# Efficient Data Retrieval Scheduling for Multi-Channel Wireless Data Broadcast

Zaixin Lu*, Yan Shi†, Weili Wu*, and Bin Fu‡
*Department of Computer Science, University of Texas at Dallas, Richardson, Texas
†CSSE Department, University of Wisconsin - Platteville, Platteville, Wisconsin
‡Department of Computer Science, University of Texas - Pan American, Edinburg, Texas

***Abstract*—Wireless data broadcast is an efficient technique of disseminating data simultaneously to a large number of mobile clients. In many information services, the users may query multiple data items at a time. In this paper, we study the data retrieval scheduling problem from the client's point of view. We formally define the Largest Number Data Retrieval (LNDR) problem with the objective of downloading the largest number of requested data items in a given time duration, and the Minimum Cost Data Retrieval (MCDR) problem which aims at downloading a set of data items with the minimum energy consumption. When the time needed for channel switching can be ignored, a Maximum Matching optimal algorithm is exhibited for LNDR which requires only polynomial time; when the switching time cannot be neglected, LNDR is proven to be $\mathcal{NP}-$hard and a greedy algorithm with constant approximation ratio is developed. We also prove that the MCDR problem is $\mathcal{NP}-$hard to be approximated within to any nontrivial factor and a parameterized heuristic is devised to solve MCDR non-optimally.**

***Index Terms*—Wireless data broadcast, Multi-channel, Data retrieval, $\mathcal{NP}-$hard, Approximation and Inapproximation**

## I. Introduction

Wireless data broadcast has been a popular data dissemination method in the mobile computing environment. In a typical wireless data broadcast system, a base station will broadcast information over one or multiple broadcast channels repeatedly. Clients will listen to the channels, wait for the requested data and download them when they arrive. Wireless data broadcast is especially suitable for public information, such as weather, traffic, and stock quote, because of its scalability and flexibility [24].

Two major performance concerns for a wireless data broadcast system are the response time and the energy efficiency. Response time is the time interval between the moment a client tunes in a broadcast system with a request of one or more data items to the moment all requested data are downloaded. It is obvious that shorter response time is more desirable. On the other hand, in wireless communication environments, most clients are mobile devices operating on batteries. The smaller the amount of energy consumed during retrieving data is, the longer the battery life of a mobile device will be. Therefore, saving energy is another important issue for designing wireless data broadcast system.

Various index techniques have been introduced in wireless data broadcast systems to reduce the energy consumption [5], [20], [23], [24]. A mobile device usually works in two modes: the *active mode* and the *doze mode*. The energy consumed in the *active mode* is about 805–1400mW and that in the *doze mode* is about 60mW [14], [24]. With the help of index information, clients can learn the arriving time of their requested data in advance and "sleep" while waiting for the data to arrive, which can greatly reduce the energy consumption.

In recent years, fast development of wireless communication technologies such as OFDM (Orthogonal frequency-division multiplexing) makes efficiently broadcasting data through multiple channels possible [25]. How to allocate the data onto multiple channels to minimize the expected response time has become a hot research topic and lots of scheduling algorithms are proposed [11], [19], [21].

When a query requests only one data item, to schedule the retrieving process is straightforward. However, it is common that a query requests multiple data items at a time [9], [15], [18] (e.g., a user may submit a query of the top 10 stocks). In such cases, different retrieving schedules may result in different response time. Moreover, in a multi-channel broadcast system, retrieving data will probably need switchings among the channels, which not only consumes additional energy, but also causes possible conflicts [17], [22], [26]. Compared to the massive amount of research effort on scheduling data at the server side, there has been little work done on scheduling the data retrieval process from the client's point of view. In this paper, we focus on developing efficient scheduling algorithms for retrieving multiple data items from multiple broadcast channels. We are also the first paper presenting a thorough theoretical analysis of the data retrieval problem for wireless data broadcast. The main contribution of this paper includes:

1) We define the *Largest Number Data Retrieval* (LNDR) problem with the objective of downloading the largest number of requested data items in a given time interval. The motivation is that the users may lose patience if the downloading takes too long, and they may drop the request if the waiting time exceeds certain deadline. LNDR takes the "deadline" into consideration and therefore also describes the time-critical scenarios. When the time needed for channel switching can be ignored, a polynomial time optimal algorithm is developed. For general LNDR,

978-1-4673-0775-8/12/$31.00 ©2012 IEEE 

a Greedy algorithm with provable approximation ratio is exhibited.

2) The *Largest Weighted Data Retrieval* (LWDR) problem, which is a variation of LNDR, is discussed. It is suitable for broadcast systems in which different data items have different levels of importance for clients. How to construct a polynomial time constant factor approximation for the LWDR problem is presented.

3) When energy consumption is of first priority, we formulate another optimization problem, namely, *Minimum Cost Data Retrieval* (MCDR). We discuss the approximability and inapproximability of this problem under different conditions. Specially, when there is no energy consumption in the doze mode, we derive a polynomial time $O(\log k)$-factor approximation for MCDR and we prove it has no polynomial time $o(\log k)$-factor approximation unless $\mathcal{P} = \mathcal{NP}$, where $k$ is the number of requested data items. When energy consumption in the doze mode is counted, it is shown that MCDR has no polynomial time approximation with any nontrivial factor unless $\mathcal{P} = \mathcal{NP}$.

The rest of this paper is organized as follows: Sec. III gives the problem statements of LNDR and MCDR. Sec. IV studies the LNDR problem. In detail, Sec. IV-A efficiently solves LNDR, assuming there is no channel switching time; Sec. IV-B presents an approximation algorithm for the general case of LNDR and LWDR is discussed in Sec. IV-C. Sec. V analyzes the approximability and inapproximability of MCDR. Sec. VI gives the simulation results of our proposed algorithms; and Sec. VII concludes this paper.

## II. Related Work

Scheduling is a hot topic of wireless data broadcast. Acharya et al. proposed the scheduling problem from the server's point of view [2]. They introduced a data scheduling technique named *Broadcast Disk*. Vaidya et al. studied the single-channel scheduling problem for data of non-uniform lengths [3]. In [6], Kenyon and Schabanel proved that the problem of minimizing the expected *access time* for non-uniform-length data even with single broadcast channel is $\mathcal{NP}-$hard.

In addition to single channel, a lot of research works on data scheduling for multi-channel wireless data broadcast have been done. In [8], Prabhakara et al. presented a multi-level multi-channel model for improving the broadcast performance. Ardizzoni et al. study this problem in [19]. They developed a dynamic programming algorithm to optimally allocate skewed data over multiple channels. Zheng et al. proposed a near-optimal solution for scheduling non-uniform length data on multiple channels [21].

Dependent data broadcast refers to the situations where a query requests multiple data items which have certain relations with each other. There are several research works on data scheduling for dependent data broadcast. Chung et al. and Lee et al. independently discussed the scheduling problem for queries with multiple data items to minimize the average response time [9], [10]. Huang et al. proposed an algorithm for scheduling dependent data for ordered queries over multiple channels in [13], and they extended their work to unordered queries later in [18].

