignment to dependency groups h^1 and h^3). Cont-Coop may retrieve its cache misses from a node such as N_4 because it does not control the caching of the 2nd and 3rd group of most popular clips.

With the Dagger topology, concentrating the 2nd group of most popular clips on node N_0 is a mistake because it is isolated and its network connectivity to other nodes is through node N_1. This results in formation of bottlenecks, causing Domical to become inferior to Cont-Coop.

5 Conclusion and future research directions

This study presents a novel cooperative caching technique named Domical for a wireless home network consisting of a handful of devices. When compared with Cont-Coop, Domical provides a better average startup latency for those networks with a high degree of connectivity between devices. In addition to topologies reported in Figure 1, we studied other topologies consisting of 4, 5, and 6 nodes, providing the same general observations.

In the near future, we intend to extend Domical in several ways. First, we plan to consider larger networks consisting of tens of nodes deployed in an office setting [11]. We are interested in determining the thresholds for bandwidth contention ratios that trigger a node to decide not to participate in a cooperative group. Moreover, we intend to explore the granularity of data items cached among devices. One possible design is to break a clip into chunks and cache at the granularity of a chunk. Such a technique is considered in [4, 16]. This study constructs replicas of segments for the purposes of fault tolerance and load balancing. We plan to investigate extensions of its proposed designs to a wireless home network by exploring its availability and startup latency

characteristics. A technique such as Domical which caches at the granularity of clips would serve as a yard stick to evaluate the tradeoffs associated with this alternative design.

References

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

[2] 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.

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

[4] Y. Chae, K. Guo, M. Buddhikot, S. Suri, and E. Zegora. Silo, Rainbow, and Caching Token: Schemes for Scalable, Fault Tolerant Stream Caching. *IEEE Journal on Selected Areas in Communications on Internet Proxy Services*, Sept. 2002.

[5] Y. Chae, K. Guo, M.M. Buddhikot, S. Suri, and E.W. 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] E. Cohen and S. Shenker. Replication Strategies in Unstructured Peer-to-Peer Networks. In *Proceedings of the ACM SIGCOMM*, August 2002.

[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 Multimedia Conference*, October 1994.

[10] C. Dovrolis, P. Ramanathan, and D. Moore. Packet-dispersion techniques and a capacity-estimation methodology. *IEEE/ACM Trans. Netw.*, 12(6):963–977, 2004.

[11] R. Draves, J. Padhye, and B. Zill. Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks. In *MobiCom*, Sept 2004.

[12] J. Dollimore G. Coulouris and T. Kindberg, editors. *Distributed Systems: Concepts and Design.* Addison-Wesley, Pearson Education, 2001.

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

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

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

[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] R. Kapoor, L.J. Chen, L. Lao, M. Gerla, and M. Y. Sanadidi. Capprobe: a simple and accurate capacity estimation technique. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 67–78, New York, NY, USA, 2004. ACM.

[19] 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.

[20] 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.

[21] 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.

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

[23] J. T. Robinson and M. V. Devarakonda. Data Cache Management Using Frequency-Based Replacement. In *Sigmetrics Conference*, pages 134–142, 1990.

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

[25] M. Ségura-Devillechaise, J. Menaud, N. Loriant, R. Douence, M. Südholt, T. Fritz, and E. Wuchner. Dynamic Adaptation of the Squid Web Cache with Arachne. *IEEE Software*, 23(1):34–41, 2006.

[26] D. Wessels and K. Claffy. ICP and the Squid Web cache. *IEEE Journal on Selected Areas in Communication*, 16(3):345–357, 1998.

[27] 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.

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

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