An Evaluation of Two Policies for Placement of Continuous Media in Multi-hop Wireless Networks*

Shahram Ghandeharizadeh, Tooraj Helmi, Taehee Jung, Shyam Kapadia, Shahin Shayandeh
Computer Science Department
University of Southern California
Los Angeles, California 90089

Abstract

This paper focuses on a greedy data placement strategy for a mesh network of peer-to-peer devices that collaborate to stream continuous media, audio and video clips. The greedy placement strategy, termed Simple, strives to maximize the number of references served by the local storage of a peer. We analyze two policies to realize this placement: Frequency-based and Byte-hit. The first sorts clips based on their access frequency, assigning the frequently accessed clips to each node. Byte-hit computes the frequency of access to each byte of a clip, sorts clips based on this value, and assigns those with the highest byte-hit value to each node. Both have the same complexity and almost identical implementations. A simulation study shows Byte-hit is superior to Frequency-based for two reasons. First, it maximizes the number of peers that can simultaneously display their referenced clips. Second, it is more robust to error in access frequencies. In all our experiments, Byte-hit performs almost identical to the optimal placement strategy.

1 Introduction

Advances in communication and processing have made ad-hoc networks of wireless devices a reality. A device, termed a peer, is typically configured with a fast processor, a mass storage device, either one or several types of wireless networking cards [20, 11]. Peers strive to provide their users with ubiquitous access to their data everywhere and are building components of complex systems such as Memex [4] and MyLifeBits [12].

In this study, we focus on devices configured with wireless networking cards that offer bandwidths in the order of a few Mega bits per second (Mbps) with a limited radio range, a few hundred feet. Examples include 802.11a, b, and g networking cards [3, 2, 19]. Devices communicate with each other when they are in the radio range of one another. Some devices might be in the radio range of a WiMax (802.16a) base station or an access point, providing the ad-hoc network with access to the wired infrastructure.

In multi-hop wireless networks, two peers may communicate through a number of intermediate peers that collaborate to relay information. An interesting subclass of networks, termed mesh networks, is more constrained by bandwidth and storage limitations and less constrained by node mobility and power consumption. An example might be Canada's SuperNet [6] which delivers educational material to remote areas in Alberta. It consists of wireless peers that complement the wired infrastructure. Another application might be an office setting [20] or a home entertainment system [14] where multiple peers collaborate to provide on-demand access to content. With the latter, a household may store its personal video library on a peer for retrieval everywhere, e.g., when visiting a friend's home. A peer may encrypt its content to either protect

*A shorter version of this paper appeared in the *Twelfth International Conference on Distributed Multimedia Systems (DMS 2006)*, Grand Canyon, Aug 30-Sept 1, 2006.

it from un-authorized access, i.e., authentication, or implement a business model for generating revenues. These mesh networks raise a host of privacy and security issues [5] which constitute an active area of research by several communities and beyond the focus of this paper.

A challenge of these ad-hoc networks is how to display continuous media, audio and video clips. Continuous media consists of a sequence of quanta, either audio samples or video frames, that convey meaning when presented at a pre-specified rate [13, 18]. Once the display is initiated, if the data is delivered below this rate then the display may suffer from frequent disruptions and delays termed hiccups.

A peer may store clips in its local storage in anticipation of future references either by its user or a neighboring peer. Many studies from mid 1990s describe techniques for a hiccup-free display of a clip that resides in the local storage of a peer, [16, 13] to name a few. One may use one of these techniques when a user references a clip resident on local storage of a peer. Otherwise, this peer (denoted D_d) must locate a peer that contains the referenced clip (denoted D_p). Next, it must admit itself into the system by reserving one or more paths from D_p to D_d in order to stream the referenced clip at a pre-specified bandwidth. This process is termed admission control. It involves collaboration of other peers serving as intermediate application-based routers by requiring them to reserve their network bandwidth on behalf of this stream. By reserving bandwidth along a path, D_d overlaps its display with delivery of data from D_p, minimizing the latency observed by a user.

A peer may set aside a fraction of its available storage to store audio and video clips. Without loss of generality and in order to simplify discussion, the term storage refers to this fraction. The Simple data placement strategy strives to maximize number of references made by a peer to clips resident in its storage. It realizes this objective by assigning as many clips as possible to a peer. This assignment is trivial when the storage capacity of each peer is larger than the repository size because one may assign the entire repository to each peer. Otherwise, the greedy algorithm must determine the identity of clips that should be assigned to each peer. We present two policies to address this research topic. They are termed Frequency-based and Byte-hit. Both assume an estimate of frequency of access to clips is available. We compare these two policies by quantifying the number of peers that may display their referenced clips simultaneously. This is also termed the throughput of the system.

Obtained results provide the following key insights. First, when the estimated frequency of access to clips is precise, Byte-hit enables a larger number of requests to find their referenced titles locally, maximizing the throughput of the system. Second, when the estimated frequency of access is not precise, Byte-hit is more robust and continues to provide a higher throughput. Third, Byte-hit has the same complexity as the Frequency-based techniques with an almost identical implementation. These provide a convincing case for a system designer to employ the Byte-hit policy.

