Headlight Prefetching for Mobile Media Streaming

Shiow-yang Wu
Dept of Computer Science and Information Engineering
National Dong Hwa Univ
Hualien, Taiwan, ROC
showyang@mail.ndhu.edu.tw

Jungchu Hsu
E-TEN Information Systems Co., Ltd
No.256, Yangguang Street, Neihu Chiu
Taipei, Taiwan, ROC
jungchu.hsu@etencorp.com

Chieh-Ming Chen
Microelectronics Technology Inc.
1, Innovation Road II, Hsinchu Science-based Industrial Park
Hsinchu, Taiwan, ROC
chen_chieh-ming@mti.com.tw

ABSTRACT

Multimedia information services in mobile environments are becoming more and more important with the proliferation of 3G technologies. Media streaming, in particular, is a promising technology for providing services such as news clips, live sports, and hot movies on the fly. To avoid service interruption when the users keep moving, proper data management strategies must be employed. We propose a new *headlight prefetching* technique for the streaming access points to deal with the uncertainty of client movement and the requirement of seamless service hand-off. For each mobile client, we maintain a virtual fan-shaped prefetching zone along the direction of movement similar to the headlight of a moving vehicle. The overlapping area and the accumulated virtual illuminance of the headlight zone on a particular cell determines the degree and volume of prefetching to be made by the streaming access point of that cell. Headlight prefetching solves the issues of identifying the streaming access points responsible for prefetching, the timing and the amount of data to prefetch in a single mechanism which is simple and effective. Simulation results demonstrate that our techniques can significantly decrease streaming disruptions, reduce bandwidth consumption, increase cache utilization and improve service response time.

Categories and Subject Descriptors

H.3.4 [**Systems and Software**]: Information networks; H.3.5 [**Online Information Services**]: Data sharing; C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Algorithms, Design, Performance

Keywords

Media streaming, prefetching, mobile data management

1. INTRODUCTION

Media streaming is a promising technology for providing multimedia information services such as news clips, live sports, and hot movies in mobile environments. Effective data management for media streaming is naturally the key to the successfulness of such services. Since many users may request the same media (hot media), traditional client-server model can easily result in server bottleneck, bandwidth waste, poor cache utilization and longer delay. Furthermore, the mobility of mobile users raise the issue of seamless service hand-off. To avoid service interruption, proper data management strategies must be employed. In this paper, we propose a new technique named *headlight prefetching* for media streaming in mobile environments. The technique is designed for the streaming access points to deal with the uncertainty of client movement, the unpredictability of request pattern and the requirement of seamless service hand-off. For each mobile client, we maintain a virtual fan-shaped prefetching zone along the direction of movement similar to the headlight of a moving vehicle. The overlapping area and the accumulated virtual illuminance of the headlight zone on a particular cell determines the degree and volume of prefetching to be made by the streaming access point of that cell. When a mobile user makes an unexpected sharp turn, the *headlight shifting* technique is used such that all the previously prefetched media segments can be easily shifted to accommodate the new direction of movement. For users requesting the same media at around the same time, the *headlight sharing* technique is developed for sharing the headlight zones to avoid repeated prefetching. The set of techniques solve the issues of identifying the streaming access points responsible for prefetching, the timing and the amount of data to prefetch in a single mechanism which is simple, intuitively appealing and effective.

To evaluate the effectiveness of our techniques, we construct a Java-based simulation environment and compare the performance of different combinations of our schemes with on-demand techniques. Simulation results demonstrate that, for streaming media services in mobile environments, headlight prefetching, shifting and sharing are simple and effective techniques which can significantly decrease streaming disruptions, reduce bandwidth consumption, increase cache utilization and improve service response time.

The rest of the paper is organized as follows. Section 2 provides a survey of related issues and research work. Section 3 presents the media streaming system infrastructure we assume and characterize the challenges of dynamic data management in such environments. Section 4 introduces the idea of headlight prefetching and the associated data management techniques. In Section 5, we outline the simulation environment for experimenting with our ideas and present the results of performance evaluation on different aspects of the prefetching techniques. Section 6 concludes the paper.

2. RELATED WORK

With the proliferation of 3G technologies, the integration of cel-

MobiDE'07, June 10, 2007, Beijing, China.

lular and broadcast networks, and the rapid advances of the capabilities of handheld devices, mobile multimedia services such as live news, sports events, hot movies or even TV broadcast to mobile phones and PDAs are becoming more and more feasible and attractive to an estimated over 2 billion customers worldwide. New services and standards such as MBMS [2], DVB family(DVB-H in particular) [3, 13], BCMCS [13] and MediaFLO [5], etc., are all striving to satisfy the ever stronger demand of mobile users on up to date multimedia contents and entertainment. Seamless media streaming in mobile environments plays the key role in such services since most of them are provided in the form of streaming media.

