POWER-AWARE DATA RETRIEVAL PROTOCOLS FOR INDEXED BROADCAST PARALLEL CHANNELS[1]

Ali R. Hurson[2], Angela Maria Muñoz-Avila, Neil Orchowski, Behrooz Shirazi*, and Yu Jiao
Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
{hurson, munoz, orchowski,yjiao}@cse.psu.edu

*School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA 99164-2812
shirazi@eecs.wsu.edu

[1] The Office of Naval Research and the National Science Foundation under contracts N00014-02-1-0282, IIS-0324835, IIS-0326505, and STI-0129682 in part have supported this work.
[2] Contact author

POWER-AWARE DATA RETRIEVAL PROTOCOLS FOR INDEXED BROADCAST PARALLEL CHANNELS

Abstract

In pervasive and mobile computing environments, "timely and reliable" access to public data requires methods that allow quick, efficient, and low-power access to information to overcome technological limitations of wireless communication and access devices. Literature suggests broadcasting (one-way communication) as an effective way to disseminate the public data to mobile devices. Within the scope of broadcasting, the response time and energy consumption of retrieval methods have been used as the performance metrics to measure the effectiveness of different access methods. The hardware and architecture of the mobile units offer different operational modes that consume different energy levels. Along with these architectural and hardware enhancements, techniques such as indexing, broadcasting along parallel channels, and efficient allocation and retrieval protocols can be used to minimize power consumption and access latency.

In general, the retrieval methods attempt to determine the optimal access pattern to retrieve the requested data objects on parallel broadcast channels. The employment of heuristics provides a methodology for such ideal path planning solutions. Using informative heuristics and intelligent searches of an access forest can provide a prioritized cost evaluation of access patterns for requested data objects and hence, an optimal path for the access of requested data on broadcast air channels.

This paper examines two scheduling methods that along with a set of heuristics generate and facilitate the access patterns to retrieve data objects in the presence of conflicts in an indexed parallel broadcast channel environment. A simulation of the proposed schemes is presented to analyze the relationship between response time and power consumption.

Indexed terms: Pervasive Computing, Mobile Computing, Broadcasting, Retrieval, Power management, Access Latency, Scheduling

1. Introduction

The conventional standard for timely and reliable access to global information sources is rapidly changing. This is mainly due to the advances in communication and computation technologies, availability of small mobile devices, and the sophistication of users. As a result, users have become much more demanding in their desire to access information "anytime, anywhere" in pervasive computing environments.

Multidatabase systems (MDBS) were designed to allow timely and reliable access to heterogeneous/homogeneous data sources in an environment that is characterized as "sometime, somewhere." Within the scope of multidatabases, where clients and servers are connected over a reliable network infrastructure, researchers have addressed issues such as autonomy, heterogeneity, transaction management, concurrency control, transparency, and query resolution [1]. With the introduction of pervasive computing and mobile devices into global computing, mobile clients can now have the higher flexibility of accessing the information from different locations with varying network connectivity. The concept of mobility and rapidly expanding technology is making available a wide breadth

of access devices with different physical characteristics. However, this has introduced additional complexities and restrictions in a multidatabase system, i.e., reduced capacity and unreliable network connectivity, and reduced processing and power resource.

An MDBS with such additional physical constraints is referred to as a Mobile Data Access System (MDAS) [8]. The uncertain and unpredictable temporal and spatial characteristics of the mobile devices and wireless communication often result in an undesirable Quality-of-Service. As a result, processes that stay connected to the network for the least amount of time will have more successful completion rate. Therefore, the access time of the mobile unit in the network is an important factor in evaluating the performance of the unit in an MDAS environment. Processing power, memory, storage, display capabilities, and power limitations are all common restrictions for mobile units, and these restrictions reduce their computational capabilities. Energy consumption, in particular, greatly impacts the performance of a mobile device. This is due to the fact that temporary power sources such as batteries are the main source of power in mobile devices and the rate of increase in the battery capacity is lower than the rate of increase in chip density. As mobile technology advances and computing complexity increases, energy consumption must be minimized to retain efficient and effective usage of battery power.

Architectural enhancements allow mobile units to operate in different operational modes to improve energy consumption. When not actively retrieving information, or performing any computation, the device may operate in the reduced power mode and conserve energy [2,4,5-7]. Continuing further into the realm of MDAS, low energy consumption must also be a priority when developing algorithms to organize and access the information.