The rest of this paper is organized as follows. Section 2 provides a survey of the relevant literature. In Section 3, we present the details of Simple and its alternative policies. Section 4 presents a comparison of the alternative policies using a simulation study. Brief conclusions and future research directions are presented in Section 5.

2 Related Work

Placement of data in ad-hoc networks is an emerging area of research. We categorize prior studies into those that share or do not share storage. When storage is shared, a design strives to enhance a global performance metric that impacts all peers. When storage is not shared, a design strives to enhance a metric local to each peer. Simple does not share storage and its outlined policies strive to enhance the number of user requests satisfied using the local storage of a peer. All prior studies assume storage is a shared resource. Below, we summarize these studies.

Studies that assume storage of a peer is shared include [21, 7, 1]. The global performance metric of

Parameter	Definition
$B_{Display}$	Bandwidth required to display a clip
B_{Link}	Bandwidth of a link
C	Number of clips, $1 \leq i \leq C$
f_i	Frequency of access to clip i
$\mathcal{N}$	Number of peers
$S_{C,i}$	Size of clip i
S_{DB}	Size of the database, $S_{DB} = \sum_{i=1}^{C} S_{C,i}$
S_N	Storage capacity of a peer
S_T	Total storage capacity of peers, $S_T = \mathcal{N} \cdot S_N$

Table 1: Parameters and their definitions

Clip id	$S_{C,i}$	f_i	$f_i/S_{C,i}$
1	14	0.7	0.05
2	1	0.1	0.1
3	20	0.2	0.01

Table 2: An example showing Byte-hit may not always yield the Optimal solution.

each study includes: average search size for a clip [7], euclidian distance between a client and a server [21], and number of simultaneous displays [1]. With [7, 1], the number of replicas for a clip is proportional to the square root of its frequency of access. With [21], this number is proportional to $\frac{p_i^{0.667}}{clip\ size}$ where p_i is the frequency of access to clip i. Simple does not compute the number of replicas for a clip. A comparison of Simple with [21, 7, 1] is a future research topic, see [15].

We focus on Simple in order to (1) identify which of its policies is superior, and (2) use the superior policy for comparison with those techniques that share storage. A key contribution of this paper is to show superiority of Byte-hit. See [15] for a preliminary comparison of Byte-hit with a storage sharing technique.

3 Simple Data Placement

We assume an ad-hoc network of $\mathcal{N}$ peers and a base-station. The base-station provides access to the wired infrastructure and remote servers. It might be in the radio range of a few peers.

When a user employs a peer to reference a clip, the peer checks to see if the clip is available in its local storage. In this case, it initiates the display of the clip from its local storage. Otherwise, it tries to admit itself to stream the clip from either a peer containing the referenced clip or the base station.

Simple assigns clips to peers with the objective to maximize the number of simultaneous displays while minimizing the incurred startup latency. Startup latency is defined as the delay incurred from when a peer references a clip to the onset of its display. An effective heuristic is to maximize the number of requests serviced using the local storage of a peer, minimizing the demand for the network bandwidth.

With a homogeneous repository consisting of equi-sized clips, the placement of data is similar to the fractional knapsack problem [8]. When the objective is to maximize the frequency of access to the local data stored on each peer, defined as $\sum_{i \epsilon S} f_i$, the optimal solution is to assign the S most popular clips to each peer. Appendix A details this solution and proves its optimality.

A repository consisting of a mix of media types is more realistic. In this case, the size of clips will be different, changing the placement of data with Simple into a 0-1 knapsack problem. This problem has been well studied in the literature. A dynamic program to realize the maximum hit ratio is outlined in [8]. This solution is a pseudo-polynomial algorithm with $O(C \cdot S_N)$ complexity. Note that the granularity of S_N is in

bytes. When the storage capacity of a node is in the order of hundreds of Gigabytes, this optimal algorithm may incur a long execution time.

There exists suitable heuristics to approximate the optimal solution. An effective heuristic is to (1) sort clips based on their $f_i/S_{C,i}$ value, (2) assign the first S clips to each node where $\sum_{i \epsilon S} S_{C,i} \leq S_N$. The process is repeated until storage of a peer, S_N, is exhausted i.e. $S_N - \sum_{i \epsilon S} S_{C,i} < S_{C,j}$ for all $j \epsilon S'$. This is the **Byte-hit** policy. Another heuristic, termed **Frequency-based**, is to repeat the above procedure using f_i instead of $f_i/S_{C,i}$.

Section 4 shows Byte-hit approximates an optimal assignment. However, Byte-hit may not always yield the optimal assignment as illustrated by the following example. Assume $S_N = 14$ and a repository of 3 clips. Table 2 shows the size of each clip and its frequency of access. The Byte-hit policy assigns clip 2 to each peer, observing a 10% hit ratio. The Optimal assignment would assign clip 1 to each peer, increasing this hit ratio to 70%. While the reader may observe the Frequency-based policy would produce the optimal solution, it is important to note that one may devise counter examples to show that Frequency-based also does not always produce an optimal assignment.

