



Collaborative Multimedia Presentations in Mobile Environments

B. PRABHAKARAN

prabha@comp.nus.edu.sg

Department of Computer Science, School of Computing, National University of Singapore, Singapore 119260

Abstract. Distributed multimedia documents systems, distributed video servers are examples of multimedia presentations involving collaboration among multiple information sources. In such applications, objects have to be retrieved from their sources and presented to users according to specified temporal relationships. Objects retrieval in these collaborative applications is influenced by their presentation times, durations, and network throughput available to their sources. Replication of objects amongst the set of collaborating systems gives a choice for object retrieval. Client going through a multimedia presentation can be in a mobile environment. Here, object retrievals from collaborating servers are carried out by base stations to which the client is attached. Mobile client then downloads objects from the base station.

In this paper, we present a graph-search based algorithm for computing and negotiating throughput requirements of collaborating multimedia presentations with replicated objects in a mobile environment. This algorithm maximizes the number of cached objects (that have already been played) for handling operations such as reverse presentation.

Keywords: multimedia presentations, mobile environments, retrieval scheduling, graph-search strategy, resource reservation

1. Introduction

Objects composing a multimedia presentation can be distributed over a set of collaborating systems. As examples, we can consider distributed multimedia documents systems and distributed video servers. In a distributed multimedia documents system, objects composing the document may be stored in a set of collaborating systems. In a similar manner, blocks of a video can be distributed over a set of collaborating servers. In these applications, objects have to be retrieved from multiple sources and presented to users according to specified temporal relationships. Figure 1 describes an example collaborative multimedia document presentation scenario. Here, six objects (O_1 through O_6) composing a multimedia document are distributed over systems S_1 , S_2 , and S_3 . These objects are to be presented to the mobile client system according to the temporal relationships shown in figure 2.

In a mobile environment, clients are attached to a base station (as shown in figure 1). Base station is connected to both the wired and wireless network, and hence it handles the information exchange between media sources in the wired network and the mobile client. Base station controls the clients' communication within a *cell*. A cell can be considered as a region that is within the (communication) reach of a base station. When a client moves out of a cell and enters into a new cell, communication with the client has to be handled by the base station for the new cell. This process of transferring control of a client's communication from one base station to another is referred to as *handoff*.

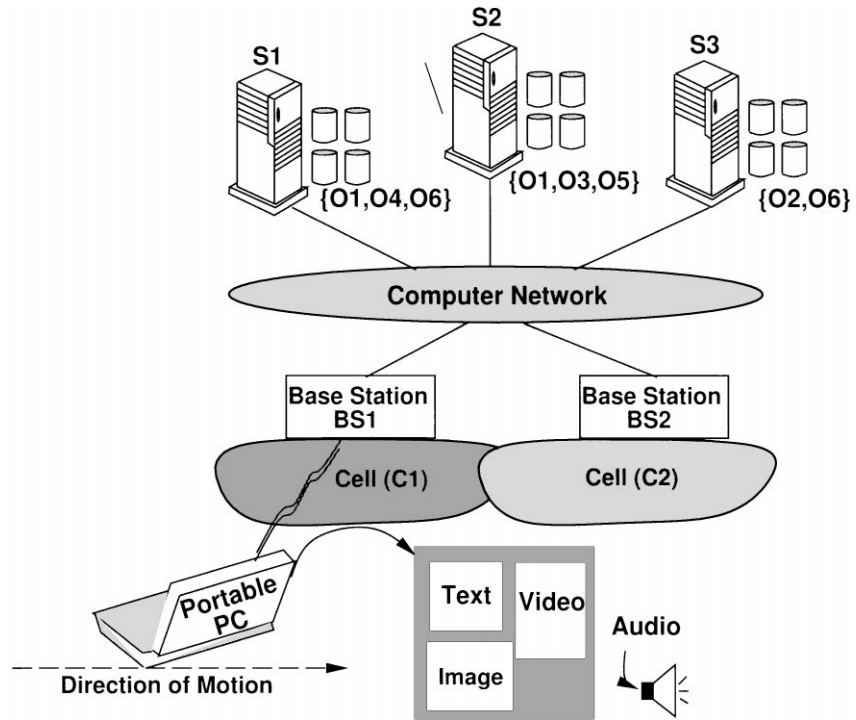


Figure 1. Collaborative multimedia presentation in mobile environment.

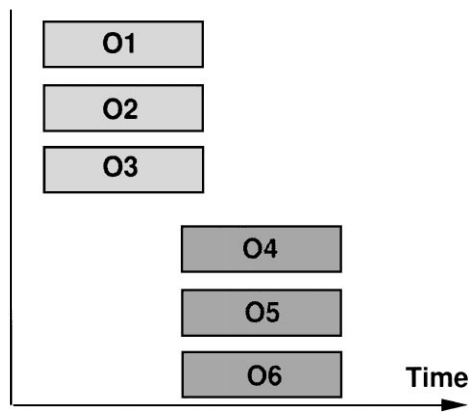


Figure 2. Collaborative multimedia document presentation example.