Traditional media streaming are mostly focused on wired networks such as Internet [7, 8, 11, 14]. Media streaming in mobile environments has been attracting much attention lately. Fitzek and Reisslein propose a real-time continuous media streaming protocol with special emphasis on dynamic transmission capacity allocation and prefetching [4]. Li and Wang develop NonStop [10] which is a set of middleware-based run-time algorithms with partition prediction and service replication for continuous media streaming in mobile and ad hoc networks. Anastasi et al. look further into the problem of energy efficiency when providing streaming media services to mobile clients [1]. Xue et al. explore group mobility to predict the future availability of wireless links for increasing total streaming capacity with P2P streaming [15]. Qin et al. propose an iterative algorithm to predict continuous link availability between two mobile peers [12]. The V3 architecture proposed by Guo et al. for live video streaming is essentially a P2P streaming architecture for moving vehicles [6]. Kyriakidou et al. discuss streaming architecture, content distribution and rate adaptation algorithms for video streaming to fast moving users in 3G networks [9]. Zhai et al. propose buffer management methods that employ statistical analysis to predict the trend of user movement among cells [16].

The problem we face is challenging in that we must deal with the uncertainty of client movement, the unpredictability of request pattern and the requirement of seamless service hand-off with a simple and unified mechanism. The set of headlight prefetching techniques we proposed can effectively cope with the dynamics of mobile environments to provide effective streaming media services.

3. STREAMING MEDIA SERVICE ARCHITECTURE

For media streaming services in mobile environments, we assume a system architecture as depicted in Figure 1. The multimedia information are provided by *streaming media servers*(SMSs). All cells have corresponding *streaming access points*(SAPs) connected with each other through traditional fixed link networks. Each SAP provides wireless media services for *mobile users* within that cell. Users not reachable from any SAP are *disconnected*. In general, let the local wireless access cost between a mobile user and an SAP be L, and the remote access cost of requesting the unit from an SMS be R. Then, based on the nature of the architecture and service charge, we assume that L is larger than R since wireless access cost is usually larger than that of fixed link. However, if a media segment is available locally on an SAP, then it only takes a local access cost of L to service the segment. If the segment is not available on the SAP cache, then a remote access cost of R must be added in addition to the local access cost. Therefore, a remote access is always more expensive than a local access. Since intermittent disconnection is unavoidable in mobile environments, effective data management strategies must be employed to conserve access cost and reduce play interruption.

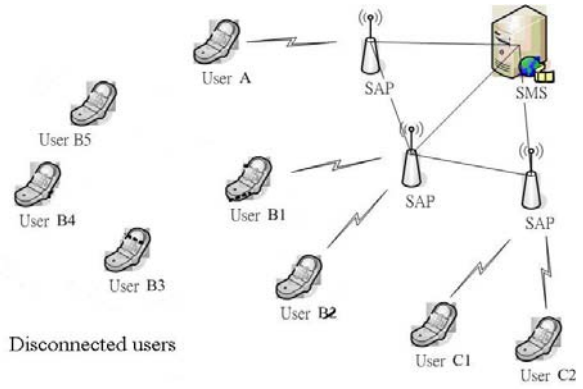

Figure 1: System architecture for mobile streaming service.

A streaming media is usually considered as a sequence of media segments. Because of the streaming nature, it is not necessary to provide the entire media all at once. A media request from a mobile client can therefore be treated as the starting request of a sequence of media segments. There are several data management challenges for streaming media services in mobile environments:

- A mobile user can request any media from any location at any time. On a request, the SAP of the cell where the user resides must locate and send the first segment as soon as possible to reduce service delay.
- Once started, the SAP must fetch and transmit the subsequent segments fast enough to catch up with the playing speed. Good prefetching and buffering techniques are required to avoid possible interruption.
- A mobile user can move and change direction at any time. Such a dynamism can only be handled by close coordination of neighboring SAPs to provide seamless streaming media services across cell boundaries.
- To reduce cost, the media segments should be served on a proximity basis. In other words, it is best to use the local SAP or to locate a nearby SAP with the requested segments to provide the service. The remote access to the SMS should only be used as a last resort.

Our goal is to develop effective dynamic data management techniques to answer the challenges of streaming media services in mobile environments. In particular, the headlight prefetching and associated techniques are designed to solve the problems of identifying responsible SAPs, determining prefetching segment assignment and handling the dynamic data management issues in a simple and unified framework. Our schemes are unique in several respects:

- The idea of virtual fan-shaped headlight prefetching zone is simple, intuitive appealing and efficient. We can use the headlight coverage area to identify the prefetching SAPs and the virtual illuminance to determine the lookahead window for each SAP. Both are straightforward and easy to compute.
- Headlight prefetching is flexible and dynamic adjustable. We can use the shape and size of the virtual fan to control the range and degree of prefetching. For mostly straight and fast moving users (eg. vehicles on a freeway, passengers on a train, ...), we can have a smaller central angle and longer radius. For slow moving clients that tend to wonder around, a

larger central angle and smaller radius allow more neighboring SAPs to be prepared for serving the clients.

- Should a mobile user make an unexpected sharp turn, we provide the *headlight shifting* technique such that all the previously prefetched segments can be easily shifted to the SAPs along the new direction.
- To efficiently reuse prefetched segments, the *headlight sharing* technique is developed to coordinate neighboring SAPs on media services and to avoid repeated segment prefetching.

As a summary, streaming media services in mobile environments entail significant challenges on highly dynamic and efficient data management. The mobile and streaming characteristics of the services also provide a window of opportunity for information system designers. We take on this opportunity to provide simple and effective dynamic data management techniques that adhere closely to the movement patterns of mobile users. It turns out that high quality streaming media services in mobile environments can indeed be achieved with proper coordinations of SAPs and right strategies for data management.