4 A Comparison

In this section, we compare the Frequency-based and Byte-hit policies. We start with a description of our simulation environment. Next, we show Byte-hit provides a higher throughput when the assumed frequency of access to clips is precise. Section 4.3 shows Byte-hit is more robust when the frequency of access to clips is erroneous. Finally, Section 4.4 analyzes scenarios with two demographics. It shows Frequency-based observes higher improvement. However, Byte-hit continues to outperform Frequency-based.

4.1 Simulation Set-up

We consider a heterogeneous repository consisting of two media types: audio and video clips with display bandwidth requirement ($B_{Display}$) of 300 Kbps and 4 Mbps. There were 3 different clip sizes for each media type. With video, we have clips with a display time of 2 hours, 60 minutes, and 30 minutes. The size of these clips are 2 Gigabytes (GB), 1 GB, and 0.5 GB, respectively. With audio, the clip display times are 4 minutes, 2 minutes, and 1 minute. Resulting clip sizes are 9 Megabytes (MB), 4.5 MB, and 2.25 MB.

With a database of C clips (say C=100), the clips are numbered sequentially from 1 to 100. Odd numbered clips are video and even numbered clips are audio. Thus, the pattern of clip sizes is 2 GB, 9 MB, 1 GB, 4.5 MB, 0.5 GB and 2.25 MB. This pattern repeats itself until all 100 clips that constitute the repository are constructed. This numbering is important because distribution of requests across clips is generated using a Zipfian distribution with a mean of 0.27. This distribution is shown to resemble the sale of movie tickets in the United States [10].

We analyzed two different network topologies: String and Grid. With String, each peer (except the two peers at the end of the topology) is in the radio range of two neighboring peers. This might represent a community of houses lined up in front of a lake. With Grid, a peer is in the radio range of those peers with a Manhattan distance of one. This means each peer, except for those on the border of the Grid, is in the radio range of 4 other peers. This is more representative of a metropolitan neighborhood in a city such as Los Angeles. The link bandwidth between two peers is 4 Mbps, B_{Link}=4 Mbps. With all topologies, we assume the presence of a base station. It is located at one end of the String topology and one corner of the Grid topology.

In each iteration of a simulation run, peers are picked in a round-robin manner to reference a clip for display. The referenced clip is selected as per the Zipfian distribution. If the referenced clip resides in local storage of the peer then the number of simultaneous displays is incremented by one. Otherwise, it invokes

N	25	49	81	100	144	196	324
C	100	196	324	400	576	784	1296
$Optimal$	0.893	0.811	0.763	0.743	0.715	0.693	0.662
$Byte-hit$	0.891	0.811	0.763	0.743	0.715	0.693	0.662
$Frequency-based$	0.868	0.786	0.665	0.621	0.552	0.501	0.428

Table 3: $\sum f_i$ for the different policies for a constant C/N ratio of 4, $S_N = 26$ Gigabytes.

a centralized admission control component that employs the Ford-Fulkerson algorithm [9]. This component tries to reserve one or more paths from the target peer to the base station with bandwidth equivalent to that required for continuous display of the clip ($B_{Display}$). If such a path exists then the request is admitted into the system and the number of simultaneous displays is increased by one. Otherwise, the request is rejected. Since the request is either admitted or rejected instantaneously, the tolerable startup latency in this case is 0. This would be different in a real system. An iteration ends once all N peers have invoked this procedure. Note that the maximum number of simultaneous displays for an iteration is N.

A key metric is the average number of simultaneous displays. It is an average of 100,000 iterations. Another metric is the hit ratio observed by each peer. It is an average across 100,000 requests over N peers. We also study how robust this metric is to errors in the clip access frequencies (see Section 4.3).

We observed identical trends for both the String and Grid topologies. This is because the base station is located at one end of the topology. Hence, for the remainder of this paper, we focus on the Grid topology. A future research direction would be to identify alternative placements of the base station(s) and quantify the effects of B_{Link} and $B_{Display}$ on the number of simultaneous displays.

4.2 Number of Simultaneous displays

Figure 1 shows the results observed with a scale-up experiment. It is termed scale-up because the number of clips (C) is a function of the number of peers (N), $C = 4N$. Thus, when we increase the number of peers (N) from 25 to 50, the number of clips (C) increases from 100 to 200. The storage capacity of each peer is fixed at 26 GB ($S_N = 26$ GB). We assume the bandwidth between the peers is fixed at 4 Mbps.

In Figure 1, the dashed line shows the desired number of simultaneous displays. Its values are dictated by the number of peers shown on the x-axis. This is the theoretical upper bound on system throughput.

