2012 IEEE 18th International Conference on Parallel and Distributed Systems

A Novel Multi-Channel Data Broadcast Scheme for Multimedia Database Systems

Xiaofeng Gao
Shanghai Jiao Tong Univ., Shanghai, P.R.China
gao-xf@cs.sjtu.edu.cn

Yi Zhu
Hawaii Pacific Univ., Honolulu, USA
yzhu@hpu.edu

Donghyun Kim
North Carolina Central Univ., Durham, USA
donghyun.kim@nccu.edu

Jianzhong Li
Harbin Institute of Tech., Harbin, P.R.China
lijzh@hit.edu.cn

Weili Wu
Univ. of Texas at Dallas, Richardson, USA
weiliwu@utdallas.edu

***Abstract*—MultiMedia DataBase Management System (MMDBMS) becomes more popular in recent years, which supports complex and large multimedia data like images, audios, and videos etc. *Data Broadcasting* is an attractive approach for data dissemination to improve the limitations in mobile environment, such as narrow bandwidth, unreliable connections, and battery limitation. However, existing data broadcast schemes are inefficient for MMDBMS. In this paper, we present four novel multimedia data broadcast schemes (namely, SDAA, MDAA, AEA, and COA) specifically for wireless multichannel communications. The major strategies are scalable coding to generate data segments to different qualities, indexing and channel assignment to minimize the expected waiting time for clients. We prove theoretically that SDAA is a 2-approximation. COA performs best when we release the constraints and it can be judged as an theoretical lower bound, while AEA outputs local optimal solution with quality allocation constraints. Finally, SDAA+AEA form a best scheduling for practical applications. We also provide numerical experiments to evaluate the system performance, proving the efficiency of our schemes.**

***Keywords*-Multimedia Data Broadcast, Data Scheduling**

I. Introduction

With the advance in technology, new database management systems have been created to support different requirements from increasing number of clients. *MultiMedia DataBase Management System* (MMDBMS) is one of such novel databases, which supports multimedia data in addition to providing traditional DBMS functionalities [11]. *Multimedia Data* typically include texts, images, audios, videos, animations, graphics, and other complex formats. In recent years, the acquisition, generation, storage and processing of multimedia data in computers and transmission over networks have grown explosively. Therefore, the exchange of multimedia information becomes really important and MMDBMS gets tremendous development. The studies on MMDBMS bring more and more attention from both academia and industrial communities.

As the rapid growth of wireless technology, wireless/mobile communication becomes popular. People prefer to access information through mobile devices from anywhere at any time. However, existing wireless services are limited by many constraints such as narrow bandwidth, unreliable connection, and battery limitation. Also, the large population of mobile clients may swamp the server when they asynchronously submit queries [19]. In addition to the characteristics of wireless communication, it is more efficient for the server to broadcast *data items* to a large client population than sending data to only one single client each at a time. *Data Broadcast* hereby becomes an attractive approach for data dissemination in mobile environment. In a typical data broadcast system, a *Base Station* (BS, or server) periodically disseminates data through radio waves with constraint RF range to massive number of mobile users according to a pregenerated program by single or multiple channels. Clients wait for the appearance of the data items on broadcast channel instead of explicitly sending requests to the server for interested data. An effective data schedule assigns a program onto multichannels, which minimizes the length of broadcast cycle and improves the system performance tremendously. It is obviously an NPC problem (can be m-reduced from Parallel Job Scheduling), resulting no efficient polynomial-time exact algorithms.

When applying data broadcast scheme to multimedia databases, we face new problems. Unlike texts or other simple data files, sending multimedia data to a remote client takes significant amount of communication bandwidth [3]. The size of multimedia data is much larger than the simple data file, which costs more energies to download. The duration of broadcast cycle on each channel will also be extended, increasing the waiting time for clients. All in all, to avoid network congestion, make the system more efficient, and improve the usage of multimedia databases, we need to reconsider existing data broadcast schemes and construct a new scheme to support multimedia data dissemination. Moreover, due to the physical constraints of mobile devices (processor speed, memory/flash size, screen resolution etc.), even for the same multimedia file, different clients may require different quality copies. For example, a picture shown on a personal laptop with screen resolution 2048×1024 has size at least 2MB, while it has size of 0.4MB in the latest *iphone 4S* with screen resolution 960×640 without loss of quality. Thus, we hope the server can provide

This work is supported in part by Natural Science Foundation of Shanghai (Grant No. 12ZR1445000); National Natural Science Foundation of China (Grant No. 61202024); US National Science Foundation (NSF) CREST No. HRD-0833184; and US Army Research Office (ARO) No. W911NF-0810510.

1521-9097/12 $26.00 © 2012 IEEE
DOI 10.1109/ICPADS.2012.28

different qualities for each datum at the same time to satisfy different client requirements.

This paper presents a novel data broadcast scheme for MMDBMS in multichannel wireless communications to achieve all these targets. The major strategies are scalable coding to generate data segments with different qualities, indexing and scheduling to minimize the expected waiting time for clients. We design four algorithms (SDAA, MDAA, AEA and COA) to minimize the broadcast length. The first three have a constraint that data segments with different qualifies in one file can only be allocated on different channels. Among them, AEA is most complicated but performs best. COA solves the ideal case without the constraint. We prove that SDAA is a 2-approximation and exhibit the efficiency of our system by mass numerical experiments.

The rest of this paper is organized as follows. Section II introduces related literatures. Section III illuminates necessary conceptions and theorems. Section IV describes our system design and four algorithms. In Section V, we illustrate the system performance with numerical experiments. Finally, Section VI gives a conclusion and future works.