For a collaborative multimedia presentation in a mobile environment, base station (to which the client is attached) needs to retrieve objects from their sources. Hence, the base station has to compute and negotiate the required resources from the network service provider to each of the media sources. The client can then download the objects from the base station over the wireless network. When the client moves into a new cell, a new base station (that is in-charge of the new cell) takes over the control of the client's communication with the wired network. Further retrieval of objects composing the multimedia presentation needs to be handled by the new base station. As an example, in figure 1, base station *BS2* starts handling the client's communication once it enters into cell *C2*.

When a client going through a multimedia presentation enters into a new cell, the following situations are possible.

- Objects to be presented at the client are available with the old base station (*BS1* in the example of figure 1). In this case, the new base station (*BS2*) can download the objects available from the old one during handoff.
- Objects to be presented at the client have to be retrieved from the collaborating media sources by the new base station. Hence, the new base station needs to compute and negotiate the required resources with these media sources.
- A combination of the above situations is also possible: some objects may be available in the old base station and some might have to be retrieved from the media sources.

Resource negotiation for multimedia presentations in mobile environments. Retrieval and presentation of objects composing a multimedia presentation in mobile, collaborative applications is influenced by their presentation times, durations, and network throughput available to their sources. As the first step, the following parameters need to be computed for retrieving objects from their sources: throughput, time, and duration of throughput requirement. This computed throughput has to be negotiated with the network service provider. Based on the offered throughput by the network service provider, client can determine a retrieval schedule for the objects composing the presentation. Replication of objects amongst the set of collaborating systems gives a choice for object retrieval. Sequence of a multimedia presentation can be modified by operations such as reverse presentation (delivering objects in the reverse order specified in the document), skip time interval, and scaling the speed. These modifications influence the throughput requirements and object retrieval schedules.

Our approach. In this paper, we present a graph-search based algorithm for computing and negotiating the throughput requirements of collaborative multimedia presentations with replicated objects in a mobile environment. This algorithm maximizes on the number of cached objects (that have already been played) for handling operations such as reverse presentation. Reverse presentation is carried out by delivering objects that have already been presented in the reverse order (similar to rewind VCR operation, difference being that reverse presentation is done at the normal speed). Hence, caching objects that have already been presented helps in handling reverse presentation. It also handles operations that modify multimedia presentation sequence such as skip time interval and scaling speed.

Table 1. Symbols used.

1	$MS(O_i)$	Non-empty set of media source(s) storing O_i
2	PST	Presentation start time
3	RST	Retrieval start time
4	$AT(O_i)$	Arrival time of object O_i
5	C_{O_i}	Network channel throughput for retrieving object O_i
6	Z_{O_i}	Size of object O_i in bits
7	T_{O_i}	Time of object presentation
8	L	Lead time before start of presentation
9	RO	Current set of objects to be retrieved
10	PO	Set of objects being presented
11	FO	Set of objects that have already been presented
12	AO	Set of arriving objects
13	$B(t)$	Buffer requirement at time t
14	SI	Skip time interval
15	F	Scale speed factor
16	BS	Base station

2. Resource negotiation algorithm

We can consider graph-search control strategy as a means of finding a path in a graph from a (root) node representing initial condition to a node satisfying the termination condition [13]. For the resource negotiation problem being discussed in this paper, initial condition represents the start of a multimedia presentation and the termination condition represents the complete schedule for all objects arrival from their sources (i.e., we do not have any more objects to be scheduled for the presentation). Each *selected* node in the path (from the initial condition to the termination one) represents the chosen schedule for one set of objects composing the presentation. This section describes the graph-search control strategy for the collaborative resource negotiation problem.

The symbols used in the discussion are summarized in Table 1. First step in resource negotiation is to estimate bounds (upper and lower) on the throughput requirements. For this, identify the first set of objects to be presented, RO_1 . Let us assume that user has an upper bound on the *lead* time before the start of presentation. Let this bound be L , i.e., this is the maximum period of time user is willing to wait before s/he sees the multimedia presentation. Hence, the lower bound on throughput for each object O_i that are to be initially presented can be estimated as: $C_{O_i} = \frac{Z_{O_i}}{L}$, where Z_{O_i} is the size of the object in bits. (This represents the lower bound since a lower throughput would lead to a lead time greater than the one specified by client). Let $MS(RO_1)$ be the set of media sources from where the required objects can be retrieved. The throughput estimate, C_{O_i} , is negotiated for the corresponding members of the set $MS(RO_1)$ with the network service provider. It should be noted here that if object(s) in RO_1 has been replicated, then the negotiation will be done with all the replicated sources.