Figure 1 shows the following two observations as a function of higher values for C and N. First, all strategies diverge from the theoretical upper bound. Second, Byte-hit and Optimal provide the same throughput, outperforming Frequency-based by a wider margin. We explain each in turn.

As we increase C and N, a smaller fraction of repository fits on each peer because the storage capacity of each peer is fixed at 26 GB, S_N=26 GB. This causes a larger fraction of requests to stream clips from the base station. With the bandwidth of a connection between two peers fixed at 4 Mbps ($B_{link} = 4$ Mbps), the admission control admits fewer requests. This causes all policies to diverge by a wider margin from the theoretical upper bound.

To explain the second observation, we show the total frequency of access to the clips assigned to each peer with Optimal, Byte-hit and Frequency-based in Table 3. Each column of this table corresponds to a different N and C values, i.e., ticks on the x-axis of Figure 1. The total frequency with each strategy drops as a function of C. This is because the storage capacity of each peer is fixed and the frequency of access to each clip decreases with higher C values.

Table 3 shows identical access frequencies with Byte-hit and Optimal which is higher than that with Frequency-based. This is because Byte-hit and Optimal assign a different collection of clips to each peer for each value of C and N. With Frequency-based, the same collection of clips is assigned to each peer for value of N greater than 49. With $N = 25$, a subset of these clips is assigned to each peer.

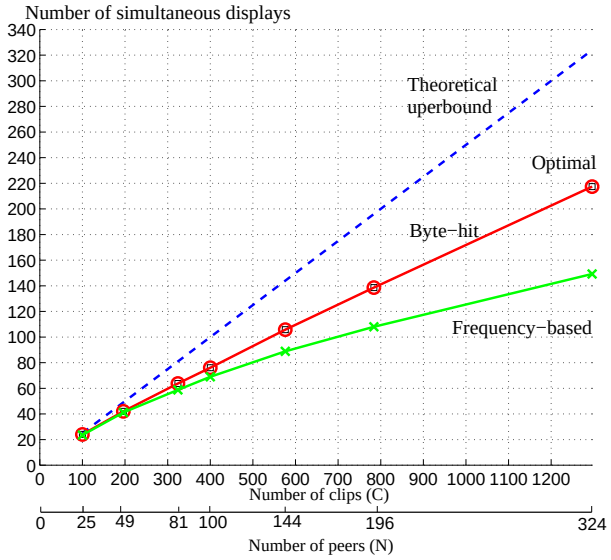


Figure 1: Number of simultaneous displays for the Optimal, Byte-hit, and Frequency-based policies as a function of the database size. Storage capacity of each peer is 26 Gigabytes. Number of peers (N) is 25% of the number of clips (C) and varies from 25 to 324.

To elaborate on these, observe that the frequency of access to each clip changes as a function of C. This changes the value of $\frac{f_i}{S_{C,i}}$, causing Byte-hit to assign a different collection of clips to each peer. With Frequency-based, the sorted order of clips using their access frequency remains unchanged as a function of C. Recall the pattern of clip sizes is: 2 GB, 9 MB, 1 GB, 4.5 MB, 0.5 GB, and 2.25 MB. Total size of each pattern is S_P=3515.75 MB. Frequency-based assigns 7 of these patterns to a peer. This leaves 1389.75 MB of free storage. Thus, the first video clip of the eighth pattern (clip 43) cannot be assigned because it is 2 GB in size. Frequency-based is able to assign only clips 44, 45, 46, and 47 of the eighth pattern. This leaves 374 MB of free space. Only audio clips can be assigned to this free space. When N is 25 and C is 100, a total of 79 clips are assigned to each peer, assigning all the audio clips and leaving 200.75 MB of free space. When N is 49 and C is 196, there are additional audio clips because the repository size is larger, allowing Frequency-based to exhaust all free storage of each peer. In this case, it assigns 117 clips to each peer. The identity of last assigned clip is 192. With values of N higher than 49, the value of C exceeds 192. This means the same collection of clips are assigned to each peer because their sorted order does not change as a function of C.

4.2.1 Summary of Other Experiments

We conducted other experiments that include a comparison of Optimal, Byte-hit and Frequency based with two other policies: First, a policy that assigns clips to a peer randomly. Second, a policy that sorts clips based on $\sqrt{f_i}/S_{C,i}$ and assigns them to each peer. Here is a summary of our observations. First, as expected, the Optimal algorithm provides the best performance, however, its execution time is significantly longer. Second, Byte-hit approximates Optimal in almost all experiments. Third, as the ratio of S_N/S_{DB} increases, even the random scheme (which ignores the clip's popularity) shows a competitive performance. This is because at large values of S_N, most of the C clips are stored in the local storage of each peer. Fourth, with a constant S_N/S_{DB} ratio, as the number peers increases, the Optimal and Byte-Hit policies perform approximately 15-30% better than the random scheme. For smaller values of N ($N = 25$), even for different