Although there are many works on various data scheduling problems from the server's point of view, there have been little work on data retrieval scheduling at the client side. Hurson et al. studied the data retrieval problem aims at minimizing the number of channel switchings and the response time in [17], [22]. In [26], Shi et al. investigated how to retrieve data from multiple channels by using multiple parallel processes. However, none of those works provide any theoretical analysis on either the data retrieval problem or their proposed algorithms. All of them assumed the data set are partitioned over multiple channels without replications, which restricts their application since in many real situations the same data may appear repeatedly on channels based on their access frequencies [2], [15], [16].

## III. Problem Formulation

Suppose a set of data items $d_1, d_2, \cdots, d_N$ are broadcasted repeatedly from a base station which has $n$ channels $c_1, c_2, \cdots, c_n$. A mobile client wants to download a subset of data items $D = \{d_{D_1}, d_{D_2}, \cdots, d_{D_k}\}$, where $k$ denotes the number of data in $D$. Without loss the generality, we assume: 1) The $n$ broadcast channels have the same bandwidth. 2) The $N$ data items have uniform length. 3) The index information is allocated on separate broadcast channels from data, and clients will retrieve the index information before downloading data. 4) The mobile clients can only access a single channel at any particular time. Note that the second assumption does not limit the use of our model when data items are of different sizes. In that case, we can treat a large data item as multiple consecutive data items of the same size.

According to the above assumptions, we can define a "time slot" as the time needed to broadcast one data item. For the rest of the paper, "time $T$" means the time slot with sequence number $T$ and a triple $Tr = \{i_{Tr}, j_{Tr}, T_{Tr}\}$ denotes the data item $d_{i_{Tr}}$ can be downloaded from channel $c_{j_{Tr}}$ at time $T_{Tr}$.

When designing data retrieval algorithms for multi-channel broadcast systems, we need to pay attention to the possible *conflicts* of retrieving data from parallel channels. $\text{Conflict}_1$: The data items allocated on the same time slot of different channels cannot be download simultaneously. $\text{Conflict}_2$: Two data items allocated on the $i^{th}$ and $(i+1)^{th}$ time slots of different channels respectively cannot be both downloaded during $[t_i, t_{i+1}]$. $\text{Conflict}_1$ is obvious. The reason of $\text{Conflict}_2$ is that switching from any channel to a different channel takes time. If the time is not negligible, a client cannot download data at time $t_{i+1}$ from one channel if it was downloading from another channel at time $t_i$, because a time slot is already the smallest unit for retrieving.

Since we have the assumption that the clients will retrieve the index information before downloading data, the *access* *time* in our study is defined as the time elapsed from the moment a client starts retrieving data to the moment all requested data are downloaded. The energy consumption is defined as $(t_{Access} - t_{Active} - h) \cdot \lambda_{Doze} + t_{Active} \cdot \lambda_{Active} + h \cdot \lambda_{Switch}$, where $t_{Active}$ is the time slots spending on downloading data, and $h$ is the number of channel switchings. $\lambda_{doze}$ and $\lambda_{active}$ are power consumptions during one time slot in the doze and active modes respectively, and $\lambda_{switch}$ is the power consumption for one channel switching. Since the locations of all the requested data are known, the energy cost in the active mode equals to $\lambda_{active}k$, where $k$ is the number of requested data. Therefore from the discussion above, one notes that to reduce $t_{Access}$ is to reduce the energy consumption in the doze mode; and smaller $h$ means less energy consumed in channel switching.

**Definition 1.** *Given a requested data set $D$, a channel set $C$, a* ***Valid Data Retrieval Schedule*** *is a set of $k$ triples $Tr_1, Tr_2, \cdots, Tr_k$, where each triple corresponds to a distinct data item in $D$, and there is no conflicts among all the triples.*

**Definition 2.** *The* ***Energy Cost*** *of a data retrieval schedule is defined as: $\lambda_{Doze}(t_{Access} - k - h) + \lambda_{Active}k + \lambda_{Switch}h$, where $k$ is the number of requested data items and $h$ is the number of channel switchings.*

**Definition 3.** *The* ***Minimum Cost Data Retrieval (MCDR) Problem****: Given a data set $D$, a channel set $C$ and three power consumption parameters $\lambda_{doze}$, $\lambda_{active}$ and $\lambda_{switch}$, find a valid data retrieval schedule such that the energy cost is minimized.*

In some time critical circumstances, clients may prefer a data retrieval schedule to download the largest number of data items in a given time duration regardless of the power consumption. The LNDR problem is defined accordingly.

**Definition 4.** *The* ***Largest Number Data Retrieval****: Given a data set $D$, a channel set $C$, and a time duration $[T_1, T_2]$, download the largest number of data items in $D$ during time $[T_1, T_2]$.*

One important assumption for both defined problems are that there may be replications of the same data items on broadcast channels. Compared with [17], [22], [26] where no data replication is allowed, this is a more generalized assumption which fits all wireless data broadcast programs.

## IV. Largest Number Data Retrieval

In this section, we study the LNDR problem. We show that a special case of this problem can be solved in polynomial time. For the general case, we devise a $\frac{1}{2}$-factor approximation with time complexity $O(nt)$, where $n$ is the number of channels and $t = T_2 - T_1 + 1$ is the number of time slots in $[T_1, T_2]$.

### A. The LNDR Problem without Conflict$_2$

In Sec. III, two types of conflicts are defined. If the time needed to switch between channels is very short and assumed to be negligible, the LNDR problem without Conflict$_2$ can be solved in polynomial time. Similar assumption was also applied to design data allocation methods for generating multi-channel broadcast programs [9], [13], [18].

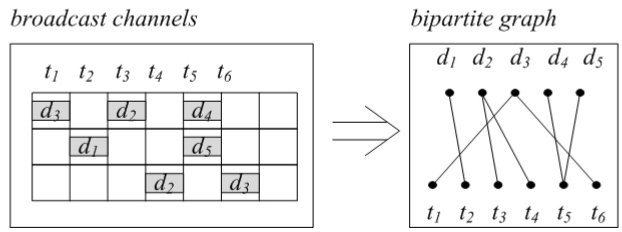


Fig. 1. Convert Largest Number Data Retrieval to Maximum Matching

**Theorem 1.** *There exists a Maximum Matching algorithm to solve LNDR without Conflict$_2$ in $O((k+t)^{\frac{1}{2}}nt)$ time, where $k$ is the number of requested data items, $n$ is the number of broadcast channels and $t$ is the time duration.*

*Proof:* If the time needed for channel switching can be ignored, LNDR can be converted into the Maximum Matching problem through a bipartite graph, thus can be solved in polygonal time. An example is shown in Fig. 1. Given an instance of the LNDR problem $\mathcal{L}$, we construct a bipartite graph $G(V_D, V_T, E)$ as follows: 1) For each data item $d_i$, define a vertex $v_i$ in the vertex set $V_D$. 2) For each time slot $T_j$, define a vertex $u_j$ in the vertex set $V_T$, where $T_1 \le T_j \le T_2$. 3) If there exists a channel broadcasting data item $d_i$ at time $T_j$, create an undirected edge $(v_i, u_j)$ in the edge set $E$.