In general, in a pervasive or mobile computing environment, two types of services are available to the users; on demand based services and broadcast based services. When a user's request (potentially mobile) is directed to the public data (i.e., news, weather, flight information, stock quotes) broadcasting has been suggested as an effective mechanism to access data while overcoming the technological limitations of the mobile access devices and wireless communication.

Broadcasting information is not a new concept. Whether through a guided (such as a cable) or unguided medium (such as air) the principle is the same. The concept is based on the encoding and transmission of the desired signal (analog or digital) on a certain frequency. The encoded information is supplied by one source and read by multiple receivers. In this paper, we assume that the distance between the source and receiver does not significantly affect the energy consumption of the receiver. In general, data can be broadcast either on one (single channel) or several channels (parallel air channels). Concepts such as frequency-division multiplexing (FDM), synchronous time-division multiplexing (synchronous TDM), or statistical time-division multiplexing (statistical TDM) can be used. The receiver can read data from multiple channels. For example, consider FDM scheme, where multiple signals are modulated onto multiple carrier frequencies using non-overlapping bandwidths. The receiver can read data from different frequencies by having multiple bandpass filters or adjusting the center frequency of one bandpass filter. It should be noted that switching between channels is a relatively fast operation (in the microseconds range).

Within the scope of broadcasting, indexing data entities on a broadcast channel, broadcasting data along parallel channels, and scheduling data retrieval from parallel channels have been suggested as a means to further reduce the response time and energy consumption [2,4-5,9,11]. An index provides an estimated time of arrival for the data, thus allowing the mobile unit to operate in doze mode when not actively searching for or retrieving a data element from the broadcast [12,16]. The increased operation of the device in doze mode results in an improved energy efficiency of the retrieval operation. In addition, through the use of indexing, the location of requested objects may be determined, and an ideal retrieval path can be calculated. Distribution and allocation of data elements along the parallel channels reduces the broadcast length and hence could improve the access time and power consumption.

The employment of heuristics can be used to perform a prioritized cost evaluation of access patterns for requested data objects. Artificial intelligence (AI) techniques provide methodologies for tree navigation and ideal path planning solutions. Using informative heuristics to employ the intelligent search of an access forest of data objects can provide optimal paths for the access of requested data on broadcast air channels. A drawback, however, is the computational overhead associated with the search algorithm. Generally, the more time invested in finding a solution, the more likely that solution is closer to optimal [3]. The tradeoff between response time and ideal path selection must be examined when employing intelligence in the retrieval algorithm.

This work examines a method to minimize energy consumption and response time when pulling data from indexed parallel broadcast channels for mobile devices. It will propose and analyze two protocols that schedule data retrieval from indexed parallel broadcast channels based on a prioritized set of heuristics with a goal to minimize response time and/or power consumption.

The paper is organized as follows. Section 2 overviews the background information for this work, this includes a discussion about data broadcasting, artificial intelligence, and the previous efforts to allocate and retrieve data to/from parallel channels. Section 3 studies the proposed data retrieval protocols that aim to reduce energy consumption and response time in an indexed parallel broadcast channels environment. Simulation of the proposed protocols and analysis of the simulated results are the subjects of Section 4. Finally, the conclusions derived from this work and suggestions for future research directions are discussed in Section 5.

2. Background

Mobile computing has become a reality. The emergence of ubiquitous, small, and mobile devices along with the development of fast, reliable, and accessible networks has helped to advance the research in the pervasive computing area. In the wireless mobile computing environment, massive numbers of low power mobile units query databases over a wireless communication medium. Restrictions on bandwidth, power, processing, storage, and memory are some of the issues that these units have to deal with in order to maintain portability and mobility [6].

The focus of this research is geared towards the following two requirements:

1. Minimizing the overall response time of the mobile unit, and

2. Minimizing the amount of power consumed in the retrieval process of the mobile unit.

When developing retrieval methods for requested data objects, these criteria govern the evaluation of the algorithm's performance and provide motivation for the development of improved techniques.