Heuristic evaluation function. For the graph-search algorithm, the initial set of objects to be retrieved represents the root node. Throughputs offered by the network service provider to different sources of an object (if the object is replicated) represents successor nodes for the root node. Now, we need a heuristic evaluation function to order and select one of the successor nodes. For the first set of objects, we select the source that offers minimum lead time before its presentation. In otherwords, we use the maximum offered throughput to retrieve an object. Let C'_{O_i} be the maximum throughput offered by the network to the media source(s) of object O_i . (In case of same C'_{O_i} being available to more than one source that can offer O_i , one source is arbitrarily selected). Now, we can determine the modified lead time (L') for the entire presentation as the maximum of the lead times needed for each object in RO_1 , i.e., $L' = \max_i (\frac{L_{O_i}}{C'_{O_i}})$. Now, the retrieval start time of presentation, RST , is: $RST = T_{O_i} - L'$. In effect, the heuristic evaluation function chooses the successor that has the minimal lead time in retrieving the object (i.e., we use the maximum throughput offered to retrieve an object). This helps the user to start viewing the multimedia presentation faster.

Client can now request the media sources to provide the respective objects according to the computed lead time. (This corresponds to the retrieval schedule for the objects). Buffer requirements of the initial set of objects is: $B(t) = \sum_i C_{O_i} \times t$. Available buffer space $B_{avail}(t)$ should be greater or equal to this value. In a mobile environment, available buffer space can be distributed on both base station and the client, i.e., $B_{avail} = B_{client} + B_{base\ station}$.

Example. Let us consider the collaborative multimedia document example discussed in Section 1. Initial set of objects to be retrieved RO_1 is $\{O_1, O_2, O_3\}$, as shown in figure 3. Media sources for this set, $MS(RO_1)$, is $\{\{S1, S2\}, \{S3\}, \{S2\}\}$. After computing the required network throughput for each object retrieval, client negotiates with each media source in $MS(RO_1)$. Figure 3(b) describes the tree of decision nodes using the graph-search based algorithm. Root node is the multimedia presentation start node. This node is expanded to generate two successor nodes: 2 and 3. Node 2 represents retrieval of RO_1 from $\{S1, S2, S3\}$ and node 3, the retrieval from $\{S2, S3, S2\}$. In this example, we assume that node 3 represents a minimal lead time and hence this node is selected for further expansion. (It also implies that we have selected an arrival schedule of objects from media sources $S2, S3$, and $S2$).

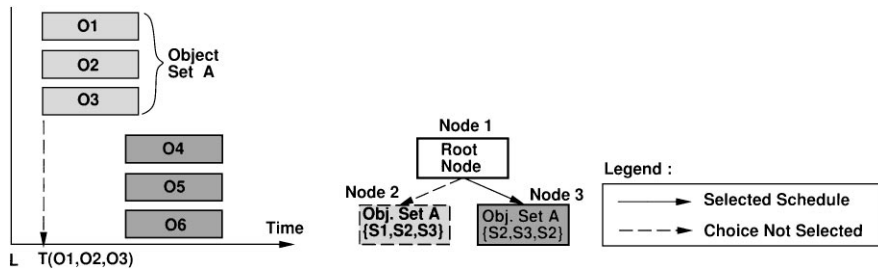


Figure 3. Computing arrival schedule of first set of objects.

2.1. Retrieving subsequent objects

Identify the next set of objects to be retrieved, RO_2 ($\forall_i \forall_j T_{O_i}$ is the next higher object presentation time after T_{O_j} , $O_i \in RO_1 \wedge O_j \in RO_2$). Bound on throughput for this set RO_2 can be estimated as follows. Lower bound on the throughput implies that the object retrieval would be started earlier and hence buffer space for the arriving object need to be available for a longer period of time. Earliest an object can be retrieved is the *retrieval start time* (RST), i.e., the time at which initial retrieval is started. Hence, the lower bound on throughput for retrieval is: $C_{O_i}^{\text{low}} = \frac{Z_{O_i}}{T_{O_i} - \text{RST}}$.

$MS(O_i)$ represents the set of media sources that can offer the object O_i . Client negotiates the required throughput for each of the sources in $MS(O_i)$. Network provides a throughput C'_{O_i} to each of the media sources that can offer object O_i . We can consider those network channels for which the offered throughput is above the lower bound i.e., $C_{O_i}^{\text{low}} \leq C'_{O_i}$. Working backwards, we can compute a set of arrival times of object O_i (at the client) from each of the media sources as $AT_{O_i} = \{T_{O_i} - \frac{Z_{O_i}}{C'_{O_i}}\}$. This procedure can be repeated for all objects in the set RO_2 .

Heuristic evaluation function. Throughputs offered by the network service provider to different sources of an object (if the object is replicated) represents successor nodes for the node being expanded (i.e., the set of objects to be scheduled next). We need to order the successor nodes using a heuristic evaluation function.