II. Previous Literatures

Data broadcast has been recognized as an efficient method for information dissemination, and gets increasing attractions. Imielinski [7] discussed how to organize massive amount of data on wireless communications to provide fast and low power access for users who equip with palmtops. Later, Pekowsky [13] discussed the possibility of broadcasting multimedia data, whose broadcasting architecture has been specified by the *Digital Video Broadcasting* consortium to allow service providers to multiplex data, using various protocols, into an MPEG-2 transport stream.

Based on these techniques, many works have been proposed over the past decade. Wu [15] designed a stretch optimal scheduling algorithm for on-demand broadcast systems. Yoshihisa [18] proposed methods to broadcast continuous media data by reducing client waiting time for multiple data via a single channel. Aksoy [2] summarized several existing *Dissemination Based Information Systems* (DBISs) and described their initial prototype of a DBIS toolkit. Lee [8], [9] redesigned the existing data broadcast schemes to deal with data-missing problem. Young [19] presented a novel protocol for disseminating data in broadcast environments such that view consistency, a useful correctness criterion for broadcast environments, is guaranteed. Yang [16] leveraged *Network Coding* (NC) to reduce the bandwidth consumption for data broadcast. However, all those technologies cannot deal with mass multimedia huge data streams. Some works also discussed the protocols for Video-on-Demand (VoD) transmission [4], [14], [20], but they focus on P2P networks, which is not applicable in data broadcast environment.

Wireless networking technologies have been improved rapidly recent years. Various types of networks have been designed under different standards to face the increasing requirements for wireless clients. Huang [6] developed a genetic algorithm to generate broadcast programs in a multiple channel environment. Anticaglia [1] proposed two heuristics to deal with data broadcast over multiple channels consisting nonuniform length data items. Yi [17] proposed effective generation of data broadcast schedules with different allocation numbers for multiple wireless channels.

III. Preliminaries

A. System Architecture

We illustrate a typical data broadcast system for multichannel broadcast environment in Fig. 1. It is the architecture of the system and how the data broadcast mechanism works in multi-channel wireless environment.

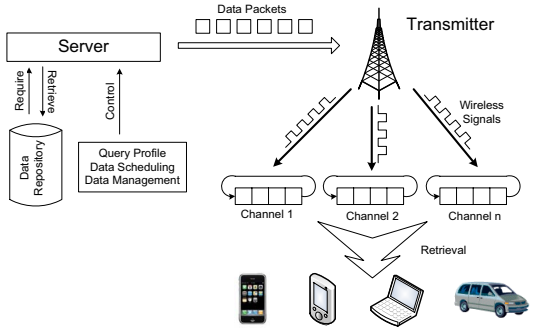


Figure 1. An Example of Data Broadcasting System

First, data are stored in a local data repository [10]. A server retrieves data items from this repository. These data items are disseminated over broadcast channels at regular intervals by a transmitter, identified by their primary key. The server performs a series of processing functions including channel assignment, scheduling, and index construction.

After data managing and scheduling, these files are referred as a broadcast program and broadcasted iteratively in a cycle until the program is expired. The program includes some auxiliary information such as data identifiers, attributes, and indices. Attributes for each data item are then used to process the query from clients. The smallest logical unit of the broadcast (called *bucket*) is assumed to be a single data item for convenience. Each cycle of the broadcast is called a *bcycle*, while the content of the broadcast is called a *bcast*. for each *bcast*, we have a series of data *slots*.

Mobile clients access onto the channel, wait the appearance of the indices, search and download data items by jumping to the target channel during appropriate time slots.

B. MIMO Antenna

Current data broadcast systems restrict clients to communication with only one channel at a time, even though the system may have multiply channels. However, with the development of antenna technology, the mobile devices now

can connect several channels simultaneously with *Multiple-Input and Multiple-Output* (MIMO) technology, which is used both in transmission and receiver equipment for wireless radio communication. MIMO is an efficient way to use multiple antennas at both transmitter's and receiver's side to improve communication performance and attain ultra high spectral efficiency [5].

MIMO technology is the kernel baseband techniques for the *Fourth-Generation Wireless Information System* (4G, Beyond 3G, Next Generation Network, NGN), which is a complete evolution and will become a total replacement of the 3G in few years. The international telecommunications regulatory and standardization bodies are working for commercial deployment of 4G roughly during 2012 to 2015.

C. Evaluation Criterions

Due to the limitation of battery capacity for mobile clients, energy efficiency is a critical issue in the design of broadcasting systems [7]. The mobile devices have dual-modes: *active mode* and *doze mode*. They can only retrieve data from broadcast channels in the active mode, which consumes much more energy than in the doze mode. Thus, there are two parameters to evaluate the broadcasting system:

Access Latency:: The period of time from when a query is issued to the moment it is responded.

Tuning Time:: the period of time a mobile client stays *active* to retrieve the requested data items.

Access latency measures the efficiency of system organization, while tuning time is used to estimate the power consumption of a mobile client to retrieve data items.

D. Data Sampling

Assume for each datum (or multimedia stream) we have K qualities $Q = \{q_1, q_2, \cdots, q_K\}$. We can use *data encoding scheme* to filter the item according to quality constraint. E.g., given an item I, let $I^1, I^2, \cdots, I^K$ denote the data segments corresponding to qualities after data encoding process. There are two classes of data encoding scheme:

Independent encoding format: High-quality parts are substituted for low-quality parts. E.g., I can be encoded according to 2 qualities. One substream I^1 contains complete images of the size $a \times b$ and another substream I^2 contains complete images of the size $2a \times 2b$. Choosing a different quality means choosing a different substream.

Hierarchical encoding format: High-quality parts are added to low-quality parts. E.g., I is encoded into I^1 and I^2 according to 2 quality constraint q_1 and q_2. I^1 may contain images of the size $a \times b$ and I^2 may contain all additional pixels that extend the format to $2a \times 2b$. To present data in the highest quality, all substreams must be presented, e.g., if a client wants item I with quality q_2, it should download both I^1 and I^2 and merge them together.