2.1 Broadcasting on Parallel Air Channels

Public data consists of frequently requested information that must be circulated to a vast number of users requesting the information. One-way data communication, known as broadcasting, is one way to effectively allocate and distribute public information over an information network. This method of communication is especially suited for a wireless environment that is limited by degraded bandwidth and resource restrictions. In addition, broadcasting scales up as the number of users increases – the necessity of dividing bandwidth between users is eliminated, and the air channels may also be used as additional storage space [5,6,9].

Broadcasting operates in a cyclic pattern and access devices must only listen or "tune-in" to a channel to retrieve information. For parallel air channels, the information may be spread out amongst the entire broadcast and on different channels. One issue associated with parallel air channels is the manner in which to efficiently and successfully locate the data element that is requested for retrieval. The literature has shown indexing as a technique well suited for locating data entities on parallel air channels [2,4,5,7,14]. Signature-based and tree-based indexing, as suggested in the literature, are the most representative of this approach [2,5]. Signature-based indexing is a technique that makes use of a signature to examine data objects for the information that is being requested. A signature is basically an abstraction of the information contained in a data object. The signature alone can then be examined to see if the data item contains the information being requested. For signatures that do not match the requested information, the mobile unit may return to a low-power doze mode and wait for the next signature. In tree-based indexing, the index includes a set of supplemental information that provides the time of arrival and location of requested objects in the broadcast. After determining the time of arrival for the object, the mobile unit may revert to doze mode and conserve power until the data item arrives. Both methods have shown reduced "tune-in" time and reduced time in the high-power operational mode during the broadcast. Comparing the results of the two methods, however, reveals that signature-based indexing entails less access time computation overhead than the tree-based indexing method [5]. On the other hand, the energy consumption of the tree-based indexing is superior to the signature-based indexing due to a shorter "tune-in" time and better spatial locality given by the global information provided in the index.

The application of an index along the air channel incurs a space overhead that increases the broadcast length and hence a longer response time. Based on the studies reported in [2], it was concluded that indexing along the broadcast channel degrades the response time, moderately. However, such degradation is greatly offset by the improvement in energy consumption. To remedy this performance degradation, indexing along parallel air channels was studied in [7,9]. This study showed that allocation of the data elements on the parallel air channels reduces the broadcast length, access latency, and power consumption.

2.2 Conflicts in parallel air channels

One of the main problems in a parallel channel environment is the possibility of conflicts while pulling data elements from the air channels. An access conflict affects both the response time and power consumption. If there is a conflict, the retrieval process has to wait for the next broadcast cycle (s) to download the remaining requested data elements [7].

***Definition 1**: Two data elements are defined to be in conflict if it is impossible to retrieve both on the same broadcast cycle.*

***Definition 2**: A K-object request is an application request intended to retrieve K data elements from a broadcast. In our model, we assume that each channel has the same number of pages of equal length and, without loss of generality; each data element is residing on only a single page.*

A parallel broadcast can be modeled as an N x M grid, where N is the number of pages per broadcast, and M is the number of channels. In this grid, for a K-object request, K data elements ($0 \le K \le MN$) are randomly distributed throughout the MN positions of the grid. The mobile unit can only tune into one channel at a time, however, it can switch channels, but it takes time to do this — during channel switching the mobile unit cannot retrieve data from the air channel. This brings up the *overlapped page range* issue. An overlapped page of N means that a data element is in conflict with another data element on different channels in the same page or the next N-1 pages. Based on the common page size and the network speed, the time required to switch from one channel to another is equivalent to the time it takes for one page to pass in the broadcast ─ as a result, in this study, the overlapped page range is 2 [9]. Thus, it is impossible for the mobile unit to retrieve both the ith page on channel A and $(i+1)^{th}$ page on channel B (where $A \neq B$). Figure 1 depicts the aforementioned grid model and conflicts.