4. HEADLIGHT PREFETCHING

As stated earlier, an important characteristic of streaming media is the continuous playing requirement. Once a media is started, the requesting user expects a smooth and seamless viewing experience. Fetching a segment upon its request is not likely to catch up with the playing speed. Prefetching and buffering are almost a must in such case. Traditional prefetching is simply done by maintaining a sliding window immediately ahead of the current segment by the SAP in charge of the media service. This is only satisfactory when the user is moving in a straight ahead pattern. For irregular moving patterns, the traditional approach may fail miserably. The problem is that if the user changes direction and moves toward a new SAP, the later may not have anything prepared for the user. The uncertainty of user movement can easily result in frequent disruption and unpleasant viewing experience. To have all surrounding SAPs prefetch the needed segments for the user is clearly not cost effective. We therefore need a good mechanism to continuously identify the proper set of prefetching SAPs for a moving client. Those SAPs that are more likely to be visited next should be selected with higher probability. Even if the set of prefetching SAPs is successfully identified, we still have another important problem of determining the right media segments for each SAP to prefetch. The simplest approach of having all selected SAPs prefetch the same set of media segments is certainly not satisfactory. To save processing and communication costs, the ideal case is to prefetch just the segments needed to keep the media viewing smooth. Similar to the previous issue, we need to cope with the uncertainty of user's moving speed and pattern by dynamically determining the prefetching segments assignment. Those SAPs that are more likely to be visited next should be assigned more segments to prefetch. In addition to the prefetching SAPs identification and segment assignment problems, we also need to determine the right timing for the SAPs to start or stop prefetching. Otherwise the prefetched segments may either arrive too early or too late.

The idea of headlight prefetching is to have a simple and unified mechanism for solving the issues discussed above. A *headlight prefetching zone* as depicted in Figure 2 is a virtual fan-shaped area along the direction of user movement similar to the headlight of a moving vehicle. All SAPs of the cells touched by the headlight zone are selected as the prefetching SAPs for the user. The overlapping area and the accumulated virtual illuminance of the headlight zone on a particular cell determines the degree and volume of prefetching to be made by the SAP of the cell. Headlight prefetching solves the issues of prefetching SAPs identification, segment assignment and the prefetch timing in a single mechanism which is simple and intuitive appealing.

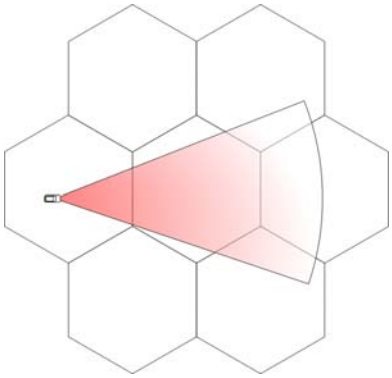

Figure 2: The headlight prefetching zone.

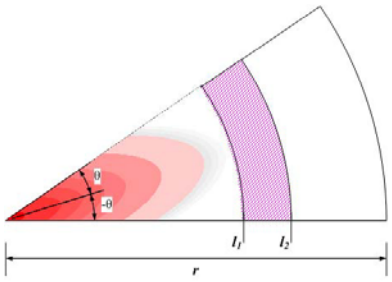

Figure 3: The headlight model.

The headlight prefetching zone is modeled by two parameters as illustrated in Figure 3. The radius r determines the extent of look ahead for prefetching. The angle θ is used to control the span of coverage. Both are dynamically adjustable. In general, faster moving users need longer radiuses. Users that tend to wonder around need larger θs to have more SAPs ready to carry on the services when a user changes direction unexpectedly. The headlight zone serves as a prediction of possible future interaction of the user with neighboring cells. By using such a simple and intuitive appealing metaphor, we can easily identify the set of SAPs that need to prefetch media segments for the user.

Once the prefetching SAPs identification problem is solved, we still need to take on the segment assignment problem. The headlight metaphor gives us yet another simple way to handle this problem. A basic characteristic of a vehicle headlight is that the farther away from the vehicle the lower the illuminance. The area immediately in front of the vehicle has the highest brightness. This characteristic matches exactly the requirement of the segment assignment problem. Since a user can change direction at any time, the media segments prefetched by the SAPs farther away from the user are less likely to be actually used. They should be assigned fewer segments to save the cost. On the other hand, the SAPs closer to the user are more likely to be responsible for providing the media services. They should be allocated more segments to prevent undesirable disruption. Due to such a close resemblance, we use the accumulated virtual illuminance of the cell area covered by the headlight zone as the weight for segment assignment. In this way, we solve both problems in a unified framework.

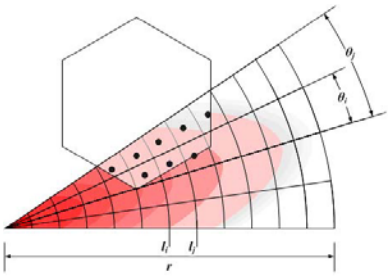

Figure 4: The headlight zone and the computation of virtual illuminance.