In this paper, we choose *hierarchy encoding scheme* to decompose datum for the parallel property of MIMO.

IV. A Novel Data Broadcasting Scheme

In this section, we firstly define the symbols system (in Table I) and then design our system architecture. Next, we propose four scheduling algorithms under different constraints with performance analysis and examples.

Table I
Symbol Description

Symbol	Description
S	A data broadcast system
N	No. of channels in the system
P	No. of data items in a program
K	No. of quality requirements
U	Channel constraint for one client
R	Channel broadcast rate
C	Channel set, $C = \{c_1, \cdots, c_N\}$
T	No. of channel groups, $T = \lfloor \frac{N}{K} \rfloor$ (Alg. 1) or $\frac{N}{U}$ (Others)
Q	Set of qualities, $Q = \{q_1, q_2, \cdots, q_K\}$
I_i	Data items, $1 \le i \le P$
G_i	Quality group for I_i, $G_i = \{I_i^1, \cdots, I_i^K\}$
CG_j	Channel groups, $CG_j = \{c_{(j-1)U+1}, \cdots, c_{jU}\}$ (Alg. 2)
$\mathcal{G}$	All data groups in system, $\mathcal{G} = \{G_1, \cdots, G_P\}$
$\mathcal{CG}$	All channel groups, $\mathcal{CG} = \{CG_1, \cdots, CG_T\}$
l_i^k	Length of I_i^k
lg_j	Length of channel group CG_j
n_i^k	No. of channels assigned to I_i^k

A. System Design

Given a data broadcast system S with one data repository, N channels, and K quality constraints. All channels are independent, each with the same broadcast rate R. Each client can connect with at most U channels at one time. We assume the system is reliable and synchronized, and $N > U > K$ (Usually, N can be large because the BS can deploy as many MIMO antenna as it can serve. K is very small, and U depends on the number of antennas in mobile devices and the number of pathes between sender and receiver. Furthermore, MIMO plus OFDMA can tremendously increase the possible channels a client can use at one time [12]).

When retrieving a datum I, the *source video encoder* will decompose I to several substreams with quality constraints, then allocate these packets over U channels. Data on multiple channels will broadcast simultaneously, such that the receiver will download them at the same time. Next, the receiver will merge corresponding files together to get the item it wants. The whole process can be seen as Fig. 2.

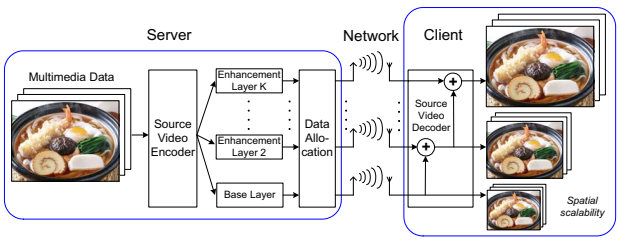


Figure 2. An Example of Broadcasting Multimedia Stream

Next, we will construct four algorithms to efficiently allocate data to minimize *access latency* and *tuning time*.

B. Simple Data Allocation Algorithm (SDAA)

Since $N > K$, the most intuitive way is to decompose each required datum I_i to K substreams as a group $G_i = \{I_i^1, I_i^2, \cdots, I_i^K\}$, then allocate them on K channels. If the length of each I_i^k is not equivalent, we leave the slot unoccupied such that $\forall p, q,\ l_i^p = l_i^q$, where $1 \le p, q \le K$ and $p \ne q$. For each datum we repeat the same process, then allocate the group to the channels with the shortest length of bcast. To simplify the system architecture, we assume $N = n \times U$ (n is a positive integer) and there are P data items at a moment. Alg. 1 describes the detailed processes.

Algorithm 1 Simple Data Allocation Algorithm-SDAA

Input: Data group $\mathcal{G} = \{G_1, G_2, \cdots, G_P\}$, each has K substreams with K qualities. Channels $\mathcal{C} = \{c_1, c_2, \cdots, c_N\}$.
Output: A broadcast schedule $Sche$.

1: Set $T = \lfloor \frac{N}{K} \rfloor$.
2: Sort $\mathcal{G}$ nondecreasingly, rename as $\mathcal{G}' = \{G'_1, \cdots, G'_P\}$.
3: Divide $\mathcal{C}$ into T groups as $\mathcal{CG} = \{CG_1, \cdots, CG_T\}$, $CG_i = \{c_{(i-1)K+1}, c_{(i-1)K+2}, \cdots, c_{iK}\}$, $\forall 1 \le i \le T$.
4: **for** $i = 1$ to P **do**
5: Choose $CG_j = \min_{1 \le j \le T}\{lg_j\}$
6: Append G'_i to CG_j on corresponding channels.
7: **end for**
8: Output $Sche$.

Theorem 1. *The time complexity of SDAA is $O(P \ln P)$.*

Proof: The only time consuming for SDAA is the sorting part for $\mathcal{G}'$, which takes $O(P \ln P)$. Other calculations only take $O(1)$. Thus the theorem holds. ■

In SDAA, each datum occupies K channels, so one client will connect at most K channels to download their required datum. Each time, SDAA appends a datum to a group of channels with shortest $dcast$ length. Table II is an example of 6 data items with data segment length to the qualities.