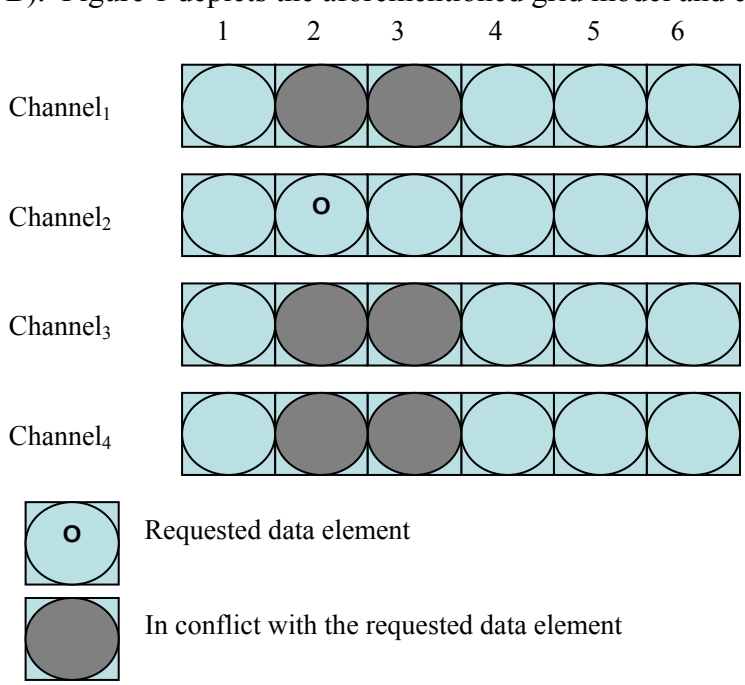

Figure 1. Sample Broadcast with M = 4, and N = 6.

In order to predict and analyze the impact of the conflicts, statistical techniques could be used to calculate the expected number of broadcast cycles required to retrieve a set of objects. In our previous efforts, a set of heuristics was used to minimize the effect of conflict on access latency and power consumption. Two heuristics namely, *Next Object* and *Row Scan* heuristics were used [7]. The *Next Object* heuristic always retrieves the next available data element in the broadcast, while the *Row Scan* strategy reads all the data elements from one channel in each pass. A benefit of these heuristics is that they do not incur any computational overhead, however, they have no means of reducing the number of passes required for retrieving the requested data elements; therefore, the response time may be higher in comparison to an algorithm that employs channel switching as a means to reduce the number of broadcast passes. Previous studies showed that when the number of requested data elements exceeds 45% of the total objects, *Row Scan* heuristic provides a better solution than *Next Object*; moreover, it reduces the delay associated with switching between channels.

2.3 Performance Metrics

Our performance analysis of the proposed retrieval algorithms is based on two performance metrics: response time, and energy consumption, which are proportional to:

- Access Time – The elapsed time from a mobile unit's generation of a request until the requested data items are retrieved.
- Tune-in Time – The time during which a mobile client is active and retrieving information from the data broadcast.

The access time performance metric gives a general evaluation of the efficiency of the retrieval method based on the total latency involved with retrieving all of the items. The tune-in time, however, gives information that is useful for deducing the power consumption reduction capabilities of the algorithm. While the computer is not "tuned in" to the broadcast it may operate in a low-power doze mode and conserve energy. Using this correlation, the less tune-in time that occurs during a retrieval operation, the less energy that will be consumed in the process.

2.4 Artificial Intelligence

Artificial intelligence techniques are well suited for path planning tasks. This makes artificial intelligence techniques good candidates for path planning of user requests on indexed parallel air channels [3]. Techniques such as depth-first search, breadth-first search, uniform cost method, and best-first search [3,18] could be employed to efficiently retrieve data elements from a broadcast while taking the response time and energy consumption constraints into consideration. To maximize the results, heuristics may be developed to generate access patterns with the least latency and energy costs. When using heuristic information an estimate can be made for determining how promising a configuration is with respect to reaching a goal, therefore fitting the design of a best-first search [18].

A greedy search makes use of heuristics to evaluate a search tree in a top down fashion. Using the heuristics to evaluate the computed cost of each node in the tree, an ideal path can be expanded that produces the optimal tree(s) ─ only the node that predicts the lowest cost will be expanded, thus reducing the size of the overall access forest. Once the best-first algorithm processes the entire tree, the result should provide the optimal

path(s) that maximizes the heuristics employed in the algorithm. One drawback to this method, however, is computational overhead that will increase the response time of the search method due to computational complexities of the algorithm ─ a tradeoff between response computation-time and response execution-time in a real-time situation [3].

3. Energy-Efficient Data Retrieval Scheme for Indexed Broadcast Parallel Channels

Proper allocation and retrieval mechanisms are needed to reduce the energy consumption of a mobile unit in the presence of conflicts. In addition, these mechanisms have to satisfy the *timely access* requirement of a mobile user regardless of the complexity of the request.