The arrival times of objects ($AT(O_i)$) in the set RO_2 may overlap. Hence the buffer requirements for the retrieval of RO_2 need to be checked against the maximum availability according to the relation, $\forall_t B(t) = (B_{PO}(t) + B_{FO}(t) + B_{AO}(t))$ (i.e., buffer requirement at time t is the sum of buffer requirements at t for the sets of playing objects (PO), played objects (FO), and arriving objects (AO)). For avoiding buffer overflow, the required buffer space should be less than the available one (i.e., $B(t) \leq B_{\text{avail}}(t)$). For handling reverse presentation user input, we do not release played objects (FO). If buffer requirements overshoot the maximum availability at any point in time, then some object in the *FinishedObjects* set (FO) need to be released. However, different combinations of arrival times of the objects in RO_2 can lead to different buffer requirements. Heuristic evaluation function orders the successor nodes and selects one as follows.

- If $B(t) \leq B_{\text{avail}}$, then follow the same methodology used for retrieval of the initial set of objects, i.e., choose the successor that has the minimal lead time in retrieving the set of objects. Hence, we use the maximum throughput offered to retrieve an object. Motivation for selecting this heuristic evaluation is: when we choose minimal lead time for retrieving an object, we are in effect postponing the sequence of presentation by using operations such as reverse presentation or skip time interval.
- If $B(t) > B_{\text{avail}}(t)$ at any point in time, then we choose the set of arrival times for objects in RO_2 ($AT(RO_2)$) that requires minimal release of buffer space (i.e., we maximize the number of objects held in FO , in order to handle reverse presentation requests).

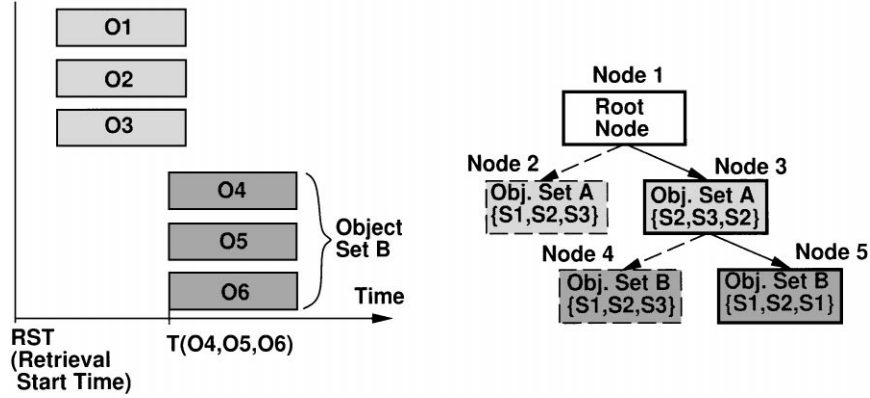


Figure 4. Computing arrival schedule for second set of objects.

Example. Figure 4 describes the steps involved in identifying the arrival schedule for the next set of objects in the example multimedia presentation shown in figure 1, $RO_2 = \{O_4, O_5, O_6\}$. The set of media sources that can offer these objects is $MS(RO_2) = \{\{S1\}, \{S2\}, \{S3, S1\}\}$. After computing throughput requirements for retrieving each object in RO_2 , client negotiates with the media sources in $MS(RO_2)$. Figure 4(b) describes the tree of decision nodes using graph-search algorithm. Node 4 represents retrieval of RO_2 from $\{S1, S2, S3\}$ and node 5, the retrieval from $\{S1, S2, S1\}$. In this example, we assume that node 5 represents retrieval requiring minimal release of played objects FO from cache.

2.2. Handling mobility of the client

When client going through a multimedia presentation is mobile, it can move from one cell to another. Base station that is in charge of the particular cell then handles the objects retrieval from the collaborating media sources. During the expansion of nodes (i.e., while scheduling objects retrieval from their sources), we need to consider the mobility of the client as well. A node can be expanded depending on the probability of client's mobility. As an example, in figure 5, node 6 is expanded with probability for client attaching to base station $BS2$ ($p(BS2)$) and the probability of the client staying with the same base station $BS1$ ($p(BS1)$). During handoff of client to another base station, playing objects (set PO) and played objects (set FO) can be transferred to the new base station (to facilitate handling of reverse presentation). Further scheduling of objects retrieval (i.e., further expansion of nodes in the search graph) is done for the new base station.

Probability of client's mobility can be estimated using shadow cluster concept proposed by Levine et al. [8], for resource reservation in mobile environments. Shadow cluster concept determines probability of a client moving into a cell based on its direction and rate of movement, as shown in figure 6.