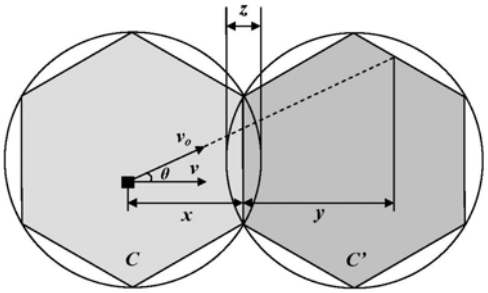

Figure 5: The movement of a user in a cell.

The computation of the virtual illuminance, however, is not that straightforward. It is quite easy to compute the illuminance of a complete slice of the headlight as indicated by the following formula where k is a tunable constant.

$$I = \int_{-\theta}^{\theta}\int_{l_1}^{l_2} \alpha \times r \; dr d\theta \quad (1)$$

$$\alpha = \frac{k|\cos\theta|}{r^2} \quad (2)$$

The problem is in determining the cell area covered by the headlight zone since the intersected area could be in any shape. To avoid the costly computation of the covered area, we use a much simpler approach to approximate the virtual illuminance as illustrated in Figure 4. More specifically, we partition the headlight zone into smaller grids and precompute the virtual illuminance as well as the center of each grid. Since each grid is in a regular shape, the virtual illuminance can be easily computed by the following formula.

$$I(\theta_i, \theta_j, l_i, l_j) = \int_{\theta_i}^{\theta_j}\int_{l_i}^{l_j} \alpha \times r \; dr d\theta \quad (3)$$

On the need to determine the segment assignment of a particular cell, we simply add up the illuminance of all grids whose centers fall inside the cell. Since the granularity of the grids can be changed easily, we can have higher level of precision at any time by using finer partitioning.

4.1 Prefetching Segment Assignment

To determine the segment assignment, we need to consider both the user movement and media playing speed. Let us assume the parameters listed in Table 1. From the illustration in Figure 5, it is quite easy to see that

$$v = v_o \cos\theta \quad (4)$$

$$t = \frac{x + z/2}{v} \quad (5)$$

$$t' = \frac{y + z/2}{v} \quad (6)$$

Therefore the number of segments that should be handled by the current cell is tP. If the current media segment being played is S_i, then SAP_C must prefetch the segments $S_{i+1}, S_{i+2}, \ldots, S_{i+tP}$. The first segment that need to be prefetched by $SAP_{C'}$ is S_{i+tP+1}. The expected number of segments to be handled is $t'P$. The starting segment and the total number of segments to prefetch for all other SAPs within the headlight zone can be determined in a similar way. Since it is clearly not cost worthy for all such SAPs to prefetch the full range of segments, therefore we use the virtual illuminance as a weighting factor to determine the actual number of segments to prefetch. More specifically, the exact segment assignment for $SAP_{C'}$ to prefetch is

$$S_{i+tP+1}, S_{i+tP+2}, \ldots, S_{i+tP+t'P\lceil\frac{I'}{L}\rceil} \quad (7)$$

	Description
S	The media stream which is a sequence of segments $\{S_1, S_2, \ldots, S_n\}$.
C	The current cell where the user resides.
C'	The next cell the user is approaching.
SAP_C	The SAP of the cell C.
$SAP_{C'}$	The SAP of the cell C'.
v_o	The current velocity of the user.
θ	The angel between v_o and the axis connecting the centers of C and C'.
v	The projected velocity of the user on the axis.
z	The length of the overlapping zone between C and C'.
x	The distance along the axis between the user's current position and the cell boundary.
y	The distance along the axis between the projected entrance and exit points of the user in C'.
P	The media playing speed in number of segments per time unit.
t	The time remained of the user within the current cell C.
t'	The predicted time of stay of the user within the next cell C'.
I'	The virtual illuminance incident on C'.
L	The total virtual illuminance incident on the entire headlight zone.

Table 1: Parameters for determining the segment assignment.

4.2 Headlight Shifting

Headlight prefetching is quite effective for largely stable moving users. However, if the user makes a sharp turn, then most or even all of the prefetching done on the previous headlight zone may be completely useless since the user is no longer heading toward the predicted direction. We can of course start a new round of headlight prefetching for the new situation. However, this will double the prefetching cost. Since the media stream is played continuously, the segments needed for the new headlight zone overlap significantly with that of the old zone. Therefore the better way is to shift these segments to the new zone. We call the idea *headlight shifting*. For efficient headlight shifting, we need to solve at least

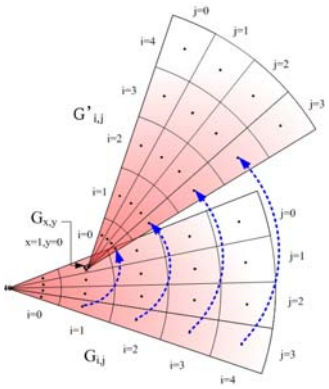

Figure 6: Direct mapping for headlight shifting.