Table II
PARAMETERS FOR EXAMPLE

$N = 10$	$P = 6$	$K = 3$	$U = 5$
$G_1 = \{I_1^1, I_1^2, I_1^3\}$	$l_1^1 = 18$	$l_1^2 = 15$	$l_1^3 = 23$
$G_2 = \{I_2^1, I_2^2, I_2^3\}$	$l_2^1 = 12$	$l_2^2 = 16$	$l_2^3 = 7$
$G_3 = \{I_3^1, I_3^2, I_3^3\}$	$l_3^1 = 24$	$l_3^2 = 7$	$l_3^3 = 2$
$G_4 = \{I_4^1, I_4^2, I_4^3\}$	$l_4^1 = 3$	$l_4^2 = 8$	$l_4^3 = 29$
$G_5 = \{I_5^1, I_5^2, I_5^3\}$	$l_5^1 = 2$	$l_5^2 = 34$	$l_5^3 = 9$
$G_6 = \{I_6^1, I_6^2, I_6^3\}$	$l_6^1 = 10$	$l_6^2 = 10$	$l_6^3 = 10$

Fig. 3 is the resulting schedule of the broadcast program by SDAA. From the figure, we can see there exists one empty channel, because we want to keep data broadcast concurrently such that clients can download data simultaneously. To take advantage of this empty channel, we can split long data segments and assign part of them on this channel.

Theorem 2. *SDAA is a 2-approximation.*

Proof: Let G^* denotes the optimal schedule. Since the quality decomposition process is independent with channel scheduling, each G_i has fixed $\{I_i^j\}$ segments and thus $\max G_i = \max_{1 \le j \le K} I_i^j$ is also fixed. We use G_i^* to denote $\max G_i$. Easy to see, $G^* \ge \max_{1 \le i \le P} G_i^*$. Let $L = \sum_{i=1}^{P} G_i^*$ be the total time unit to broadcast, and only T channel groups are available, each channel should contain L/T average data units. Consequently, there must exist one channel with at least that much unit. $G^* \ge L/T$.

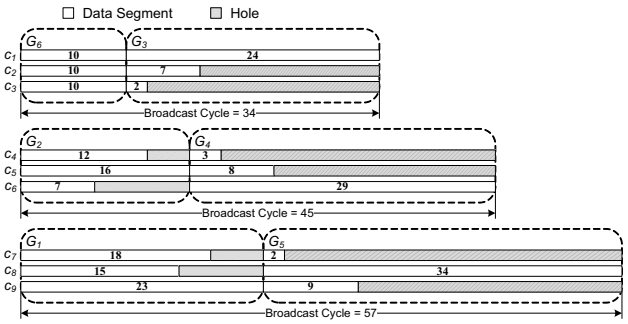


Figure 3. The Result of Algorithm-SDAA 1

Consider the solution of SDAA. Let L_P denote the starting point of G_P as the last datum, and $bcycle$ the output of SDAA. Then $bcycle = L_P + G_P^*$. All other channel groups must contains more than L_P units according to the algorithm. If we split the schedule into two disjoint time intervals by L_P, then the latter interval has length at most G^*. Now consider the former interval, the total amount of time units in this interval is $T \cdot L_P$, which is no more than the total time units. Thus $L_P \le L/T \le G^*$ and $bcycle \le 2G^*$. We thereby get a 2-approximation. ■

SDAA is simple, but may easily bring congestion and inequality. For each datum, whatever quality the client requires, it must download base layer substreams. If the lengths of substreams in each group vary a lot, the broadcasting system will have too many unoccupied slots, and the *tuning time* will be controlled by the longest substream. With the observation that each client can connect with at most C channels at one time, we have the following improvement.

C. Modified Data Allocation Algorithm (MDAA)

In SDAA, clients only use K channels, so we can use other $U - K$ channels to shorten the *tuning time* and *access latency*. The main idea is: for each group G_i, find the maximum length substring I_i^k, split it into two halves, append an index on the head of each half, and replace I_i^k in G_i. Repeat this step until the total channel number is U.

Note that the channel assignment remains the same as SDAA. However, the channels are divided by $T = N/U$. (Divide $\mathcal{C}$ into T groups $\mathcal{CG} = \{CG_1, \cdots, CG_T\}$, where $\forall 1 \le i \le T$, $CG_i = \{c_{(i-1)U+1}, c_{(i-1)U+2}, \cdots, c_{iU}\}$.) If we have a family of data groups, we sort them in nondecreasing order. For each group, we append it to the group of channels with the least data segment length.

Therefore, we only need to deal with single group of data items. If we modify each group of data items, then the rest of the process is the same as SDAA. Thus, the input is a group of data G_i, then the output is $G_i{}'$, such that in the next phase we can easily order all $G_i{}'$ and assign to the channel groups by SDAA. Alg.2-MDAA shows the process.

Algorithm 2 Modified Data Allocation Algo.-MDAA

Input: Data group $G_i = \{I_i^1, \cdots, I_i^K\}$ and constraint U.
Output: Modified data group G_i', which contains U items.

1: Set $h = K$
2: **while** $h \leq U - 1$ **do**
3: find $I_i^k \in G_i$ with $\max\{l_i^k\}$;
4: split I_i^k into $I_i^{k_1}$ and $I_i^{k_2}$, append an index to each head, and replace I_i^k with $I_i^{k_1}$, $I_i^{k_2}$; h+=1;
5: **end while**
6: Reorder G_i according to qualities, and output G_i.

Theorem 3. *For each datum G_i, MDAA takes $O(U^2)$ time.*

Proof: The propose of MDAA is to fill $U - K$ empty channels for each G_i. For each empty channel, we select a data segment with maximum length and split it into two pieces. Since finding a maximum one among K items needs $O(K)$ time, the total time is $\sum_{a=0}^{U-K}(K + a) = O(U^2)$. ■

Totally, SDAA+MDAA requires $O(PU^2 + P \ln P)$ time.

Using SDAA+MDAA, we can get a program with data in Table II shown in Fig. 4. Since we split a data segment into two, clients cannot recognize this data by their primary keys. We need to insert an additional key as a head to each half. Compared to the result in Fig. 3, bcycle reduces significantly.