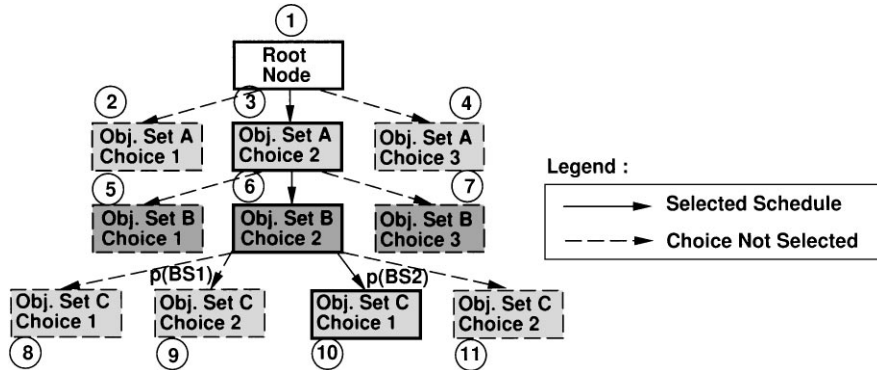


Figure 5. Schedule tree generated by graph-search based algorithm.

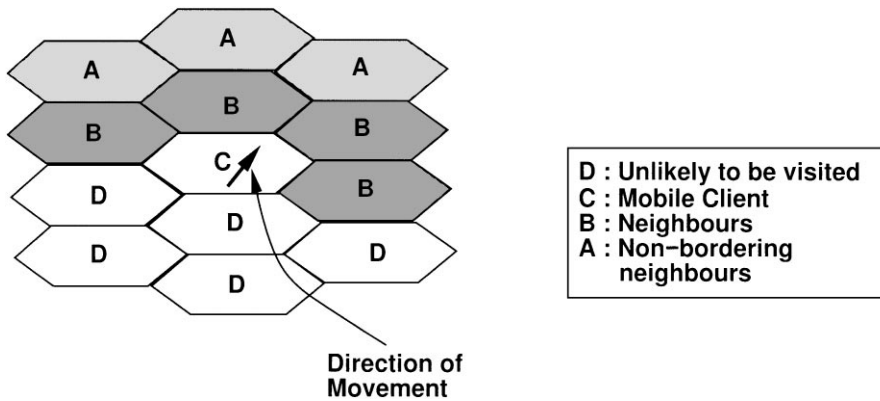


Figure 6. Collaborative multimedia presentation in mobile environments.

2.3. Negotiation algorithm

The above sequence of computation and negotiation is summarized briefly in the following Algorithm A.1.

Algorithm A.1. Create multimedia presentation

1. Create a search tree T consisting of solely the start node, representing the set of objects to be retrieved, RO_1 .
2. *Expand* this node: Compute the throughput requirements for retrieving each object in RO_1 from its source(s). For each combination of media sources, compute the buffer requirements $B(t)$ over the retrieval time. Each combination gives a successor for the node being expanded.

3. Expansion of nodes in Algorithm A.1 is also made taking into account the probability of client mobility. When the client is *handed over* to another base station, objects in the set PO and FO can be transferred to the new base station. Further execution of Algorithm A.1 is done for the new base station.
4. Order and select one of the successor nodes using the following heuristic evaluation function f .
 - f_a : Order the nodes based on the lead time for presenting the set of objects, when $B(t) \leq B_{\text{avail}}(t)$. Select the node that gives minimal lead time for retrieving the set of objects.
 - f_b : Order the nodes based on the buffer space that need to be released for accommodating the objects to be retrieved, when $B(t) > B_{\text{avail}}(t)$ at any point in time. Select the successor node that requires minimal release of buffer space, i.e., we maximize the number of cached objects (that have already been played), in order to handle reverse presentations.
5. Identify the next set of objects RO_i to be presented such that $\forall_i \forall_j T_{O_k}$ is the next object presentation time after T_{O_l} , where $O_k \in RO_i \wedge O_l \in RO_j \wedge i > j$.
6. Repeat step 2 until arrival schedule for all objects composing the multimedia presentation have been computed.

Handling insufficient resources. If the offered network throughput or available buffer space is insufficient, we can consider reducing the *quality* of an object. Quality of an object can be modified by playing with parameters such as resolution. Modifying object qualities has been described in [2, 19]. We do not go into the details of object quality modification in this paper. Also, we do not discuss the choice of objects to be released in case of buffer overflow. Standard techniques such as *least recently used* can be used for this purpose.

2.4. Properties of the resource negotiation algorithm

The resource negotiation Algorithm A.1 discussed above generates an efficient objects retrieval schedule in terms of the following parameters.

- Minimize the lead time for objects retrieval if there is no buffer overflow. This strategy has the following implications. It forces the lead time for retrieval of initial set of objects to be minimal, hence the presentation could start as early as possible. For subsequent set of objects, this strategy postpones object retrieval as much as possible (within the limit of object presentation start time). Hence, if operations such as reverse presentation or skip time interval modify the presentation sequence and the set of objects are not needed for presentation, then the cost of retrieval can be minimized as much as possible.
- Minimize the release of buffer in case of overflow. Hence, we maximize the number of (already played) objects that can be held in the buffer.