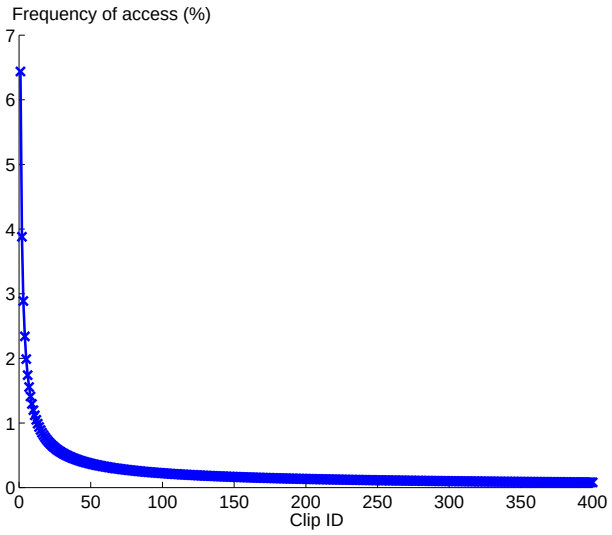


Figure 2: Percentage frequency of access per clip for a repository of $C = 400$ clips using a Zipfian distribution with a mean of 0.27.

Policy	One Demo-graphic	Layout #1	Layout #2	Layout #3	Layout #4
Byte-hit	76.9	75.67 (-1.59%)	77.99 (1.3%)	78.99 (2.71%)	79 (2.73%)
Frequency -based	68.9	69.29 (0.5%)	74.48 (8.09%)	75.71 (9.88%)	75.71 (9.88%)

Table 4: Number of simultaneous displays, numbers in parenthesis are percentage improvements relative to one demographic.

values of S_N/S_{DB}, all schemes except random exhibit similar performance. Fifth, Byte-hit and $\sqrt{f_i}/S_{C,i}$ provided identical throughput in all our experiments.

4.3 Robustness to Frequency of Access

We analyzed the sensitivity of the alternative policies by placing data using a Zipfian distribution of access. Next, the workload generator was modified to generate a clip request using a *shifted*-Zipfian distribution. The latter is generated by shifting the original Zipfian distribution with parameter g. The value of g ranges from 1 to C. This means the frequency of access for clip i is assigned to clip $(i + g)\ mod\ C$. Figure 2 shows the frequency of access for clips for a repository consisting of $C = 400$ clips. When $g = 1$, the workload generator issues requests relative to the original distribution (g=0) in the following manner: Clip 2 is referenced as frequently as Clip 1, Clip 3 is referenced as frequently as Clip 2, so on until Clip 400 is referenced as frequently as Clip 399, and Clip 1 is referenced with the same access frequency as Clip 400. For a given shift value g, we conducted 100,000 iterations of each experiment using a Grid topology of 100 peers. Note that the original placement of clips remains unchanged in each iteration. Figure 3 shows the experimental setup.

Figures 4.a, 4.b, and 4.c show the minimum, average, and maximum number of simultaneous displays supported by a grid topology consisting of 100 peers. As we increase the value of g, both techniques support

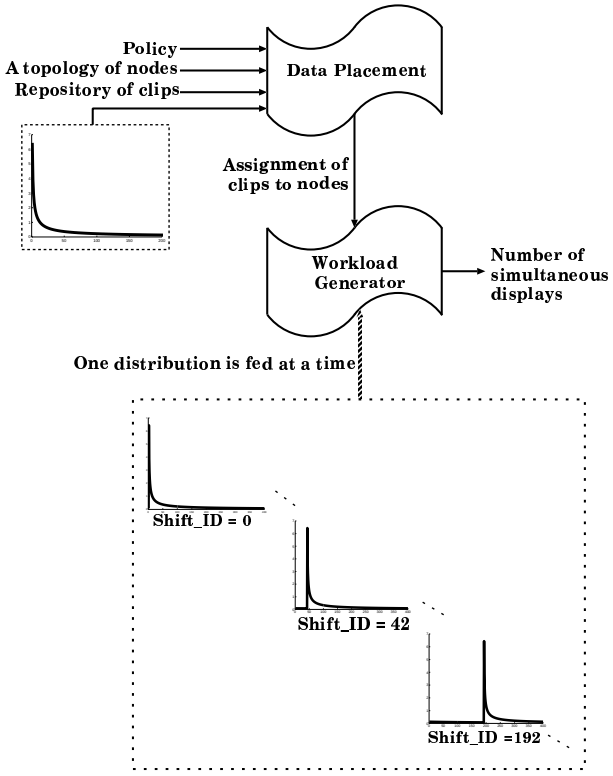
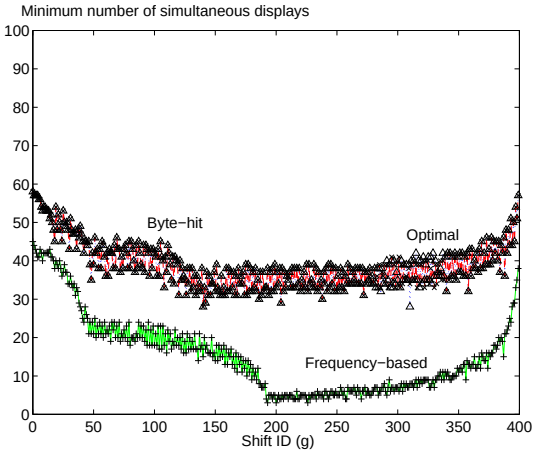