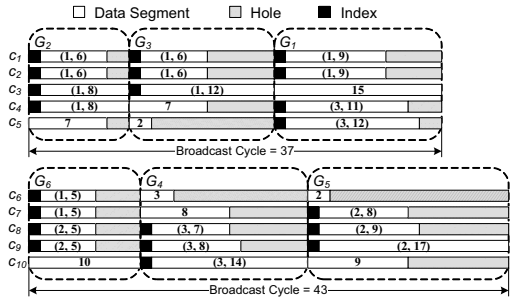


Figure 4. The Result of Algorithm 2-MDAA

However, MDAA is still not good enough. E.g., if we apply MDAA to a datum I_i^k with $l_i^k = 24$ on 2 empty channels, we can get 3 segments with length$\max\{13, 7, 7\}$, but if we divide I_i^k into three equal segments, the optimal length is $\max\{9, 9, 9\}$, which saves 30.7% of the bcycle. Hence, we make a further optimization as follows.

D. Average Estimation Algorithm (AEA)

AEA can be divided into three consecutive phases. Phase 1 recursively estimates the average data length (defined as A_w). In each iteration, it computes A_w of the remaining data items, allocates datum whose length $< A_w$ to available channels, and remove it from unassigned list. Obviously, A_w keeps growing. When Phase 1 terminates, the l_i^k of each unallocated I_i^k is longer than the final estimated A_w.

Then for the each remaining I_i^k, the second phase assigns $\lfloor l_i^k / \lceil A_w \rceil \rfloor$ channels to allocate I_i^k with length $\lceil A_w \rceil$. After this phase, each remaining I_i^k has length $0 \leq l_i^{k'} < \lceil A_w \rceil$.

If there exists $l_i^{k'} > 0$, the third phase allocates the remaining data according to two rules: 1) if available channel exists, allocate data with $\max\{l_i^{k'}/n_i^k\}$ (n_i^k: the number of channels assigned to I_i^k); and 2) if no available channel exists, distributed $l_i^{k'}$ equally to $n_i^{k'}$.

Alg. 3-AEA describes the details. Similar as MDAA, we only deal with data segments in one group.

Algorithm 3 Average Estimation Algorithm-AEA

Input: Data group $G_i = \{I_i^1, \cdots, I_i^K\}$ and constraint U.
Output: Modified group of data items G_i'

1: Set $w = 1$, $h = U$, $A_0 = 0$. /∗ Phase 1 ∗/
2: **while** $A_{w-1} \neq A_w$ **do**
3: **for** $k = 1$ to K **do**
4: If $(0 < l_i^k \leq A_w)$ set $l_i^k = 0$, $n_i^k = 1$, $h- = 1$
5: **end for**
6: $w+ = 1$, $A_w = (\sum_{k=1}^{K} l_i^k)/h$
7: **end while**
8: **for** $k = 1$ to K **do** /∗ Phase 2 ∗/
9: If $(l_i^k \neq 0)$ set $n_i^k = \lfloor \frac{l_i^k}{\lceil A_w \rceil} \rfloor$, $l_i^k - = n_i^k \times \lceil A_w \rceil$, $h- = n_i^k$
10: **end for**
11: **while** $h > 0$ **do** /∗ Phase 3 ∗/
12: Find I_i^a with $\max_{1 \leq k \leq K} \{\frac{l_i^k}{n_i^k}\}$, $l_i^a = 0$, $n_i^a + = 1$, $h- = 1$
13: **end while**
14: Append remaining I_i^k with $\frac{l_i^k}{n_i^k}$ to channels assigned in Phase 1 and 2. Reorder G_i with q_k and output G_i.

Theorem 4. *For each G_i, AEA takes $O(K \ln K)$ time.*

Proof: In each iteration, we calculate the average length A_w and remove the data items with length smaller than it, which takes $O(\ln K)$. There are at most K iterations, and thus needs $O(K \ln K)$. In phase 2, we order left segments and insert them to empty channels, which takes $O(K \ln K)$. In phase 3, we deal with the last remaining segments, with at most $O(K)$. Totally, Alg. 3 takes $O(K \ln K)$. ■

Thus, AEA needs $O(PK \ln K + P \ln P)$ time totally. Using AEA, we get a schedule for Table II. The result can be seen in Fig. 5. From this figure we can see that AEA improves the system performance a lot. The unused holes in the system reduces, and the length of cycles shrinks much.

E. Channel Overlapping Algorithm (COA)

AEA gets an optimal solution when each channel only accepts one quality. To evaluate the system performance, we further release this constraint as an ideal case to compute

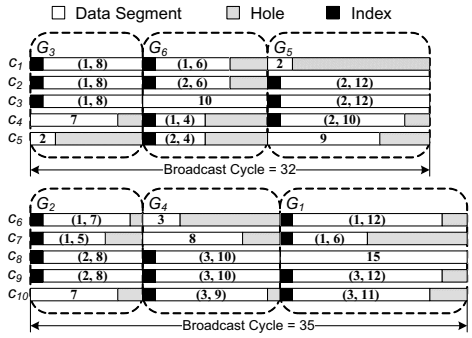


Figure 5. The Result of Algorithm 3-AEA

a lower bound for local optimization and propose *Channel Overlapping Algorithm* (COA). COA guarantees that the length of the data group G_i is almost the same as $A = \lceil (\sum_{k=1}^{K} l_i^k)/U \rceil$. A is acting as a threshold to control the data length of each G_i. Note that by interleaving data streams with different qualities, we loose the opportunity for clients to connect only part of the channel groups to download the required data with different quality requirements. Thus, COA is used for theoretical analysis.