With this construction, finding a schedule to download the largest number of data items in the time duration $[T_1, T_2]$ equals to finding a Maximum Matching in the bipartite graph $G(V_d, V_t, E)$. The Hopcroft-Karp algorithm [1] finds a Maximum Matching in $O(|V|^{\frac{1}{2}}|E|)$ time, where $V$ is the vertex set and $E$ is the edge set. We have $|V| \le t + k$ and $|E| \le nt$, hence the LNDR without Conflict$_2$ can be solved in $O((k+t)^{\frac{1}{2}}nt)$ time. ■

### B. The General LNDR Problem

Although channel switching time is usually shorter than one time slot, it is not negligible [17]. We assume, in this study, the channel switching takes one time slot, since it is already the smallest unit for retrieving. We next present a $O(nt)$ time $\frac{1}{2}$-factor approximation algorithm based on a greedy strategy for the general LNDR problem. The pseudo-code is shown in Alg. 1.

**Theorem 2.** *The Greedy-LNDR (Alg. 1) is a $\frac{1}{2}$-approximation for LNDR. It has time complexity $O(nt)$, where $t$ is the total number of time slots, and $n$ is the number of channels.*

*Proof:* As shown in Alg. 1, each time we select a channel that downloads the largest number of consecutive data items in the next time period. Assume that *Opt*

**Algorithm 1** Greedy-LNDR

Input: a time duration $[T_1, T_2]$ and a set of channels with requested data items;
Output: a data retrieval schedule $S$;
Let $P \leftarrow \emptyset$ ($P$ holds the data items already downloaded.);
let $t = T_1$;
**while** $t < T_2$ **do**
  **if** There exist data items not in $P$ and broadcast at time $t$ **then**
    Find a channel that contains the largest $(h \leq T_2 + 1 - t)$ number of consecutive data items not in $P$ and its first data item starts at time $t$;
    Put the $h$ data items into $P$ and update $S$.
    $t \leftarrow t + h + 1$(one time slot is used for switching);
  **else**
    $t \leftarrow t + 1$;
  **end if**
**end while**

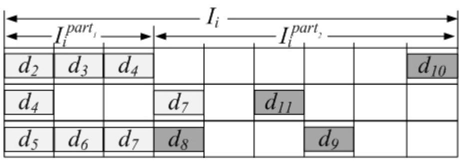


Fig. 2. An illustration of Greedy-LNDR (Alg. 1)

is an optimal schedule that can download the largest number of data items during $[T_1, T_2]$. We compare the schedule $GL$ resulted by Alg. 1 with $Opt$. Assume the solution $GL$ downloads $m$ sets of consecutive data items $D_1, D_2, \cdots, D_m$. Obviously, $D_1, D_2, \cdots, D_m$ are disjoint and we can partition the entire period $[T_1, T_2]$ into $m$ continuous intervals $I_1, I_2, \cdots, I_m$, each interval $I_i$ starting with the first data item in $D_i$. As shown in Fig. 2, each interval $I_i$ can be further divided into two subparts: $I_i^{part_1}$ and $I_i^{part_2}$. $I_i^{part_1}$ contains the time slots to download data items $d_{i_1}, \cdots, d_{i_h}$, and $I_i^{part_2}$ contains the idle time slots (i.e., from the end of the $i^{th}$ downloading to the beginning of the $(i+1)^{th}$ downloading). Note that in $I_i^{part_2}$, data items not downloaded yet in $D_1 \cup D_2 \cup \cdots \cup D_{i-1}$ can only appear immediately after the time slot of $d_{i_h}$ on a different channel, otherwise it will be the beginning of a new interval.

For each interval $I_i = [T_{i_b}, T_{i_e}]$, where $T_{i_b}$ and $T_{i_e}$, respectively, are the beginning and ending time of interval $I_i$, we define $GL^*(I_i) = GL([T_1, T_{i_b} - 1]) \cap Opt([T_{i_b}, T_{i_e}])$, where $Opt([T_{i_b}, T_{i_e}])$ denotes the set of data items downloaded during $[T_{i_b}, T_{i_e}]$ by schedule $Opt$ and $GL([T_1, T_{i_b} - 1])$ represents the set of data items downloaded between $[T_1, T_{i_b} - 1]$ by schedule $GL$. We claim that $|GL^*(I_i) \cup GL(I_i)| \geq |Opt(I_i)|$. Assume $GL$ download $h$ data items during $I_i$, there are only two possible cases (as demonstrated in Fig. 2):

**Case-**1: If the optimal schedule $Opt$ downloads a set of $h$ data items in $I_i^{part_1}$ and $Opt(I_i^{part_1}) \cap GL([T_1, T_{i_b} - 1]) = \emptyset$, then $Opt(I_i^{part_2}) \subseteq GL([T_1, T_{i_b} - 1])$. In the example, assume Alg. 1 downloads $\{d_2, d_3, d_4\}$, and $Opt$ downloads $\{d_5, d_6, d_7\}$, then $d_8$ must belong to $GL([T_1, T_{i_b} - 1])$. Otherwise, Alg. 1 will download $\{d_5, d_6, d_7, d_8\}$ instead of $\{d_5, d_6, d_7\}$. For a similar reason, all the shaded data items $\{d_9, d_{10}, d_{11}\} \subset GL([T_1, T_{i_b} - 1])$.

**Case-**2: If $Opt$ downloads less than $h$ data items in $I_i^{part_1}$ or downloads at least one data in $GL([T_1, T_{i_b} - 1])$, then only the first positions of $I_i^{part_2}$ may contain data items not in $GL([T_1, T_{i_b} - 1])$ and $Opt$ can download at most one of them. That is, $Opt$, either downloads $h + 1$ data items with at least one from $GL([T_1, T_{i_b} - 1])$, or downloads at most $h$ number of data in $I_i$.

In both cases, $|GL^*(I_i) \cup GL(I_i)| \geq |Opt(I_i)|$, thus $\sum_{i=1}^{m} |GL^*(I_i) \cup GL(I_i)| \geq \sum_{i=1}^{m} |Opt(I_i)| \geq |Opt|$ and $\sum_{i=1}^{k} |GL^*(I_i) \cup GL(I_i)| = \sum_{i=1}^{m} |GL^*(I_i)| + \sum_{i=1}^{m} |GL(I_i)| \leq 2 \cdot |GL|$. We now can conduct that $|GL| \geq \frac{|Opt|}{2}$.

In Alg. 1, it takes at most $n|D_i|$ time for downloading the data items in $D_i$. We have $\sum_{i=1}^{m} |D_i| \leq t$, therefore the total time complexity is $O(nt)$. ■

In [22], Hurson et al. study a similar problem as LNDR. The difference is that they assume the data items are residing on channels without replication and the channels have the same broadcast cycle. A heuristic was proposed in that paper to schedule the data retrieving in one broadcast cycle. They generate all possible access patterns from all channels and selects the best one. The method of exhaustive search guarantees an optimal solution, but makes the computation very high and not feasible in practice.