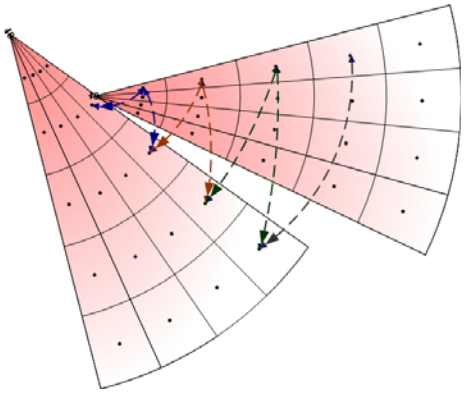

Figure 7: The overlapped mapping strategy.

two problems:

- Since the same media segment may be available on more than one SAPs, how to choose or map the shifting source and target?
- Different SAPs may have different number of prefetched segments available for shifting, how do we distribute and balance the shifting load?

We propose three strategies for headlight shifting. The simplest and most intuitive way is to map the grids of the new zone to corresponding grids of the old zone with possible offset based on user's current position. This is called *direct mapping* and is illustrated in Figure 6. More specifically, let the old and new grids be $G_{i,j}$ and $G'_{i,j}$ respectively. Also assume that the position where the user makes the turn is in grid $G_{x,y}$. Then we simply map $G'_{i,j}$ to $G_{i+x,j}$. In other words, the SAP whose cell covers $G'_{i,j}$ can simply request the media segments to be directly shifted from the SAP whose cell covers $G_{i+x,j}$.

Direct mapping works fine if the user is at a position close to the entrance of a grid since most of the segments are not yet played and therefore the mapping is useful and efficient. However, the problem with direct mapping is that if the user is at a position about to leave a grid then the mapping is likely to be off by one grid since most of the segments have already been played. The problem can be easily solved by the second strategy named *overlapped mapping*. Instead of mapping the grids one-to-one, we extend the mapping into a one-to-many overlapped mapping as illustrated in Figure 7. More specifically, a parameter called *overlapped length(OL)* is defined to denote the degree of overlapping. Now the new grid $G'_{i,j}$ is mapped to the old grids from $G_{i+x,j}$ to $G_{i+x+OL,j}$. The example in Figure 7 has $OL = 2$. The strategy reduces to the direct mapping strategy if $OL = 0$.

The most distinctive benefit of both direct and overlapped mapping strategies is that all shifting sources and targets can be easily computed without any search. However, the resulting mapping may not be optimal in terms of shifting cost since the same segment may be available on more than one SAP. The lowest cost shifting source may not be the one that is mapped directly. We therefore propose the third strategy named the *shortest distance mapping strategy*. After determining the segment assignment for the new zone, the distances between a new SAP in the zone and the SAPs in the old zone with the assigned segments can be computed. We can therefore map the new SAP to the nearest neighbor in the old zone with the required segment. If there are more than one qualified neighboring SAPs available to choose from, we follow the *lowest shifted volume first* strategy to balance the load.

4.3 Headlight Sharing

Headlight shifting only takes the headlight zone of one user into consideration. Segments not yet prefetched by any SAP of the old zone can only be retrieved from the remote source. Since the same media may be viewed by more than one user at the same time, especially for hot medias, the headlight zones of different users may have many segments in common. If they overlap with each others, then it is very likely that we can find the needed segments from other zones with or without shifting. We call this idea *headlight sharing* since segments prefetched for a zone by an SAP are shared with neighboring SAPs with overlapping headlight zones. Once a requested segment is located in the neighborhood, the cost of prefetching from the remote server can also be saved.

To facilitate headlight sharing, a distributed index structure is constructed on each SAP for maintaining the availability of media segments on other SAPs. For ease of presentation, we assume that the set of all medias be $\mathcal{M}$ and the set of all segments of media i be G_i. A separate *index table* $T_i = \{S_i, M\}$ is maintained for each neighboring SAP from which an index message is received. S_i is the id of the SAP. $M = \{(m_j, s_k, t) | m_j \in \mathcal{M}, s_k \in G_{m_j}\}$ is the partial set of media and segments available on that SAP. Each segment has an expiration time t to indicate the period of validity. An *SAP index* keeps track of all the available index tables while a *segment index* maintains, for each segment, a list of SAPs from which the segment can be found. The information is obtained from the index exchange between SAPs. To avoid additional communication overhead, the index messages are piggybacked with the headlight prefetching messages. With the index ready, the segments prefetched by an SAP can be easily shared with other SAPs to facilitate headlight sharing. A sample index is illustrated in Figure 8.

The problem now comes to the efficient maintenance of the distributed index. To keep a complete index of all medias and segments on each SAP is clearly cost prohibitive. Therefore only selected information is exchanged based on the following strategies:

- *Random*: Randomly select part of the available segments to exchange index.
- *Local popularity*: Select those segments that are locally popular based on the rationale that they may be popular on other SAPs as well.
- *SAP-specific popularity*: Segments that are popular among the users coming from the same SAP are selected to send

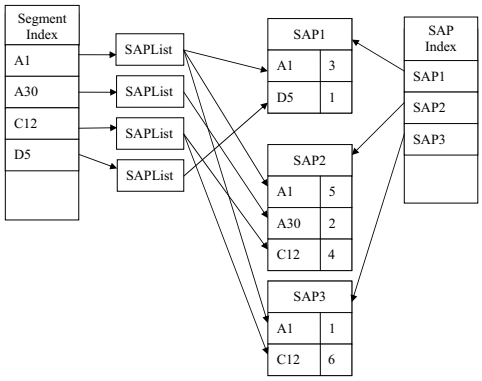

Figure 8: Distributed index for headlight sharing.

the index to that particular SAP. This is to satisfy different request patterns of users from different cells.