Algorithm A.1 has the following properties.

Theorem 2.1. *Algorithm A.1 always terminates for finite multimedia presentation sequence.*

Proof of Theorem 2.1: Let us assume the opposite that Algorithm A.1 does not terminate. Termination is prevented only if new nodes are forever added to the decision tree created by Algorithm A.1. Addition of a new node implies scheduling arrival time(s) of new multimedia object(s). However, for a finite multimedia presentation, number of objects to be scheduled is finite. Hence, termination of A.1 is *not* prevented for finite multimedia presentation sequences. \square

Theorem 2.2. *If there is a retrieval schedule that uses minimum client buffer for a finite multimedia presentation sequence, Algorithm A.1 terminates by finding the schedule.*

Proof of Theorem 2.2: The heuristic evaluation function f used by Algorithm A.1 chooses a successor node as follows:

- If $B(t) \leq B_{\text{avail}}$, select the node that represents the minimal lead time for objects retrieval.
- If $B(t) > B_{\text{avail}}(t)$ at any point in time, select the node that requires minimal release of buffer space.

This means that when Algorithm A.1 selects the successor to a node n , the chosen successor uses minimum client buffer for the particular set of objects. The above argument implies that at *any point* in time, the nodes chosen by Algorithm A.1 represent the optimal (in terms of buffer usage) schedule for those sets of objects. We have (by Theorem 2.1) that Algorithm A.1 always terminates for finite presentations. Hence, Algorithm A.1 always terminates by finding a retrieval schedule that uses minimum client buffer for finite sequence of multimedia presentations. \square

3. Handling presentation sequence modification

Sequence of presentation of a multimedia document can be modified by operations such as reverse presentation, skip, and scaling the speed. These operations modify the resource requirements for the multimedia presentation. At each step of the resource negotiation algorithm (when the client has finalized on the media source and object arrival schedule), we can compute a profile for throughput requirements to each media source, as shown in figure 7. This throughput profile can be used in handling presentation sequence modification. In this section, we will discuss how the graph-based algorithm handles modifications to presentation sequence.

Reverse presentation. This operation directs the sequence of a multimedia presentation backwards in the timeline. Objects that were already played need to be presented again according to temporal relationships in the reverse direction. Reverse presentation will be handled in the following manner.

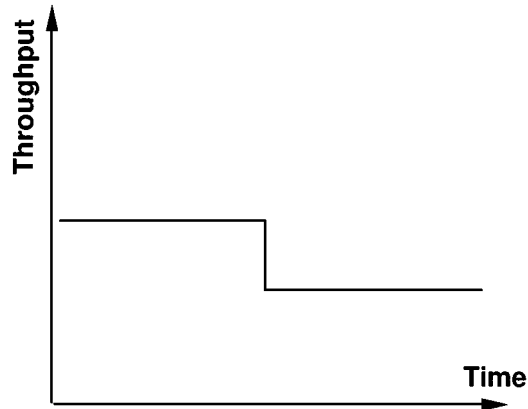


Figure 7. Throughput profile for communication with object source S_2 .

1. First step is to stop the arriving objects (if any) as they will not be useful for reverse presentation.
2. Present objects in the set PO (playing objects) and then objects in the set FO (played objects) according to deadlines in the reverse direction.
3. Check the throughput profile to collaborating media sources to determine whether other objects to be presented in the reverse mode can be retrieved using the already agreed throughput. If so, compute the arrival schedule for objects and negotiate with the media sources for delivering the required objects. For this invoke Algorithm A.1 with the restriction that network throughput is not negotiated with media sources. This negotiation step is needed to ensure that sufficient disk bandwidth is available at each media source for delivering the required object.
4. Invoke Algorithm A.1 for objects that cannot be scheduled using the existing throughput profile.

Skip time interval. Here, the client has to start viewing objects that will be presented (in the normal presentation sequence) after an interval SI . This operation can be handled in the following manner.

1. Identify the set of objects RO_i that need to be presented at after the skip interval SI . If the objects are part of playing objects (set PO) or arriving object (set AO), then the client can start the presentation once the objects are available. (This step is usually valid when skip interval is very small).
2. For objects that are not in PO and AO , check the throughput profile to media sources to see if the required throughput is available. If available, the media source(s) can be used for further retrieval operation without network resource negotiation. However, the objects that need to be retrieved may (in most cases, will) be different from the one that was committed to by the media sources. Hence, the client has to negotiate with the media sources to ensure availability of sufficient disk bandwidth for retrieving the new set of

objects. For this, invoke Algorithm A.1 with the restriction that network throughput is not negotiated with media sources.

3. Invoke Algorithm A.1 for objects that cannot be scheduled using the existing throughput profile.

Scale speed. Duration of presentation objects have to be scaled up or down by the given factor F . Presentation time of objects T_{O_i} will also be modified based on F . Scaling speed of presentation can be handled in the following manner.