If we restrict the LNDR problem with conditions that 1) all the channels have the same cycle lengths and 2) the data in one cycle are non-replicative, this special LNDR problem equals to the problem studied in [22]. We call a broadcast program satisfying 1) and 2) Flat Data Scheduling (FDS). From the proof of Thm. 2, one notes that if the broadcast is FDS and the time duration is at most one broadcast cycle, Alg. 1 guarantees an optimal solution. In the proof of Thm. 2, we have $\sum_{i=1}^{k} |GL^*(I_i) \cup GL(I_i)| \geq \sum_{i=1}^{k} |Opt(I_i)|$, and a FDS program has no replicative data in one cycle. Therefore, $GL([T_1, T_{i_b} - 1]) \cap Opt([T_{i_b}, T_{i_e}]) = \emptyset$ and $\sum_{i=1}^{k} |GL(I_i)| = \sum_{i=1}^{k} |Opt(I_i)|$.

One important issue, however, is that there is no polynomial time optimal solution for general LNDR unless $\mathcal{P} = \mathcal{NP}$. To show the decision version of general LNDR problem is $\mathcal{NP}-$hard, we reduce in polynomial time the 3SAT problem to it. The 3SAT problem is that given a 3CNF (Conjunctive Normal Form) formula $\mathcal{F}$, determine whether $\mathcal{F}$ has a satisfiable assignment.

**Theorem 3.** *The general LNDR problem is $\mathcal{NP}-$hard.*

*Proof:* Let $\mathcal{F}$ be a 3CNF formula with $m$ variables $x_1, x_2, \cdots, x_m$ and $l$ clauses $f_1, f_2, .., f_l$. We construct the instance of LNDR as follows:

1) Build two channels, $channel_X$ and $channel_{\bar{X}}$. Both

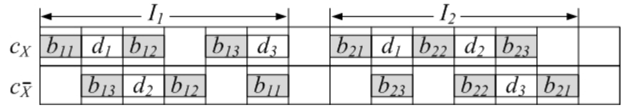


Fig. 3. $3SAT <_m^p LNDR$

of them have broadcast cycle length $2ml + m$ and are partitioned into $m$ equal disjoint intervals $I_1, I_2, \cdots, I_m$, such that each interval contains $2l$ time slots. Two intervals $I_i$ and $I_{i+1}$ are separated by one time slot, where $0 \leq i < m$.

2) For each clause $f_j$, when $f_j$ contains $x_i$, define a data item $d_j$ and allocate it on the $(2j)^{th}$ position of the $i^{th}$ interval of $channel_X$; it is allocated on the $(2j-1)^{th}$ position of the $i^{th}$ interval of $channel_{\bar{X}}$ when $f_j$ contains $\bar{x_i}$.

3) Define a set of data items $D_{b_i} = \{b_{i_1}, b_{i_2}, \cdots, b_{i_l}\}$ for each variable $x_i$. Allocate all the elements in $D_{b_i}$ on the even positions of the $i^{th}$ interval of $channel_X$ in this order; and allocate them on the odd positions of the $i^{th}$ interval of $channel_{\bar{X}}$ in the reverse of this order.

4) Let $D_d = \bigcup_{j=1}^{l}\{d_j\}$ and $D_b = \bigcup_{j'=1}^{m} D_{b_{j'}}$. The requested data set $D = D_b \cup D_d$.

We use a 3CNF $\mathcal{F} = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_2) \wedge (x_1 \vee \bar{x_2})$ to demonstrate the reduction. The data items are arranged on $channel_X$ and $channel_{\bar{X}}$ as the way shown in Fig. 3.

We have to show $\mathcal{F}$ is satisfiable if and only if there is a valid data retrieval schedule $S$ between time duration $[1, 2ml + m]$ to retrieve all the data items in $D$.

1) If $\mathcal{F}$ is a "yes" instance, we can switch among the two channels according to the instance so that all the data items can be received in one cycle, which is $2lm + m$ time slots.

2) If all the data items in $D$ can be downloaded in $2ml + m$ time slots. That is, all the data items in $D_b$ and $D_d$ in can be downloaded in $m$ intervals. There are $m$ disjoint sets in $D_b$, the data items in $D_{b_i}$ are allocated on the $i^{th}$ interval of two channels alternatively and reversely. Obviously, we can download at most $l$ data items in $D_b$ within one interval. In addition, if there is any switching between $channel_X$ and $channel_{\bar{X}}$ in the $i^{th}$ interval, we lose the opportunity to download all the data in $D_{b_i}$, i.e., if $S$ downloads all data items in $D_d$ and $D_b$ in $2ml + m$ time slots, it chooses either $channel_X$ or $channel_{\bar{X}}$ to download data in the $i^{th}$ interval for any given $i$. Therefore, each variable can be assigned with either *true* or *false* to satisfy $\mathcal{F}$.

In sum, we prove the general LNDR problem is $\mathcal{NP}-$hard. ■

### C. A variant of the LNDR problem

Sometimes the importance of data items even in one request are not the same. If there is a deadline, then the client would prefer to download the important items first. To handle this kind of situation, we define a variant of LNDR by adding weights to data items.

**Definition 5.** *The* ***Largest Weighted Data Retrieval (LWDR) problem****: A set of $k$ data items are broadcasted via $n$ channels. Each data item $d_i$ has a weight $w_i$, $(1 \leq i \leq k)$. Given a time duration $[T_1, T_2]$, LWDR is to download a set of data items $D'$, such that $\sum_{d_i \in D'} w_i$ is maximized.*

In the proof of Thm. 1, we convert the LNDR problem into a bipartite graph. The lower set of vertices represents the time slots, and the upper set of vertices represents the requested data. An edge $(u, v)$ is added if at time slot $u$, the data item $v$ is downloadable. For the LWDR problem, we just add a weight to each edge $(u, v)$, which is identical to the weight of data item $v$. It is well known that the weighted maximal matching in bipartite graph can be solved by Hungarian algorithm in polynomial time. Therefore, the LWDR problem without conflict$_2$ can be solved efficiently in polynomial time. We next show that the general LWDR problem with conflict$_2$ has polynomial time $\frac{1}{2}$-factor approximation. Alg. 2 provides the pseudo-code.

**Algorithm 2** Weighted Maximum Matching

```
Input: a time duration [T_1, T_2] and a set of channels
    with weighted data items;
Output: a data retrieval schedule S;
Construct a weighted bipartite graph G(V_d, V_t, E);
Find a Maximum Matching M in G by Hungarian
    algorithm;
Partition M into two subparts M_1 and M_2 (M_1 con-
    tains data with odd time slots and M_2 contains data
    with even time slots); Let M_1^* denote the set of data
    in M_1 which has no conflict with M_2 and M_2^* denote
    the set of data in M_2 which has no conflict with M_1;
if sum_{d_i in M_1} w_i >= sum_{d_j in M_2} w_j then
    S <- M_1 ∪ M_2^*;
else
    S <- M_2 ∪ M_1^*;
end if
```

**Theorem 4.** *Alg. 2 is a polynomial time $\frac{1}{2}$-approximation for the general LWDR problem.*

*Proof:* Assume $M$ is the maximal weighted matching. Partition $M$ into $M_1$ and $M_2$ such that $M_1$ contains all the edges $(u, v)$ with odd time slots and $M_2$ contains all the edges $(u, v)$ with even time slots. Clearly, one of $M_1$ and $M_2$ has the sum of weights to be at least half of the sum of weights of $M$. Assume that $\sum_{d_i \in M_1} w_i \geq \sum_{d_j \in M_2} w_j$, then $M_1 \geq \frac{1}{2}M$ is a $\frac{1}{2}$-approximation. ■

## V. Minimum Cost Data Retrieval

Consider now the MCDR problem, we first show two negative results, then present a parameterized heuristic.