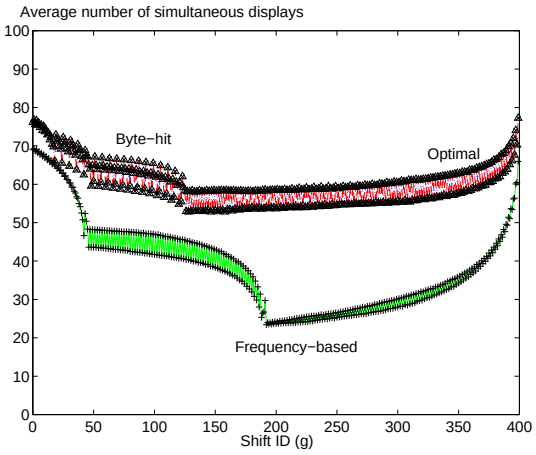
Figure 3: Experimental setup for robustness to error in frequency of access.

Policy	One Demo-graphic	Layout #1	Layout #2	Layout #3	Layout #4
Byte-hit	0	13.57	15.9	20.2	16.92
Frequency -based	0	33.34	37.8	46.15	39.03

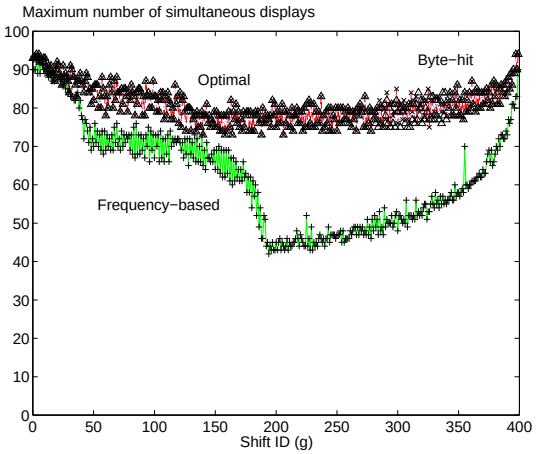
Table 5: Number of peers (out of 100) referencing data from their neighbors. Numbers with one demographic are zero because the clips assigned to each node are identical.



4.a) Minimum



4.b) Average



4.c) Maximum

Figure 4: Minimum, average, and maximum number of simultaneous displays supported by a grid topology consisting of 100 peers.

fewer simultaneous displays because they must reference the remote base station more frequently. Obtained results show Byte-hit is more robust than Frequency-based. This is because Byte-hit assigns 232 clips to each peer. Frequency-based assigns only 117 clips to each peer. Many of the 232 clips are the smaller audio clips with a high byte-hit value. When the frequency of access to these clips increases, Byte-hit services these requests using the local storage of a peer.

System throughput drops in a step manner as a function of g. Below, we elaborate on this trend with the Frequency-based policy. The explanation for Byte-hit is similar. With Frequency-based, see Figure 4.b, the average number of simultaneous displays drops sharply as g approaches 42. This drop levels off when g is 43. There is another sharp drop as g approaches 192. The explanation for this is as follows. Recall from Section 4.2 that Frequency-based assigns the first 42 clips to each peer. As g approaches 42, the frequency of access to these clips decreases, reducing the number of requests that find their referenced clips locally. They are directed to the base station and the admission control rejects many due to the limited bandwidth between peers. This explains the first sharp drop as shift ID approaches 42.

When we increase g (shift ID) from 42 to 100, the drop becomes gradual. This is due to assignment of audio clips to local storage of the peers. Recall from Section 4.2 that the id of the last assigned clip is 192. As g approaches 192, throughput degrades faster as fewer references are serviced using local storage of a peer. The lowest point is when g is 193. Beyond this point, the throughput increases because the number of references to the first 42 clips starts to increase once again.

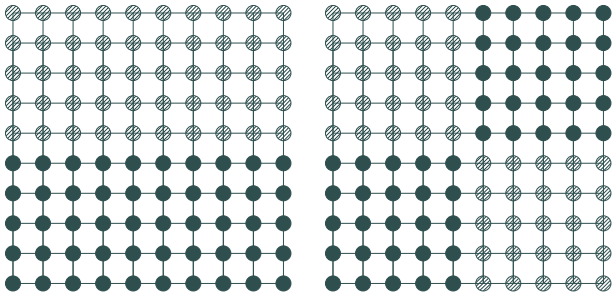
Similar trends are observed with Byte-hit where it assigns 232 clips to each peer. The trend is not as dramatic because it assigns many small audio clips to each peer and the id of the last assigned clip is 400.