- *SAP-specific rareness*: Segments that are locally popular but rarely requested by the users coming from a particular SAP are selected to send the index to that SAP. This is because the segments available locally are to satisfy either the local requests or headlight prefetching. The availability of the prefetched segments are already known to the SAPs that send the requests. Therefore we only need to send the index of those segments that are not known to other SAPs.

Headlight sharing works closely with headlight prefetching and shifting. On receiving a segment request, be it from a local user or a prefetch request, an SAP looks up the index after a cache miss to see if the segment is available on other SAPs. The segment can then be retrieved from a nearby SAP rather than from the remote source. During the headlight shifting, those segments that are in the new assignment but no SAP to shift from can very likely be satisfied using the headlight sharing index. Since the index messages are piggybacked with the prefetching messages and the index search cost is quite low in comparison with segment transmission cost, the overhead of headlight sharing is almost negligible. Later in Section 5, we will show that different combinations of these techniques result in significant performance improvement over on-demand and simple look ahead prefetching in terms of response time, interruption rate and completion rate.

5. SIMULATION AND EVALUATION

To evaluate the performance of our techniques, we have developed a Java-based simulation environment. The set of simulation parameters and their value ranges are listed in Table 2.

The first set of experiments is to compare the performance of different combinations of headlight prefetching, shifting and sharing strategies. All strategies are tested under two request patterns: random and Zipf distribution. We are particular interested in the total number of interruptions during a media playback and the average download time of a media segment since they are the dominant factors affecting service quality experienced by the users. Figure 9 shows that all strategies perform significantly better than traditional on-demand strategy with simple lookahead prefetching. Since the inclusion of the later makes others hard to compare, we exclude the strategy for all subsequent figures. From Figure 10 and Figure 11 we can see that shifting and sharing indeed improve the performance of headlight prefetching, especially when used together. All strategies achieve better performance under Zipf distribution since hot medias can be quickly shared with other SAPs.

Simulation Parameters	Value Range
Number of round per simulation	2500
Number of base stations	100
Cell size	$1 \sim 5$ km
Base station processing capacity	30 (segments)
Size of base station cache	20% of medias
Number of users	2000
Speed of users	$10 \sim 50$ (meter/s)
Turning angles	$0^\circ \sim 90^\circ$
Size of client cache	2% of average media size
Client side buffering	2%
Playback rate	0.5 (segment/s)
Number of medias	$10 \sim 50$
Media length	$400 \sim 600$ (segments)
Request pattern	random or Zipf distribution ($\theta = 1.5$)
Remote access cost	3
Local access cost	2
Trans cost between SAPs	1

Table 2: Simulation parameters and settings.

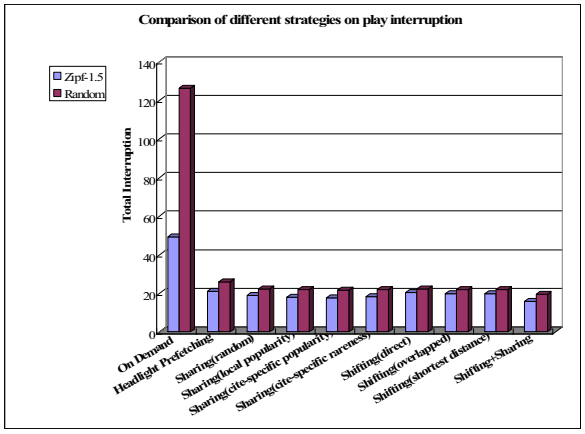

Figure 9: Comparison of all strategies on play interruption.

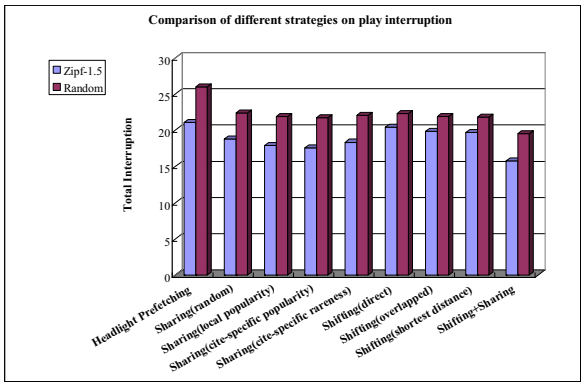

Figure 10: Comparison of different headlight prefetching strategies on play interruption.

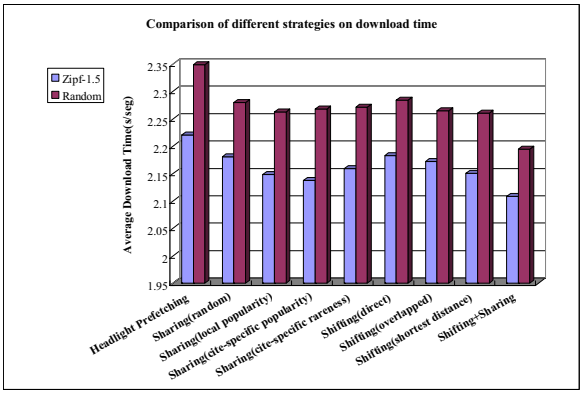

Figure 11: Comparison of different headlight prefetching strategies on average segment download time.