### A. Inapproximability of MCDR

As discussed in Sec. III, the energy consumption for downloading $k$ data items equals to $\lambda_{Doze}(t_{Access} -$

$t_{Active} - h) + \lambda_{Active} t_{Active} + \lambda_{Switch} h$, where $h$ is the number of channel switchings. Since the index information is assumed to be obtained before data retrieving, we have $t_{Active} = k$, and the energy consumption in the active mode is $\lambda_{Active} k$. In the setting of [14], the energy consumed in the active mode is about 13–24 times of that in the doze mode. When requested data size is fixed, with the increasing of bandwidth, the ratio $\lambda_{Active}/\lambda_{Doze}$ becomes larger and $t_{Access}$ becomes shorter. If the bandwidth is efficiently large, we may assume the energy consumption in the doze mode is negligible and minimizing the energy consumption is equivalent to minimizing the number of switchings. We next show that the problem of minimizing the number of switchings, with arbitrary number of channels, is $\mathcal{NP}-$hard to be approximated within to $o(\log k)$.

**Theorem 5.** *There is no $o(\log k)$-factor polynomial time approximate algorithm to minimize the number of switchings unless $\mathcal{P} = \mathcal{NP}$, where $k$ is the number of requested data items.*

*Proof:* We prove the theorem by a reduction from Set Cover. The inputs of a Set Cover are $m + 1$ sets: $S_1, S_2, \cdots, S_m$ and $S$, where $S_1, S_2, \cdots, S_m$ are subsets of $S$. The target is to find the least number of subsets $S_{i_1}, S_{i_2}, \cdots, S_{i_l}$ such that $S_{i_1} \cup S_{i_2} \cup \cdots \cup S_{i_l} = S$. So, if we assume each element in $S$ is a requested data item, we can set up $m$ channels to broadcast the $m$ subsets. The Set Cover problem directly equals to the minimum switching problem. It is well known that the Set Cover problem has an $O(\log n)$-factor polynomial time approximation and it has no $o(\log n)$-factor polynomial time approximation unless $\mathcal{P} = \mathcal{NP}$ [4]. Therefore, an $o(\log n)$-factor polynomial time approximate algorithm for MCDR with $\lambda_{Doze} = 0$ implies $\mathcal{P} = \mathcal{NP}$. ■

Consider now the general MCDR problem for an arbitrary number of channels. In contrast to the minimum switching problem, introducing the energy consumption in the doze mode makes the problem hard to approximate within to any nontrivial factor.

**Theorem 6.** *For any constant $\mu > 1$, there is no polynomial time opt$^\mu$-approximation for MCDR unless $\mathcal{P} = \mathcal{NP}$, where opt is the minimum energy consumption.*

*Proof:* We prove the theorem by exhibiting a polynomial time reduction from 3 Dominating Matching. For 3 disjoint sets $X$, $Y$ and $Z$, let $J$ be a subset of $X \times Y \times Z$. $M \subseteq J$ is a 3 Dimensional Matching if for any two distinct triples $(x_1, y_1, z_1) \in M$ and $(x_2, y_2, z_2) \in M$, we have $x_1 \neq x_2$, $y_1 \neq y_2$ and $z_1 \neq z_2$. Given a set $J$ and an integer $m$, decide whether there exists a 3 Dimensional Matching with $|M| \geq m$ is $\mathcal{NP}-$hard and the $\mathcal{NP}-$hardness holds even in the special case that $m = |X| = |Y| = |Z|$, namely, 3 Dominating Matching.

Given an instance of 3 Dominating Matching $\mathcal{M}$ with inputs $X$, $Y$, $Z$ and $J$, the corresponding instance of MCDR is built by setting $|J|$ broadcast channels with a requested data set $D = X \cup Y \cup Z$. Each channel broadcast a triple of data in $J$ repeatedly during $[1, 4m-1]$ time, where $m = |X| = |Y| = |Z|$. After that the $|J|$ channels only broadcast unrelated data $\notin (X \cup Y \cup Z)$ during the next $q$ time slots, where $q = \lceil \frac{(\lambda_{Active} 3m + \lambda_{Switch}(m-1))^{\mu+1}}{\lambda_{Doze}} \rceil$. Thus all the channels have cycle length $q + 4m - 1$.

If $\mathcal{M}$ is a "yes" instance, i.e., there are $m$ disjoint triples in $J$. Consider the solution of MCDR which downloads the $m$ triples sequentially. That is, all the data are downloaded in $4m-1$ time slots with $m-1$ number of channel switchings. Indeed, this is an optimal solution. We can download at most 3 data items from one channel, thus to download $3m$ data, we need at least $m - 1$ switchings. We claim that, for any valid schedule, $t_{Access} \geq 4m - 1$. Suppose, for the sake of contradiction, that $t_{Access} < 4m - 1$, we have $h \leq t_{Access} - t_{Active} < m - 1$, since $t_{Active} = 3m$ and $t_{Access} < 4m - 1$. This contradicts the fact that $h \geq m - 1$. Therefore, the minimum energy consumption is $\lambda_{Active} 3m + \lambda_{Switch}(m - 1)$.

Conversely, consider $\mathcal{M}$ is a "no" instance, the requested data cannot be downloaded in $4m - 1$ time slots, thus cannot be downloaded in $4m - 1 + q$ time slots. The energy consumed in the doze mode is at least $(\lambda_{Active} 3m + \lambda_{Switch}(m - 1))^{\mu+1}$.

Assume there exists a polynomial time opt$^\mu$-factor approximation for MCDR and $\mathcal{M}$ is a "yes" instance, the energy consumption is at most $(\lambda_{Active} 3m + \lambda_{Switch}(m - 1))^{\mu+1}$. Therefore, a polynomial time opt$^\mu$-approximation for MCDR implies $\mathcal{P} = \mathcal{NP}$. ■

### B. A Parameterized Heuristic

The simple greedy $O(\log n)$-factor approximation for the Set Cover problem brings an $O(\log n)$-factor approximate algorithm for the minimum switching problem. We next present a parameterized heuristic for MCDR. Let $P$ holds the data items already downloaded, $D[T_1, T_2]$ denotes the set of data in $[T_1, T_2]$, and $D^*[T_1, T_2]$ denotes a set of downloadable data in $[T_1, T_2]$, which is a subset of $D_{c_i}[T_1, T_2] \backslash P$. The energy consumption of downloading $D^*[T_1, T_2]$ distributed to each requested data is $\frac{\lambda_{Switch} n_s + \lambda_{Doze}(T_2 - T_1 - |D^*_{c_i}[T_1, T_2]| - n_s) + \lambda_{Active}|D^*_{c_i}[T_1, T_2]|}{|D^*_{c_i}[T_1, T_2]|}$, where $n_s$ denotes the number of switchings. Each time the algorithm searches for the minimum cost downloadable set $D^*_{c_i}[T_1, T_2]$, and terminates when all the requested data are downloaded. Alg. 3 demonstrates the pseudo-code.