4.4 Alternative Demographic Layouts

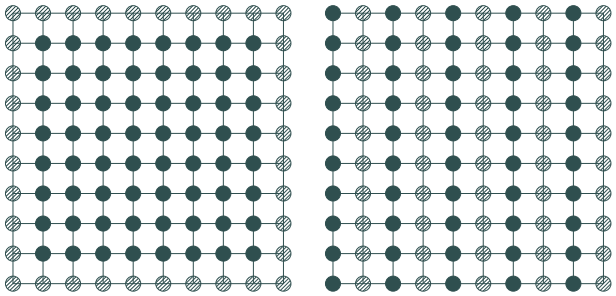
Finally, we compared the impact of different demographics on the performance of alternative policies. Multiple demographics force Simple to store different clips on each peer. This enables a peer to act as a server for a pending request. Once again, we assumed a 100 peer configuration with a repository of 400 clips as described in Section 4.1. We assumed two demographics with access patterns that mirror one another. The first demographics, termed dgr1, assumes the access frequency of Figure 2. The second, termed dgr2, is a mirror of the first because it assumes clip 400 is referenced with the same frequency as clip 1 of dgr1, clip 399 is accessed as frequently as clip 2 of dgr1, and so on. A peer may belong to either dgr1 or dgr2. We analyzed four different layouts for peers as shown in Figure 5. When the workload generator issues a request on behalf of a peer, it uses its demographic to determine the identity of the referenced clip.

Table 4 shows the total number of simultaneous displays supported by Byte-hit and Frequency-based with each layout. We have included results from Figure 1 pertaining to one demographic with 100 peers. Numbers in parentheses show the percentage improvement with different layouts relative to the one demographic scenario. These show Frequency-based observes higher improvement. This is because a larger number of peers stream clips from their neighbors, see Table 4. With Frequency-based and 2 demographics, each peer holds 117 clips and the entire network holds 234 clips. (A peer belonging to dgr1 holds 117 clips different from those assigned to a peer belonging to dgr2.) This is twice that with one demographic. With Byte-hit, the number of unique clips in the ad-hoc network increases modestly (by 31, from 232 to 263). Thus, its observed improvement relative to one demographic is insignificant.

Note that Frequency-based does not outperform Byte-hit (compare the two rows of Table 4) because its number of unique clips (234) in the ad-hoc network remains less than Byte-hit (263).



5.a) Layout #1 5.b)Layout #2



5.c) Layout #3 5.d) Layout #4

Figure 5: Four alternative layouts of peers belonging to two different demographics denoted with solid and shaded circles.

5 Conclusions and Future Research Directions

The primary contribution of this paper is to show the superiority of Byte-hit policy when assigning clips to peers of a mesh network. When compared with the Frequency-based policy, Byte-hit provides a higher throughput and is more robust to the error in access frequencies. Byte-hit performs almost identical to the Optimal policy in all our experiments. (Discussions of Table 2 show Byte-hit is not always Optimal.) This means it is difficult, if not impossible, to design a policy that would outperform Byte-hit significantly.

A decentralized implementation of Simple using Byte-hit might be as follows. Periodically, either a remote server or one of the peers publishes v clips. Each clip has an anticipated frequency of access for different demographics. This meta-data is broadcasted to all peers in the network. A peer uses the demographics of its household to lookup the frequency of access for each of the ρ clips occupying its available storage. It computes the Byte-hit value of these clips and the newly published v clips. Both lists are combined and sorted based on their Byte-hit ratio. Next, it assigns clips from this list to its available storage. We compare this assignment with the existing ρ clips to produce two lists. List one contains those clips to be evicted. List two contains κ clips out of v to be stored in the local storage. Next, it stores the identity of these κ new clips in a message and transmits this message to the publishing peer. The publishing peer may employ either unicast or multicast to disseminate those new clips selected for storage by one or more peers.

A future research direction is to explore other placement strategies that utilize storage of a peer as a shared resource and compute the number of replicas for each clip [21, 7, 1]. Some preliminary results are reported in [15]. One may consider more sophisticated variations where a clip is partitioned into fragments and different peers store different fragments. The first few fragments of a clip might be replicated more aggressively because they are needed more urgently [17]. When displaying a clip, a peer will display its first

few fragments from its local storage while streaming its remaining fragments from the neighboring peers.

6 Acknowledgments

This research was made possible by an unrestricted cash gift from Microsoft research, a fellowship award from the Annenberg Center for Communication, and NSF research grant IIS-0307908.

References

[1] A. Aazami, S. Ghandeharizadeh, and T. Helmi. An Optimal Continuous Media Replication Strategy for Ad-hoc Networks of Wireless Devices. In *Tenth International Workshop on Multimedia Information Systems (MIS)*, August 2004.

[2] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou. A multi-radio unification protocol for ieee 802.11 wireless networks. In *BROADNETS '04: Proceedings of the First IEEE International Conference on Broadband Networks*, pages 344–354, Washington, DC, USA, 2004. IEEE Computer Society.