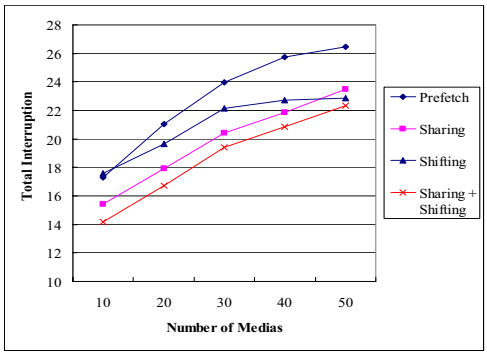

Figure 12: Play interruption with various numbers of medias.

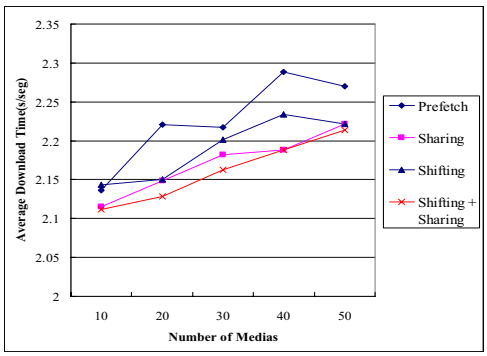

Figure 13: Average segment download time with various numbers of medias.

To evaluate the scalability of our approach, we vary the number of total medias then measure the interruption and average segment download time. Figure 12 and Figure 13 demonstrate that the increase in play interruption and download time are both limited. The combination of shifting and sharing, again leads to best result.

Traveling speed is an important factor in mobile environments since the faster the average speed, the less time we have to prepare media segments for users. Figure 14 and Figure 15 show that there is a critical point on the service performance. When the average speed is below 40, both play interruption and average download time are relative independent of the variation in speed. When the average speed is too fast such that the prefetching can no longer

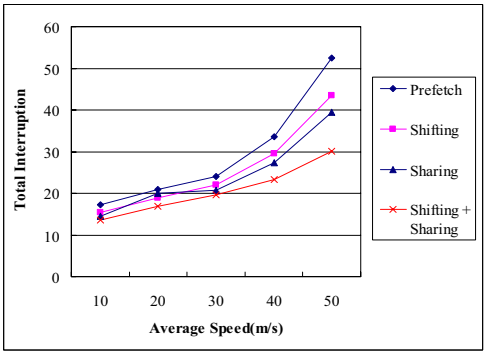

Figure 14: Play interruption with various speed levels.

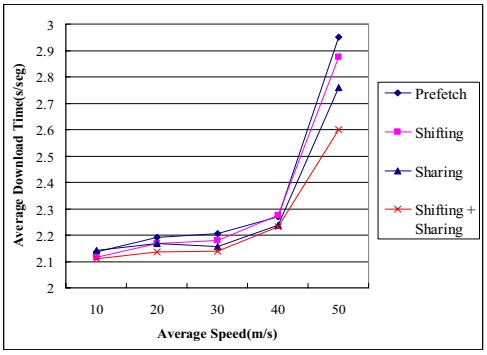

Figure 15: Average segment download time with various speed levels.

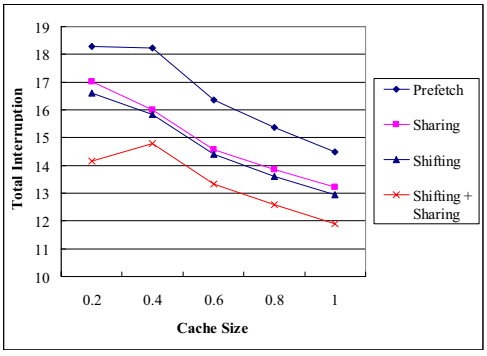

Figure 16: Play interruption with various cache sizes.

catch up with the mobile users, we observe a significant increase in both play interruption and download time, especially the later.

The size of client cache also has significant impact on system performance since the larger the cache, the more we can share with others. Figure 16 and Figure 17 demonstrate the performance of headlight prefetching on different cache sizes where the size is measured as the portion of an average media size. Both figures indicate that our system can take advantages of larger cache to provide better services. We note that a cache size 1 does not necessarily imply zero interruption since a media is started after 2% buffering.

As discussed in Section 4.3, headlight sharing has the benefit of reducing the cost of remote access. Figure 18 demonstrates the effect by comparing the number of messages that need to be sent between headlight prefetching and headlight sharing. The saving is especially evident when the number of total medias is small since

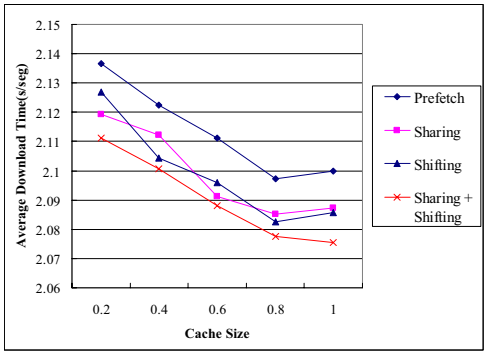

Figure 17: Average segment download time with various cache sizes.