The parameter $W$ reflects the preference between the computational complexity and the algorithm performance. It is intuitive that better solution can be obtained by increasing $W$.

## VI. Simulation

We perform three experiments to evaluate our proposed algorithms. In addition to the proposed algorithms, we also implement two heuristics Next Object (NO) and Row Scan (RS) [17] for comparison purposes. Next Object is a natural data retrieval strategy which always downloads the next

**Algorithm 3** MCDR Heuristic

Input: a requested data set $D$, a channel set $C$, and a parameter $W$.
Let $P \leftarrow \emptyset$;
Let $t \leftarrow 1$; ($t$ is the current time slot.)
**while** $P \neq D$ **do**
  Use the brute-force method to find a downloadable data set $D^*[t, t+w]$ with lowest cost, where $0 < w \leq W$;
  $P \leftarrow P \cup D^*[t, t+w]$;
  $t \leftarrow t + w + 1$;
**end while**

available data in a channel and Row Scan retrieves all the requested data from one channel in each pass. The performance metrics used in the experiments are ADN (Average Download Number) and AEC (Average Energy Consumption), where ADN is defined as the the average number of data items downloaded in the given duration, and AEC is defined as the average energy consumption for downloading a set of requested data. We first simulate a special data broadcast environment, in which the channel switching time is assumed to be negligible. The performances of Greedy-LNDR (GL, Alg. 1) and Maximum Matching (MM, Alg. 2) are compared with NO. In the second experiment, we compares the three algorithms in a general broadcast environment, i.e., the channel switching time is not negligible. The last experiment evaluates the performance of MCDR Heuristic (MH, Alg. 3), NO and RS in terms of Average Energy Consumption (AEC) in a general broadcast environment. The broadcast environments will be introduced in detail in Sec. VI-A.

### A. Simulation Environment

To evaluate our algorithms, we construct two types of broadcast programs: *Special Data Broadcast without channel switching time* (SDB) and *General Data Broadcast with channel switching time* (GDB). In both types of programs, we assume index information is allocated on separate broadcast channels from data, and clients will retrieve the index information before downloading data. We simulate a base station with $n$ broadcast channels, the bandwidth of each channel is 1Mbit/sec [17], [22]. The database to be broadcast has $N$ data items, each of size $512$ bytes. The time duration is denoted by $t$. The data items of query data set $D$ is generated with access probabilities following the Zipf distribution [7], which is a typical model for web data of non-uniform access patterns [20]. Given the number of data items $N$ and the skew parameter $\theta$, the Zipf distribution is: $p_i = \frac{(\frac{1}{i})^\theta}{\sum_{i=1}^{N}(\frac{1}{i})^\theta}$, where $p_i$ is the access probability of data $d_i$ and $0 \leq \theta \leq 1$.

For both SDB and GDB, we adopt two data scheduling methods for data allocation at the server side: Flat Data Scheduling (FDS) and Skew Data Scheduling (SDS) [19].

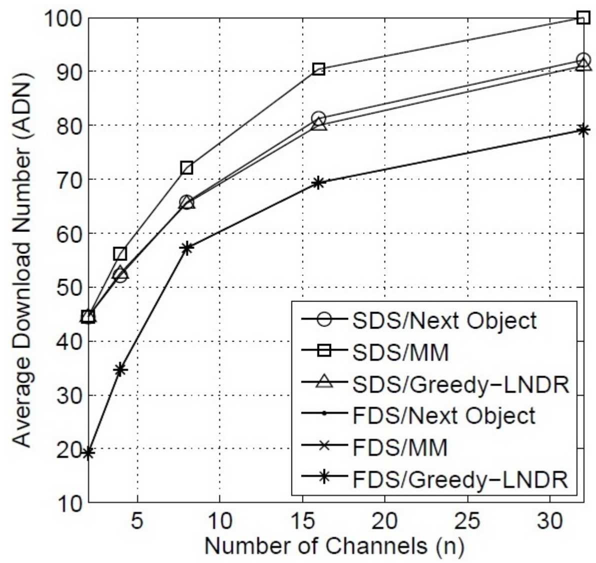


Fig. 4. Experimental Results for Special LNDR without Channel Switching Time when $N = 1000$, $k = 100$, $t = 100$ and $\theta = 0.8$

With FDS, data items are equally distributed over $n$ channels, each of cycle length $N/n$. With SDS, the data items are allocated over $n$ broadcast channels according to their access probabilities, with the objective of minimizing the expected response time. In [19], a dynamic programming algorithm is presented, which provides optimal SDS. The parameters of both programs vary in the range: $N = 1000$, $20 \leq k \leq 200$, $50 \leq t \leq 200$ and $2 \leq n \leq 32$ respectively. For each experiment, we simulate 10000 requests to get ADN and AEC.

### B. Simulation Results

The simulation results of the first experiment are shown in Fig. 4. Since the RS heuristic is proposed for reducing the number of channel switchings [17] and is not suitable for LNDR, its results are omitted henceforth for the first two experiments. According to the results in Fig. 4, when applying FDS for data allocation at the server side and the channel switching time is assumed to be negligible, NO, MM and GL download the same number of data items (i.e., all the three solution are optimal), which validates our conclusion in Sec. IV-A and IV-B. When FDS is replaced by SDS at the server side, the number of data items downloaded by NO (respectively GL) increases more than 15 percent, which implies that SDS outperforms FDS for data allocation at the server side. Moreover, when MM scheduling is used to replace GL or NO at the client side, the ADN increases again 10 percent approximately. From, Fig. 4, we can conduct that significant better performance can be obtained by using a proper combination of server side scheduling and client side scheduling. MM, which guarantees an optimal solution, does perform better than NO and GL in the SDB environment.

Although MM downloads maximum number of data items in the SDB environment, it is not optimal in the