[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] V. Bush. As We May Think. *The Atlantic Monthly*, 176(1):101–108, July 1945.

[5] W. Cheng, L. Golubchik, and D. Kay. Total Recall: Are Privacy Changes Inevitable? a position paper. In *Proceedings of the First ACM Workshop on Continuous Archival and Retrieval of Personal Experiences*, New York, NY, October 2004.

[6] S. M. Cherry. Across the great divide. *Spectrum IEEE*, 41(1):36–39, January 2004.

[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, and R. Rivest, editors. *Introduction to Algorithms*, chapter 17.2. MIT Press, 1989.

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

[10] A. Dan, D. Dias, R. Mukherjee, D. Sitaram, and R. Tewari. Buffering and Caching in Large-Scale Video Servers. In *Proc. of COMPCON*, 1995.

[11] R. Draves, J. Padhye, and B. Zill. Comparison of Routing Metrics for Static Multi-Hop Wireless Networks. In *ACM SIGCOMM*, August 2004.

[12] J. Gemmell, B. Gordon, R. Lueder, S. Drucker, and C. Wong. MyLifeBits: Fullfilling the Memex Vision. In *ACM Multimedia*, December 2002.

[13] J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe. Multimedia Storage Servers: A Tutorial. *IEEE Computer*, 28(5):40–49, 1995.

[14] S. Ghandeharizadeh. H2O Clouds: Issues, Challenges and Solutions. In *Fourth IEEE Pacific-Rim Conference on Multimedia*, December 2003.

[15] S. Ghandeharizadeh, T. Helmi, and S. Shayandeh. To Share or Not To Share Storage in Mesh Networks: A System Throughput Perspective. In *Submitted for publication*, 2006.

[16] S. Ghandeharizadeh, S. Kim, and C. Shahabi. On Configuring a Single Disk Continuous Media Server. In *Proceedings of the ACM SIGMETRICS/Performance*, May 1995.

[17] S. Ghandeharizadeh, B. Krishnamachari, and S. Song. Placement of Continuous Media in Wireless Peer-to-Peer Networks. *IEEE Transactions on Multimedia*, April 2004.

[18] S. Ghandeharizadeh and R. Muntz. Design and Implementation of Scalable Continuous Media Servers. *Parallel Computing*, 24(1):91–122, May 1998.

[19] IEEE. Ieee 802.11.

[20] K. Jain, J. Padhye, V. Padmanabhan, and L. Qui. Impact of Interference on Multi-hop Wireless Network Performance. In *ACM Mobicom*, Sept 2003.

[21] S. Jin and L. Wang. Conten and Service Replication Strategies in Multi-hop Wireless Mesh Networks. In *Proceedings of the IEEE/ACM MSWiM*, October 2005.

A Appendix 1

The complexity of Simple using frequency-based policy is $O(C \cdot \log C)$. Below, we prove this policy is optimal for a homogeneous repository consisting of equi-sized clips.

Let S_N be the storage capacity of a peer. For a homogeneous repository consisting of clips with a constant size and display time, $S_{C,i} = S_{C,j}$, $D_i = D_j$, $1 \leq i, j \leq C$, where C is the number of clips in the repository, the optimal placement with Simple is as follows. First, sort clips using their f_i values. Next, assign the first S clips to each node such that $\sum_{i \epsilon S} S_{C,i} \leq S_N$.

Lemma 1 *The assignment produced by Simple is optimal.*

Proof: *Recall that the objective is to maximize the local hit ratio, $\sum_{i \epsilon S} f_i$, for each H2O device. Let S_{opt} represent the optimal set of clips present on each H20 device. Then, S'_{opt} represents the complement of S_{opt}.*

(i) The Optimal solution contains no idle storage i.e. if $S_N - \sum_{i \epsilon S_{opt}} S_{C,i} \geq S_{C,j}$, $j \epsilon S'_{opt}$, then adding the clip j from S'_{opt}, such that $f_j = \max_{k \epsilon S'_{opt}} f_k$, to the H20 device will increase $\sum_{i \epsilon S_{opt}} f_i$, yielding a contradiction.

(ii) We want to show that the Optimal algorithm has no inversions i.e. clip pairs (i, j), $i \epsilon S_{opt}$, $j \epsilon S'_{opt}$, $f_i < f_j$. This is because we can swap clips i and j and the resultant hit ratio $\sum_{k \epsilon \{S_{opt} - i\}} f_k - f_i + f_j$ has increased. In this way, we iteratively eliminate all inversions. Note that, since the clips are sorted by their f_i within both S_{opt} and S'_{opt}, we maintain a pointer to the head of each of these sets. We move the pointer at the head of S_{opt} ahead till we find a clip i for which there exists a clip j at the head S'_{opt} such that $f_i < f_j$. Then clips i and j are exchanged. Repeating this process iteratively, the Optimal algorithm transforms to the solution produced by our greedy algorithm. □