3.1 Retrieval of Objects from Indexed Broadcast Parallel Channels

In indexed broadcast parallel channels an access protocol is composed of three steps:

1) ***Initial probe***: The client tunes into the broadcast channel to determine when the next index is broadcast.
2) ***Search***: The client accesses the index and determines the offset for the requested data elements.
3) ***Retrieve***: The client tunes into the channel and downloads all the required data elements.

The retrieval protocols proposed in this paper focus on the last two steps of the general access protocol (steps 2 and 3). The idea is to produce an ordered access list of requested objects that reduces:

- The number of passes over the air channels, and/or
- The number of channel switches.

During the search step, the index is accessed to determine the offset and the channel of the requested data elements. With this information available a sequence of access patterns to pull data from the air channels is generated using the retrieval scheme proposed in subsequent sections. Finally the retrieval step is performed following the generated access patterns. Figure 2 shows a more detailed operational sequence:

1. Probe the channel and retrieve offset to the next index.
2. Access the next index
3. Do {Search the index for the requested data element
4. Calculate the offset of the data element
5. Get the channel on which the data element will be broadcast
6. } while there is an unprocessed requested data element
7. Generate access patterns for the requested data elements (using retrieval scheme)
8. Do {Wait for the next broadcast cycle
9. Do {Reach the first data element as indicated by the access pattern
10. Retrieve the data element
11. } while there is an un-retrieved data element in the access pattern
12. } while there are unprocessed access patterns

Figure 2. Flow diagram of the proposed retrieval scheme.

3.2 **Retrieval Scheme1 ─** Reducing the number of passes

The retrieval scheme based on a set of heuristics determines the sequence of data elements to be retrieved in each broadcast cycle. It attempts to minimize the energy consumption of the mobile unit and the total number of passes required to satisfy the user query. The schedule is determined based on the following three prioritized conditions:

1) *Eliminate the maximum number of conflicts.*
2) *Retrieve the maximum number of data elements.*
3) *Minimize the number of channel switches.*

The scheme determines the order of retrieval utilizing a forest — an *access forest*.

Definition 3: *An access forest is a collection of trees (access trees), where each access tree represents an access pattern during a broadcast cycle. The structure of the access forest, i.e., the number of trees and the number of children that any parent can have, is a function of the number of broadcast channels.*

An *access tree* is composed of two elements; nodes and arcs.

- **Node:** A node represents a requested data element. The nodes are labeled to indicate its conflict status: mnemonically, "C_1" represents if the data element is in conflict with another data element(s) in the broadcast; and "C_0" indicates the lack of conflict.

  Each access tree in the access forest has a different node as a root — the root node is the first accessible requested data element on a broadcast cycle. This implies that an access forest can have at most *n* trees where *n* is the number of broadcast channels.
- **Arcs**: The arcs of the trees connect the nodes and are weighted. The weight denotes whether or not a channel switch is required in order to retrieve the next scheduled data element in the access pattern. A branch in a tree (a sequence of nodes and arcs) represents a possible access pattern of data elements during a broadcast cycle with no conflicts. Starting from the root, the total number of branches in the tree represents all possible access patterns during a broadcast cycle.

This scheme allows one to generate all possible non-conflicting weighted access patterns from all channels. The generated access patterns are ranked based on their weight — a weight is the sum of the arc weights along the pattern — and then the one(s) that allows the maximum number of data retrievals with the minimum number of channel switches (lowest weight) is selected. It should be noted that the time needed to build and traverse the access forest is a critical factor that must be taken into account to justify the validity of this approach. This issue will be addressed later. The following working example provides a detailed guide to illustrate the steps taken to generate the access patterns for each broadcast cycle:

Step 1) **Search**: Based on the user's query, this step determines the offset and the channel number of the requested data elements on the broadcast channels. Figure 3 depicts a request for eight data elements from a parallel broadcast on 4channels.

Step 2) **Generation of access forest:** For each broadcast channel, search for the requested data element with the smallest offset (these data elements represent the roots of the access tree). According to our running example, the data elements with the smallest offsets are O_1, O_3, O_6 and O_8.