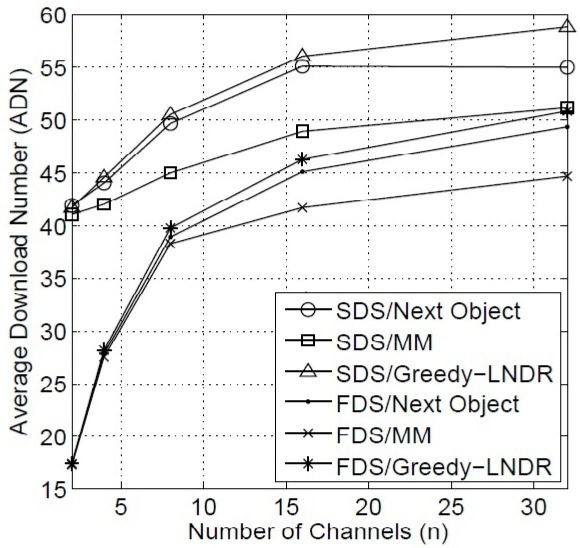


Fig. 5. Experimental Results for General LNDR with Channel Switching Time when $N = 1000$, $k = 100$, $t = 100$ and $\theta = 0.8$

| $k/t$ | Schema | ADN$_{n=4}$ | ADN$_{n=8}$ | ADN$_{n=32}$ |
|---|---|---|---|---|
| 20/50 | FDS/MM | 3.69 | 6.75 | 12.63 |
| | FDS/NO | 3.69 | 6.75 | 13.97 |
| | FDS/GL | 3.70 | 6.77 | 14.05 |
| | SDS/MM | 9.44 | 10.93 | 13.04 |
| | SDS/NO | 9.50 | 11.16 | 15.86 |
| | SDS/GL | 9.55 | 11.47 | 17.22 |
| 100/100 | FDS/MM | 27.47 | 38.25 | 44.59 |
| | FDS/GL | 27.78 | 38.86 | 49.33 |
| | FDS/NO | 28.09 | 39.76 | 50.85 |
| | SDS/MM | 42.02 | 44.92 | 51.07 |
| | SDS/GL | 43.94 | 49.57 | 54.89 |
| | SDS/NO | 44.55 | 50.48 | 58.82 |
| 200/100 | FDS/MM | 40.94 | 48.39 | 50.63 |
| | FDS/NO | 42.33 | 51.11 | 54.42 |
| | FDS/GL | 43.95 | 54.52 | 62.11 |
| | SDS/MM | 56.49 | 54.64 | 51.23 |
| | SDS/NO | 60.18 | 62.33 | 64.33 |
| | SDS/GL | 64.47 | 69.03 | 70.37 |
| 200/200 | FDS/MM | 81.70 | 86.31 | 93.33 |
| | FDS/NO | 84.69 | 96.02 | 106.33 |
| | FDS/GL | 87.55 | 100.16 | 113.80 |
| | SDS/MM | 94.21 | 99.99 | 101.95 |
| | SDS/NO | 102.90 | 114.60 | 110.35 |
| | SDS/GL | 106.48 | 117.26 | 126.69 |

TABLE I
EXPERIMENTAL RESULTS FOR GENERAL LNDR WHEN $N = 1000$ AND $\theta = 0.8$

GDB environment. After obtaining a maximum matching, it has to deal with the possible conflicts because the channel switching takes time. Fig. 5 and Tab. I demonstrate the simulation results of the second experiment. Consider the results shown in Fig. 5, when $n < 15$, the ADN of all schemes increase greatly as $n$ increases. When $n > 15$, the increments of MM slow down. Similar as the results of the first experiment, SDS performs better than FDS when applying the same data retrieval scheduling at the client side. But unlike in the SDB environment, both GL and NO outperform MM with respect to the ADN in the GDB environment. When downloading small number of data from few broadcast channels, as shown in Tab. I, the number of data items downloaded by MM is slightly less than that downloaded by GL or NO. When $k \geq 100$ and $n > 4$, the performance of MM becomes poor. Specifically, when $k = 200$ and $T = 100$, the ADN of MM decreases as $n$ increases. This agrees with our intuition in that increase the number of channels causes more conflicts in the maximum matching. GL always works better than NO and MM in terms of ADN in the GDB environment.

The results of the third experiment are exhibited in Tab II and Fig. 6. It is worthy to mentioned that the power parameters $\lambda_{Active}$ and $\lambda_{Doze}$ highly depend on the channel bandwidth and the data size. In the experiment, $\lambda_{Active}$, $\lambda_{Doze}$ and $\lambda_{Switch}$ are set to 130mW, 6mW and 13mW respectively, as used in [22]. We show the performance of MH with the parameter $W = 10$. From Fig. 6, one notes that using SDS and more broadcast channels can reduce the AEC for clients. From the client's point of view, when using NO scheduling, the energy consumption AEC$_{Doze+Switch}$ is 5.5 and 6.4 percent higher than that resulted by MH (when $n = 32$ and FDS, SDS are used at the server side respectively). RS performs significantly worse than MH and NO when $k = 100$ and $N = 1000$. In addition, several

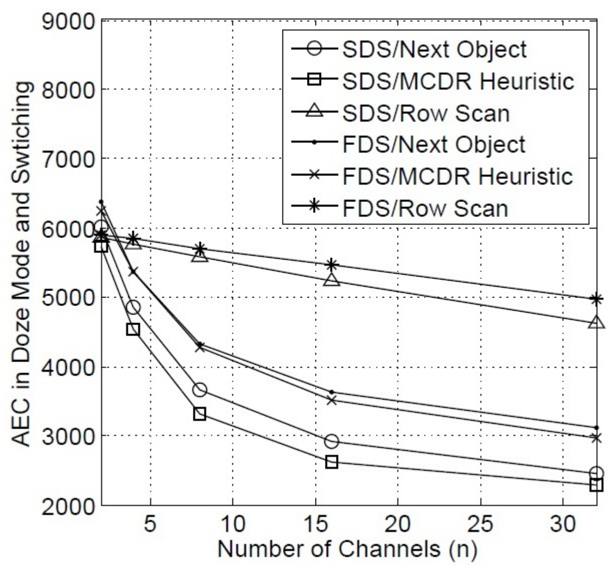


Fig. 6. Experimental Results for MCDR when $N = 1000$, $k = 100$ and $\theta = 0.8$

conclusions can be made from the results of Tab. II. First RS is not suitable for download small number of data, but it consumes the minimum energy with respect to NO and MH when $k = 500$ and $N = 1000$. Second, NO is poor at downloading large number of data in terms of energy consumption. Third, MH outperforms NO and RS when $k \in \{50, 100\}$ and its performance consistently matches that of RS when $k = 500$.

Finally, we would also like to evaluate the running time of the proposed algorithms so that we can validate the scalability. We simulate a GDB environment with relatively large parameters: $N = 10000$, $k = 500$, $\theta = 0.8$, $n = 16$ and $W = 10$. The average running time of

| $k$ | Schema | AEC | AEC$_{Doze+Switch}$ | ADN$_{Switch}$ |
|---|---|---|---|---|
| 50 | FDS/NO | 9326.58 | 2826.58 | 533.94 |
| | FDS/MH | 9067.84 | 2567.84 | 436.08 |
| | FDS/RS | 11741.66 | 5241.66 | 90.89 |
| | SDS/NO | 8920.34 | 2420.34 | 441.61 |
| | SDS/MH | 8528.13 | 2028.13 | 330.45 |
| | SDS/RS | 11428.35 | 4928.35 | 90.59 |
| 100 | FDS/NO | 17329.25 | 4329.25 | 1032.69 |
| | FDS/MH | 17218.65 | 4218.65 | 790.02 |
| | FDS/RS | 18699.11 | 5699.11 | 91.00 |
| | SDS/NO | 16661.50 | 3661.50 | 863.27 |
| | SDS/MH | 16263.20 | 3263.20 | 649.59 |
| | SDS/RS | 18585.84 | 5585.84 | 91.00 |
| 500 | FDS/NO | 73628.35 | 8628.35 | 3264.01 |
| | FDS/MH | 71412.80 | 6412.80 | 1098.98 |
| | FDS/RS | 71080.01 | 6080.01 | 91.00 |
| | SDS/NO | 73299.26 | 8299.26 | 2540.74 |
| | SDS/MH | 71112.65 | 6112.65 | 995.44 |
| | SDS/RS | 71082.79 | 6082.79 | 91.00 |

TABLE II
EXPERIMENTAL RESULTS FOR MCDR WHEN $N = 1000$, $n = 8$ AND $\theta = 0.8$

MH is only 231 milliseconds. Therefore, it scales well when $W$ is small. When $T = 1000$, the running time of GL is less than 25 milliseconds, while that of MM is about 130 milliseconds. GL is very fast because of its low computational complexity.