1. For scaling the speed down:

- (Client need not renegotiate the resources that have already been reserved. However, it can compute an object arrival schedule that helps in maximising the objects held in cache.)

Compute an object arrival schedule that maximises the objects in cache based on the throughput profile to each media source. For this, invoke Algorithm A.1 with the restriction that network throughput is not negotiated with media sources.

- For resources that have not been reserved, Algorithm A.1 has to be executed with modified object presentation times.

2. For scaling the speed up:

- Check the reserved throughput profile to each media source. If sufficient resources are available, execute Algorithm A.1 (without making network resource reservation) to compute modified object arrival schedule.
- Invoke Algorithm A.1 for objects that cannot be scheduled using the existing throughput profile.

Choices for a mobile client. Caching of objects help in handling dynamic modifications to presentation sequence. As discussed earlier, buffer space for objects caching is distributed on the client and the base station. During handoff, objects in the cache of old base station can be transferred to the new base station. If the available buffer space on the new base station is lower than that of the old one, some objects might have to be dropped from the cache. If the client is too mobile, it might be better not to transfer all the objects to the new base station(s). Another issue is the start time of objects retrieval at the new base station. This depends on the resource availability between the base station and the collaborating media sources. If the available resources are very low, objects retrieval can be started even before the client actually moves into the new cell.

4. Related work

Issues in wireless mobile computing environments are discussed in [5]. Management of data and querying aspects of databases are presented in [6, 7]. Issues and solutions for presentation of video objects in a mobile environment is described in [12]. Our work discussed in this paper is more general in that it deals with diverse media objects and also it deals with presentation of objects from multiple sources. Support for collaborative applications in mobile environments is outlined in [4]. Here, group co-ordinator services are

provided in the International Standards Organization (ISO) Open Distributed Processing (ODP) environment and collaboration aware tools are designed using these services. In our work, we address the issue of computing and negotiating resource requirements in mobile environments. We do not deal with specific architectures such as ISO-ODP.

Multimedia authoring and presentation schedule creation have been studied by many researchers [1, 2, 9–11, 14, 15]. In a similar manner, derivation of retrieval schedules for distributed multimedia presentation has also been studied in many works, such as [9, 10, 16–19]. In [10, 11], the presentation of multimedia objects is based on Petri nets description of the temporal specification. The retrieval schedule is derived by assuming a certain throughput to be provided by the network service provider. Based on the derived retrieval schedule and the assumed network throughput, estimates for the buffer resource requirements on the client system are made. However, the proposed algorithm does not check whether the estimated buffer resources are available or not. Also, it does not handle replication of objects and variations in the throughput offered by the network service provider.

In [9], Li et al. use time-flow graphs to capture interval-based *fuzzy* presentation schedules, and synchronization of independent sources. Their algorithms guarantee that there will be no gaps in the source's schedules. However, they do not address to the issues of object replication and constraints on resources such as throughput and buffer. As in [10], in [16–18], authors use Petri net model to describe temporal specifications, and they base the retrieval schedules on the fixed presentation schedules. In [19], Thimm et al. describe a method which adapts the presentation schedule to the changes in the resource availability by modifying the overall quality of the presentation.

In [2], authors have proposed a flexible retrieval schedule algorithm. The algorithm assumes a maximum throughput Th_{max} to be made available by the network to a data source. This throughput Th_{max} is then *allocated* for different object retrievals. In our approach, we determine upper and lower bounds on throughput requirements for each object retrieval. Hence, our algorithm handles network resource negotiation in a better manner. However, our algorithm does not deal with flexible presentation schedules. Also, in [2], authors do not deal with replicated object retrievals and buffers used for object retrieval is released immediately after presentation. In our approach, we minimize the amount of buffer released so that operations like reverse presentation can be handled relatively easily.

[3] suggests the use of interval caching policy. The suggestion is more suitable for handling multiple retrievals of same video objects. Also, the amount of caching suggested is based on time interval. In our approach, we maximize the number of played objects that can be held in the buffer at each stage.

5. Summary and conclusion

Collaborative multimedia presentation involves retrieval of objects from multiple sources. Retrieval of different media objects is influenced by their presentation times, durations, and network throughput available to their sources. Previous approaches have generated object retrieval schedules based on an assumed network throughput availability [2, 9, 11, 17]. These approaches do not handle replicated objects and mobile environments. They also do not address the issue of caching objects that were already played, for handling operations like reverse presentation.

In this paper, we have suggested a graph-search based algorithm to handle resource negotiations for collaborative multimedia presentations in a mobile environment. The advantage of this approach is that it handles replicated objects and also maximizes the number of objects that can be held in cache for handling modifications to presentation sequence (e.g., reverse presentation or skip).