COA also has three consecutive phases. Phase 1 assigns $\lfloor l_i^k/A \rfloor$ channels for each I_i^k with length A. Phase 2 allocates the remaining data segments into available channels: firstly allocate the data with longest remaining length and then append data segment to the same channel if total length of this data segment plus index is less than $A+1$. We repeat this step until all available channels are used. Phase 3 separates and attaches the remaining data to the channels with the total length less than the remaining spaces. However, since we insert redundant indices, it is possible that we cannot keep all data within length $A+1$. Here we loose the threshold and repeat the algorithm until we find a feasible solution. Alg. 4-COA shows the details.

Theorem 5. *For each G_i, COA takes $O(K^2 \ln K)$ time.*

Proof: COA is very similar as AEA. However, when we insert indices into the system, the threshold A may not be enough. Thus we should increase A and recalculate the schedule. We repeat this process at most K time, since each item will be separated at most U times, bringing at most UK indices. And every time when A increases, we have additional U spaces. Therefore, COA terminates after at most K iterations. Combining the proof of Theorem 4, the complexity is $O(K^2 \ln K)$. ■

Using COA, we get a schedule for Table II shown in Fig. 6, proving that COA outputs the smallest cycle length.

V. Simulation

We analyze the correctness and performance of four algorithms by numerical experiments. According to the real system constraints, we set $N \leq 100$, $U \leq 20$, and $K \leq 5$.

Algorithm 4 Channel Overlapping Algorithm-COA

Input: Data group $G_i = \{I_i^1, \cdots, I_i^K\}$ and constraint U.
Output: Modified group of data items G_i'

set $A = \left\lceil \left(\sum_{k=1}^{K} l_i^k \right)/U \right\rceil$. $f_{\{1 \text{ to } K\}} = 0$.
for $k = 1$ to K **do** /∗ Phase 1 ∗/
  If $(l_i^k \leq A)$, $f_k = 1$, else $f_k = 0$.
  $n_i^k = \lfloor l_i^k/A \rfloor$, $l_i^k = l_i^k - n_i^k \times A$
end for
$h = U - \sum_{k=0}^{K} n_i^k$
Sort I_i^k by l_i^k decreasingly; /∗ Phase 2 ∗/
set $d_{\{j=1 \text{ to } h\}} = A + 1$, $X_{\{j=1 \text{ to } h\}} = \emptyset$
for $k = 1$ to K **do**
  for $j = i$ to h **do**
    If $(l_i^k + 1 - f_k \leq d_j)$, set
    $d_j - = l_i^k + 1 - f_k$, $X_j \cup = \{(k, l_i^k)\}$, $l_i^k = 0$
  end for
end for
Set $p =$ no. of channels with $d_j > 1$. /∗ Phase 3 ∗/
if $l_i^k + p \leq \sum_{j=1}^{h} d_j$ **then** split l_i^k with d_j to fill channel
else set $A+ = 1$, goto Phase 1.
end if
Reorder channels increasingly by q_k, and output G_i.

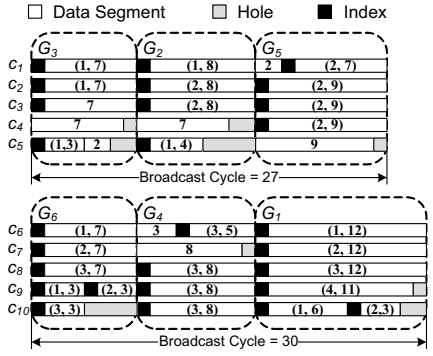


Figure 6. The Result of Algorithm 4-COA

After data allocation, we form a program with P items. Clearly, smaller $bcycle$ length shortens *access latency*.

We evaluate the performance of our system by two parameters: *average number of files on each channel* (AFC) and *average channel wastage* (ACW). AFC shows the expectation of waiting time, which is considered as the energy consumption for mobile users to access their requested multimedia data if users uniformly access to the system. ACW, measures the energy wastage at the BS. Our evaluation environment is a broadcasting system where multimedia files are broadcasted via multi-channels. In each broadcast program, the server puts required data files onto data broadcast cycle based on proposed algorithms. We assume that the broadcasting program is static during a period of time. The BS has N channels to broadcast P files simultaneously while the end users, equipped with MIMO

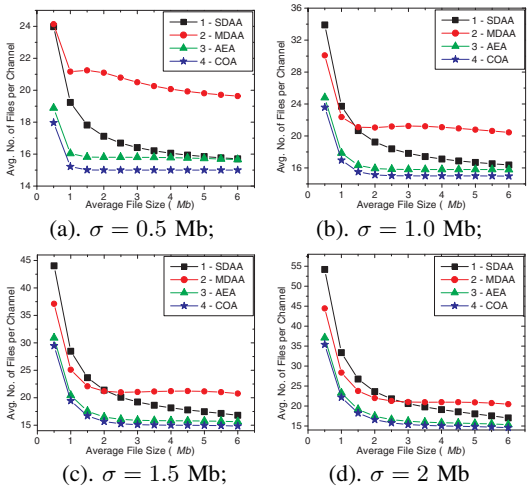
antennas, can access at most U channels at one time.

For all evaluations, we generate multimedia files based on normal distribution with certain μ and σ, where μ gives the average size for multimedia files and σ shows the variance among multimedia files. Note that if the file size is not positive, we drop it and regenerate one. The size of files range from 128 KB to 8 MB, which is the general size of multimedia files such as photo, music, and small video.

We use uniform distribution to split each file into K parts, where K is the number of qualities in the encoding system. We hope show the performance of multimedia data broadcast system under different parameters of the system (i.e. the total broadcast channel N, the number of broadcasting files P, the file size μ and σ, the access channels U, and the qualities K). Consequently, we test AFC and ACW under different parameters using four algorithms. We run the simulation 10,000 times and take the average value. Moreover, for each program, we calculate the average data length on each channel. Set L_P as the average channel length for 10,000 simulations, and L_U the average unused channel length, then we get that AFC=L_P/μ, and ACW=L_U/L_P.