In conclusion, for the LNDR problem, MM downloads the maximum number of data items in the SDB environment, and GL always achieves a better solution with respect to MM and NO in the GDB environment. Therefore, the choice between them depends on the broadcast environments. When energy consumption is of first priority, MH always outperforms NO and RS when $k$ is small. RS is an efficient method for retrieving a large percentage of data.

## VII. CONCLUSION

In this paper, the data retrieval scheduling over multiple channels is considered. Two optimization problems, MCDR and LNDR, are defined and a series of theoretical results, such as $\mathcal{NP}-$hardness, approximability and inapproximability, are proven. The simulation results show that the proposed approximation algorithms efficiently schedule the data retrieval process of downloading multiple data from multiple channels.

We also demonstrate the advantage of data allocation scheduling at the server side by simulations with various parameters. According to the experimental results, one observes that significantly better performances can be obtained by using a combination of skewed data scheduling at the server side and proper data retrieval scheduling at the client side. Our future work includes extending our work to the parallel data retrieval scheduling problem. In addition, we explore the data retrieval scheduling algorithms by assuming the index information are already obtained by clients in this study. As a direction for further research, one can study the LNDR and MCDR without that assumption.

## ACKNOWLEDGMENT

This research work is supported in part by National Science Foundation of USA under grants CNS 1016320 and CCF 0829993.

## REFERENCES

[1] J.E. Hopcroft, R.M. Karp: An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs. *SICOMP*, 2(4):225-231, 1973.
[2] S. Acharya, R. Alonso, M. Franklin, S. Zdonik: Broadcast Disks: Data Management for Asymmetric Communication Environments. *SIGMOD 1995*, pp.199–210, 1995.
[3] N. Vaidya, S. Hameed: Log Time Algorithms for Scheduling Single and Multiple Channel Data Broadcast. *MOBICOM 1997*, pp. 90–99, 1997.
[4] U. Feige: A Threshold of $\ln n$ for Approximating Set Cover. *JACM*, 45:314–318, 1998.
[5] T. Imielinski, S. Viswanathan, B.R. Badrinath: Data on Air: Organization and Access. *TKDE*, 9:353–372, 1996.
[6] C. Kenyon, N. Schabanel: The Data Broadcast Problem with Non-Uniform Transmission Time. *SODA 1999*, pp. 547–556, 1999.
[7] C.D. Manning, H. Schutze. *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.
[8] K. Prabhakara, K.A. Hua, J. Oh: Multi-Level Multi-Channel Air Cache Designs for Broadcasting in a Mobile Environment. *ICDE 2000*, pp. 167–176, 2000.
[9] Y.D. Chung, M.H. Kim: Effective Data Placement for Wireless Broadcast. *Distrib. Parallel Dat.*, 9(2):133–150, 2001.
[10] G. Lee, M.S. Yeh, S.C. Lo, A. Chen: A Strategy for Efficient Access of Multiple Data Items in Mobile Environments. *MDM 2002*, pp.71–78, 2002.
[11] W.G. Yee , S.B. Navathe,E. Omiecinski, C. Jermaine: Efficient Data Allocation over Multiple Channels at Broadcast Servers. *IEEE T. Comput.*, 51(10):1231–1236, 2002.
[12] W.G. Yee, S.B. Navathe: Efficient Data Access to Multi-Channel Broadcast Programs. *CIKM*, pp. 153–160, 2003.
[13] J.L. Huang, M.S. Chen, W.C. Peng: Broadcasting Dependent Data for Ordered Queries without Replication in A Multi-Channel Mobile Environment. *ICDE 2003*, pp. 692–694, 2003.
[14] M. V. Lawrence, L. S. Brakmo, W. R. Hamburgen: Energy management on handheld devices, *ACM Queue*, 1:44–52, 2003.
[15] J.L. Huang, M.S. Chen: Broadcast Program Generation for Unordered Queries with Data Replication. *SAC 2003*, pp. 866–870, 2003.
[16] K. Foltz, L. Xu, J. Bruck: Scheduling for Efficient Data Broadcast over Two Channels. *ISIT 2004*, pp. 113–116, 2004.
[17] J. Juran, A.R. Hurson, N. Vijaykrishnan, S. Kim: Data Organization and Retrieval on Parallel Air Channels: Performance and Energy Issues. Wireless Networks, 10(2): 183–195, 2004.
[18] J.L. Huang, M.S. Chen: Dependent Data Broadcasting for Unordered Queries in a Multiple Channel Mobile Environment. *TKDE*, 16(9):1143–1156, 2004.
[19] E. Ardizzoni, A.A. Bertossi, S. Ramaprasad, R. Rizzi, M.V.S. Shashanka: Optimal Skewed Data Allocation on Multiple Channels with Flat Broadcast per Channel. *IEEE T. Comput.*, 54(5):558–572, 2005.
[20] S. Jung, B. Lee, S. Pramanik: A Tree-Structured Index Allocation Method with Replication over Multiple Broadcast Channels in Wireless Environment. *TKDE*, 17(3):311–325, 2005.
[21] B. Zheng, X. Wu, X. Jin, D.L. Lee. Tosa: A Near-Optimal Scheduling Algorithm for Multi-Channel Data Broadcast. *MDM*, pp. 29–37, 2005.
[22] A.R. Hurson, A.M. Munoz-Avila, N. Orchowski, B. Shirazi, Y. Jiao: Power Aware Data Retrieval Protocols for Indexed Broadcast Parallel Channels. *Pervasive and Mobile Computing*, 2(1):85–107, 2006.
[23] Y. Yao, X. Tang, E.P. Lim, A. Sun. An Energy-Efficient and Access Latency Optimized Indexing Scheme for Wireless Data Broadcast. *TKDE*, 18(8):1111–1124, 2006.
[24] J. Xu, W.C. Lee, X. Tang, Q. Gao, S. Li: An Error-Resilient and Tunable Distributed Indexing Scheme for Wireless Data Broadcast. *TKDE*, 18(3):392–404, 2006.
[25] T. Jiang, W. Xiang, H.H. Chen, Q. Ni: Multicast Broadcast Services Support in OFDMA-Based WiMAX Systems *IEEE Commun. Mag.*, 45(8): 78–86, 2007
[26] Y. Shi, X. Gao, J. Zhong, W. Wu: Efficient Parallel Data Retrieval Protocols with MIMO Antennae for Data Broadcast in 4G Wireless Communications. *DEXA 2010*, pp.80–95, 2010.