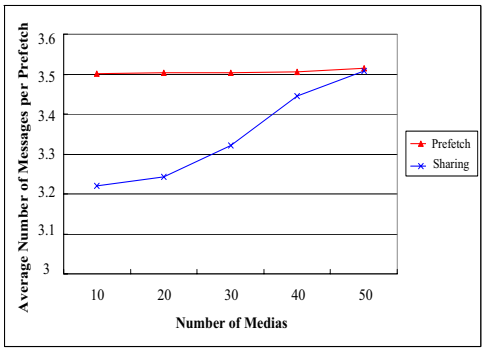

Figure 18: The prefetch message reduction on headlight sharing.

the chance of successful sharing increase.

As a summary, the set of headlight prefetching techniques are flexible and effective for media streaming services in mobile environments. Headlight prefetching provides a simple but uniform mechanism to allocate the resources on SAPs to the prefetching of needed media segments to the places where a user is most likely to be in the near future. When the movement pattern of a user does not follow the predicted track, headlight shifting and sharing leverage off the downloaded segments to quickly provide seamless media streaming services along the new track. Simulation and performance evaluation demonstrate that the headlight prefetching technique with the help of both headlight shifting and sharing, provides the best performance overall.

6. CONCLUSIONS AND FUTURE WORK

We have proposed a new set of techniques to facilitate data management for media streaming in mobile environments. The headlight prefetching techniques provide good performance in comparison with traditional on-demand or simple prefetching techniques. To offer even better data management in response to the changes in user behavior such as access and moving patterns, we are developing an adaptive headlight prefetching technique such that the shape and size of the headlight zone can be dynamically adjusted to accommodate the changes in speed or direction. We are also developing a P2P dynamic chaining method for the sharing of information among peers to maximize cache utilization and streaming benefit.

7. REFERENCES

[1] G. Anastasi, M. Conti, E. Gregori, A. Passarella, and L. Pelusi. An energy-efficient protocol for multimedia streaming in a mobile environment. *Journal of Pervasive Computing and Communications*, 1(4):301–312, Dec 2005.

[2] A. Boni, E. Launay, T. Mienville, and P. Stuckmann. Multimedia broadcast multicast service - technology overview and service aspects. In *5th IEE International Conference on 3G Mobile Communication Technologies*, pages 634–638, 2004.

[3] G. Faria, J. A. Henriksson, E. Stare, and P. Talmola. DVB-H: Digital broadcast services to handheld devices. *Proceedings of the IEEE*, 94(1):194–209, Jan 2006.

[4] F. Fitzek and M. Reisslein. A prefetching protocol for continuous media streaming in wireless environments. *IEEE Journal on Selected Areas in Communications*, 19(10):2015–2028, Oct 2001.

[5] K. Grajski. MediaFLO technology and implementation. In *Proceedings of NAB2006*, Las Vegas, April 2006.

[6] M. Guo, M. H. Ammar, and E. W. Zengura. V3: A vehicle-to-vehicle live video streaming architecture. *Pervasive and Mobile Computing*, 1(4):404–424, Dec 2005.

[7] M. M. Hefeeda and B. K. Bhargava. On-demand media streaming over the internet. In *9th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS 2003)*, pages 279–285, May 2003.

[8] C.-M. Huang and T.-H. Hsu. A user-aware prefetching mechanism for video streaming. *World Wide Web Journal*, 6(4):353–374, Dec 2003.

[9] A. Kyriakidou, N. Karelos, and A. Delis. Video-streaming for fast moving users in 3g mobile networks. In *MobiDE 2005: 4th International ACM Workshop on Data Engineering for Wireless and Mobile Access*, pages 65–72, 2005.

[10] B. Li and K. H. Wang. NonStop: Continuous multimedia streaming in wireless ad hoc networks with node mobility. *IEEE Journal on Selected Areas in Communications*, 21(10):1627–1641, Dec 2003.

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

[12] M. Qin, R. Zimmermann, and L. S. Liu. Supporting multimedia streaming between mobile peers with link availability prediction. In *Proceedings of the 13th annual ACM international conference on Multimedia*, Singapore, November 6-12, 2005.

[13] J. Wang, R. Sinnarajah, T. Chen, Y. Wei, and E. Tiedemann. Broadcast and multicast services in cdma2000. *IEEE Communications Magazine*, 42(2):76–82, Feb 2004.

[14] D. Wu, Y. Hou, W. Zhu, Y.-Q. Zhang, and J. Peha. Streaming video over the internet: approaches and directions. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(3):282–300, Mar 2001.

[15] G.-t. Xue, Z. qing Jia, J. yuan You, and M. lu Li. Group mobility model in mobile peer-to-peer media streaming system. In *Proceedings of the 2004 IEEE International Conference on Services Computing (SCC'04)*, pages 527–530, Sep 2004.

[16] J. Zhai, X. Li, and Q. Li. Statistical buffering for streaming media data access in a mobile environment. In *Proceedings of the 2006 ACM Symposium on Applied Computing*, pages 1161–1165, 2006.