A. Verification of μ and σ

We analyze the relationship between data size and system efficiency, since in multimedia databases the size of files varies a lot, influencing the efficiency of broadcasting system greatly. To evaluate this relationship, we test our system under different values of μ and σ, and then produce the result of AFC and ACW. To make the result comparable, set $N = 40$, $U = 10$, $P = 600$, $K = 2$ for each scenario.



(a). $\sigma = 0.5$ Mb; (b). $\sigma = 1.0$ Mb;

(c). $\sigma = 1.5$ Mb; (d). $\sigma = 2$ Mb

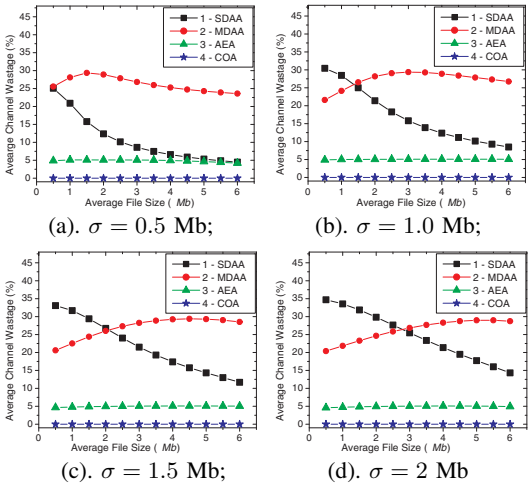
Figure 7. Average Packet Length vs Average Number of Files per Channel

Fig. 7 shows the relation between μ and AFC: x-coordinate is the value of μ, and y-coordinate is the variation of AFC for different algorithms. We can see that:

a) COA obtains the shortest AFC, since it uses all empty "holes" to shorten data length on each channel. This conclusion remains for other comparisons.

b) When μ increases, AFC firstly decreases sharply and then becomes flat. Because when μ is small, data sizes vary a lot with σ, bringing big AFC; when μ increases, it becomes the dominating parameter. Compared with μ, the variation of σ doesn't influence data size that much. Thus AFC decreases a lot. However, when μ continually increases, data sizes only vary a little, so AFC gradually decreases to a constant.

c) When μ is small, MDAA has smaller AFC than SDAA. When μ increases, MDAA gets even worse results than SDAA because MDAA takes more channels and splits data files into smaller segments, while the total length of one data group remains the same. Thus MDAA has larger AFC.



(a). $\sigma = 0.5$ Mb; (b). $\sigma = 1.0$ Mb;

(c). $\sigma = 1.5$ Mb; (d). $\sigma = 2$ Mb

Figure 8. Average Packet Length versus Average Channel Wastage Rate

Fig. 8 shows the relation between μ and ACW:

a) AEA gets reasonable and flat channel usage rate, which means AEA is suitable for many situations. Whatever the input data are, AEA tries best to set the average length for every channel, which is similar to COA. Since we have quality constraints, AEA is 5% higher than COA.

b) When μ is small, MDAA has smaller ACW than SDAA. when μ increases, MDAA becomes worse. Two lines intersect each other in Fig. 8(a-d) at the same μ as Fig.7.

In all, even when σ is big (means data files in broadcasting system vary lot), AEA and COA can schedule data segments very well, with small number of files per channel and little channel wastage. They are stable and reliable.

B. Verification of U and K

Table III is the outputs of AFC and ACW under four algorithms with different U and K values. Then we have:

a) SDAA computes almost the same AFC and ACW, since it has no relation with U; it only deals with different K.

b) The parameter $\frac{U}{K}$ is more important to judge the efficiency of four algorithms. When $\frac{U}{K} < 2$, SDAA has better AFC and ACW than MDAA and AEA. Because the system does not have enough empty channel for MDAA and

AEA to shorten the average channel length for each files. Thus when $\frac{U}{K} \geq 2$, AEA has the best AFC and ACW values.

c) For MDAA, when $\frac{U}{K} = 2^t$, AFC and ACW reduce heavily than other results. Because MDAA is a divide-and-conquer algorithm, which works best when $U = 2^t K$.

In all, different U and K do not influence the system efficiency very much. Moreover, since our simulation proves the stability of AEA and COA, they can be used appropriately for data allocation and data division in the system.

Table III
COMPARISON BETWEEN U AND K

U	K	AFC				ACW (%)			
		SDAA	MDAA	AEA	COA	SDAA	MDAA	AEA	COA
5	2	17.12	21.10	16.84	15.00	12.36	28.90	10.92	0
8	2	17.12	17.12	16.02	15.00	12.36	12.36	6.382	0
10	2	17.12	21.10	15.81	15.00	12.36	28.90	5.087	0
20	2	17.12	21.10	15.39	15.00	12.36	28.90	2.523	0
5	3	18.64	20.19	20.17	15.00	17.46	25.69	25.63	0
8	3	18.64	20.01	17.25	15.00	17.47	25.05	13.05	0
10	3	18.64	20.18	16.70	15.00	17.47	25.68	10.18	0
20	3	18.64	20.18	15.80	15.00	17.46	25.68	5.044	0
5	4	18.86	20.14	20.14	15.00	20.47	25.54	25.54	0
8	4	18.86	18.80	18.28	15.00	20.47	20.22	17.94	0
10	4	18.86	20.14	17.74	15.00	20.46	25.51	15.43	0
20	4	18.86	20.14	16.23	15.00	20.47	25.50	7.567	0

C. Verification of N and P

Table IV is the outputs of AFC and ACW under four algorithms with different N. When N increases, the outputs of AFC and ACW decreases almost linearly. Because when N is bigger, more data files can be broadcasted simultaneously, bringing smaller AFC. ACW remains fixed, since we do not change data generation function and data allocation strategy. Thus, N has no effect on ACW. On the contrary, when P increase, AFC will increase linearly since we have more files to broadcast in one program. With the same reason as we discussed for N, ACW will not change since P also does not affect ACW. Thus, we do not show the results for P.