References

1. M.C. Buchanan and P.T. Zellweger, "Automatic Temporal Layout Mechanisms," *ACM Multimedia*, Vol. 93, pp. 341–350, 1993.
2. K.S. Candan, B. Prabhakaran, and V.S. Subrahmanian, "CHIMP: A Framework for Supporting Distributed Multimedia Document Authoring and Presentation," *Fourth ACM International Multimedia Conference*, Boston, Nov. 1996, pp. 329–340.
3. A. Dan and D. Sitaram, "A generalized interval caching policy for mixed interactive and long video workloads," *Multimedia Computing and Networking*, San Jose, Jan. 1996.
4. N. Davies, G.S. Blair, K. Cheverst, and A. Friday, "Supporting collaborative applications in a heterogeneous mobile environment," *Special Issue of Computer Communications on Mobile Computing*, 1995.
5. D. Duchamp, "Issues in wireless mobile computing," *Proc. 3rd Workshop on Workstation Operating Systems (WWOS)*, IEEE Computer Society Press: Key Biscayne, Florida, US, 1992.
6. T. Imielinski and B.R. Badrinath, "Querying in highly mobile distributed environment," *Proc. of 18th VLDB Conference*, Vancouver, Canada, 1992.
7. T. Imielinski and B.R. Badrinath, "Data management for mobile computing," *SIGMOD Record*, Vol. 22, No. 1, pp. 34–39, 1993.
8. D. Levine, M. Naghshineh, and I. Akildyz, "Shadow cluster method: Resource estimation and call admission method in multimedia mobile networks," *IEEE/ACM Transactions on Networking*, Vol. 5, No. 1, 1997.
9. L. Li, A. Karmouch, and N.D. Georganas, "Multimedia teleorchestra with independent sources: Part 1 and Part 2," *ACM/Springer-Verlag Journal of Multimedia Systems*, Vol. 1, No. 4, pp. 143–165, 1994.
10. T.D.C. Little and A. Ghafoor, "Synchronization and storage models for multimedia objects," *IEEE J. on Selected Areas of Communications*, Vol. 8, No. 3, pp. 413–427, 1990.
11. T.D.C. Little and A. Ghafoor, "Scheduling of bandwidth-constrained multiMedia traffic," *Computer Communication*, Butterworth-Heinemann, pp. 381–388, July/Aug. 1992.
12. B. Mah, S. Seshan, K. Keeton, R. Katz, and D. Ferrari, "Providing network video services to mobile clients," *Proc. 4th Workshop on Workstation Operating Systems (WWOS)*, Napa, US, Oct. 1993, pp. 48–54.
13. N.J. Nilsson, "Principles of Artificial Intelligence," Morgan Kaufmann Publishers: Los Altos, California, 1980.
14. B. Prabhakaran, "Multimedia synchronization," *Multimedia Systems and Techniques*, B. Furht (Ed.), Kluwer Academic Publishers, pp. 177–214, 1996.
15. B. Prabhakaran and S.V. Raghavan, "Synchronization models for multimedia presentation with user participation," *ACM/Springer-Verlag Journal of Multimedia Systems*, Vol. 2, No. 2, pp. 53–62, 1994. Also in the *Proceedings of the First ACM Conference on MultiMedia Systems*, Anaheim, California, Aug. 1993, pp. 157–166.
16. N.U. Qazi, M. Woo, and A. Ghafoor, "A synchronization and communication model for distributed multimedia objects," *Proc. of the First ACM Conference on MultiMedia Systems*, Anaheim, California, Aug. 1993, pp. 147–155.
17. S.V. Raghavan, B. Prabhakaran, and S.K. Tripathi, "Synchronization representation and traffic source modeling in orchestrated presentation," *Special issue on Multimedia Synchronization*, *IEEE Journal on Selected Areas in Communication*, Jan. 1995.
18. S.V. Raghavan, B. Prabhakaran, and S.K. Tripathi, "Handling QoS negotiations in orchestrated multimedia presentations," *Journal of High Speed Networking*, Vol. 5, No. 3, pp. 277–292, 1996.
19. H. Thimm and W. Klas, " δ -Sets for optimal reactive adaptive playout management in distributed multimedia database systems," *12th International Conference on Data Engineering*, Feb. 1996, pp. 584–592.



B. Prabhakaran is currently a faculty with the School of Computing, National University of Singapore. He was a visiting research faculty at the Computer Science department of the University of Maryland, College Park. He was also a faculty with the Department of Computer Science and Engineering, Indian Institute of Technology, Madras, India.

Prabhakaran has been working in the area of network requirements for distributed multimedia systems with main emphasis on synchronization models, resource (network and server) management, collaborative multimedia presentations, and mobile computing environments for multimedia systems. He has published several journal and conference papers in these areas.

Prabhakaran is guest-editing special issues in the fields of multimedia presentation and mobile computing for journals such as ACM Multimedia Systems and Multimedia Tools and Applications journal. He has served as program committee member in several international conferences in the field of multimedia systems. He has also authored a book on multimedia database management systems.