Table IV
SYSTEM PERFORMANCE UNDER DIFFERENT N

N	AFC				ACW (%)			
	SDAA	MDAA	AEA	COA	SDAA	MDAA	AEA	COA
20	37.72	40.27	35.47	30.00	20.47	25.51	15.42	1.83E-5
30	26.94	26.85	23.65	20.00	20.47	25.51	15.42	1.87E-5
40	18.86	20.14	17.74	15.00	20.47	25.51	15.42	1.83E-5
50	15.72	16.11	14.19	12.00	20.46	25.51	15.42	1.87E-5
60	12.57	13.42	11.82	10.00	20.46	25.50	15.42	1.75E-5
70	11.09	11.51	10.14	8.572	20.46	25.51	15.42	1.75E-5
80	9.431	10.07	8.868	7.500	20.47	25.51	15.42	1.67E-5
90	8.572	8.949	7.883	6.667	20.46	25.50	15.43	1.87E-5
100	7.544	8.055	7.095	6.000	20.46	25.50	15.43	1.67E-5

VI. CONCLUSION

This paper studies data broadcast problem for multimedia database management systems under multichannel wireless environment. Our target is to design data broadcast schemes to reduce the tuning time and access latency, so that the whole system is more efficient. The key point is using *hierarchical encoding* for multimedia data and *MIMO antenna* technology to connect several channels simultaneously. We propose four algorithms (SDAA, MDAA, AEA, COA) to schedule data segments onto channels. SDAA is a sample greedy ordering scheme, which is proved to be a 2-approximation, while MDAA, AEA and COA are improvement of local arrangement. We also provide simulations to illustrate the influence of different parameters to our system.

REFERENCES

[1] S.Anticaglia, F.Barsi, A.Bertossi, L.Iamele, and M.Pinotti, Efficient Heuristics for Data Broadcasting on Multiple Channels, *Wireless Networks*, 14:219-231, 2008.

[2] D.Aksoy, M.Altinel, R.Bose, et.al., Research in Data Broadcast and Dissemination, *AMCP 1998*, 194-207.

[3] Y.Cai, and J.Zhou, An Overlay Subscription Network for Live Internet TV Broadcast, *TKDE*, 18(12):1711-1720, 2006.

[4] M.Cao, A.Liu, P2P-based VoD Service Using Multiple Cache Replacement Algorithm, 258-261, *ICICSE 2012*.

[5] S.Cui, et.al., Energy-Efficiency of MIMO and Cooperative MIMO in Networks, *JSAC*, 22(6):1089-1098, 2004.

[6] J.Huang, and M.Chen, Dependent Data Broadcasting for Unordered Queries in Multi Channel Mobile Environment, *TKDE*, 1143-1156, 2004.

[7] T.Imielinski, S.Viswanathan, and B.Badrinath, Data on Air: Organization and Access, *TKDE*, 9(3):353-372, 1997.

[8] C.Lee, and Y.Leu, Efficient Data Broadcast Schemes for Mobile Computing Environments with Data Missing, *Information Sciences*, 172:335-359, 2005.

[9] S.Lee, C.Hwang, and M.Kitsuregawa, Efficient, Energy Conserving Transaction Processing in Wireless Data Broadcast, *TKDE*, 18(9):1225-1238, 2006.

[10] C.Lin, and D.Lee, Adaptive Data Delivery in Wireless Communication Environments, *ICDCS 2000*, 444-456.

[11] H.Kosch, and M.Döller, Multimedia Database Systems: Where are We Now?, Special Talk at *the IASTED DBA-Conference in Innsbruck*, 2005.

[12] A.Osseiran, V.Stankovic, E.Jorswieck, T.Wild, M.Fuchs, and M.Olsson, A MIMO Framework for 4G Systems: WINNER Concept and Results, *(SPAWC 2007)*, 1-5.

[13] S.Pekowsky, and A.Andorfer, Multimedia Data Broadcasting Strategies, *IEEE Communications Magazine*, 138-146, 2001.

[14] C.Wu, Z.Li, X.Qiu, F.Lau, Auction-based P2P VoD streaming: Incentives and optimal scheduling, 8(1), 14:1-14:22, *ACM-TOMCCAP 2012*.

[15] Y Wu, J.Zhao, M.Shao and G.Cao, Stretch-Optimal Scheduling for On-Demand Data Broadcasts, 500-504, *(ICCCN 2001)*.

[16] D.Yang, and M.Chen, On Bandwidth-Efficient Data Brodaccast, *TKDE*, 20(8):1130-1144, 2008.

[17] S.Yi, and S.Nam, Effective Generation of Data Broadcast Schedules with Different Allocation Numbers for Multiple Wireless Channels, *TKDE*, 20(5):668-677, 2008.

[18] T.Yoshihisa, M.Tsukamoto, and S.Nishio, Scheduling Methods for Broadcasting Multiple Continuous Media Data, *ACM-MMDB 2003*.

[19] C.Young, and G.Chiu, Efficient Dissemination of Transaction-Consistent Data in Broadcast Environments, *TKDE*, 19(3):384-397, 2007.

[20] J.Zabala, H.Chamorro, et.al., Comparative Analysis and Tests of Intelligent Streaming Video on Demand for Next Generation Networks: Two Colombian Study Cases, *ICNS 2011*.