Step 3) **Root node assignment:** For each channel with at least one data element requested, generate a tree with root node as determined in step 2. The roots are temporarily tagged as "C_0".

Step 4) **Child node assignment:** Once the roots are determined, it is necessary to select the child or children nodes of each rooted access tree: for each root, and relative to its position on the air channel, the algorithm determines the closest non conflicting requested data element on each channel. With respect to a data element $O_{i,x}$ at location X on air channel i,($1 \leq i \leq n$) the closest non conflicting data element is either the data element $O_{i,x+1}$ or the data element $O_{j,x+2}$ $j \neq i$. If the child is in the same broadcast channel as the root, the arc is weighted as "0"; otherwise it is weighted as "1". Each added node is temporarily tagged as "C_0". Figure 4 shows the snapshot of our running example after this step.

The sequence of aforementioned operations, iteratively, is applied to every node of each access tree.

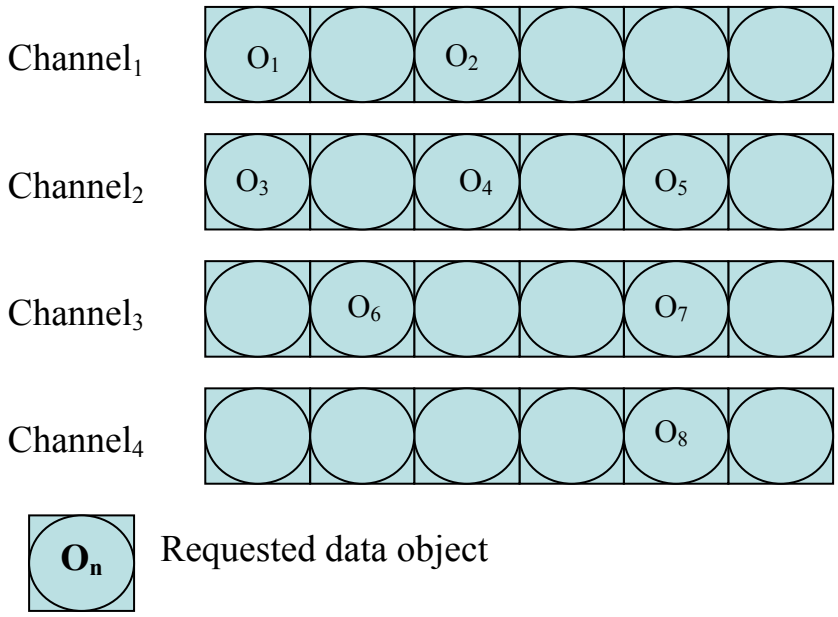

Figure 3. A parallel broadcast on 4 channels with eight requested data elements.

Step 5) **Root node label update:** Once the whole set of requested data elements is analyzed and the access forest is generated, the conflict labels of the nodes of each tree are updated. This process starts with the root of each tree. If a root is in conflict with any other root(s) a label of "C_1" is assigned to all the roots involved in the conflict. Otherwise the preset value of "C_0" is maintained.

Step 6) **Child node label update**: Step 5 will be applied to the nodes in the same level of each access tree in the access forest. As in step 5 a value of "C_1" is assigned to the nodes in conflict. Otherwise the preset value of "C_0" is maintained. Figure 5 shows our example with the updated labels.

Step 7) **Sequence selection:** The generation of the access forest then allows the selection of the suitable access patterns in an attempt to reduce the network

latency and power consumption. A suitable access pattern is then equivalent to the selection of a tree branch that:

- Eliminates the most conflicts with other branches,
- Allows more data elements to be pulled off the air channels, and
- Requires the least number of channel switches.

The sequence O_3, O_4 and O_5, represents the most suitable access pattern of our running example during the first broadcast cycle. Step 7 will be repeated to generate access patterns for different broadcast cycles. The algorithm terminates when all the requested data elements are covered in different access patterns. The sequences of data elements O_1, O_2 and O_7 and data elements O_6 and O_8 represent the last two access patterns for retrieving all of the data elements requested in our initial sample example.

Retrieval scheme1, as expected, generates a forest that grows rapidly with the number of requested data elements ─ data elements (nodes) stored in the forest are replicated in various sub-trees. The key observation needed to reduce the size of the tree is to recognize that each requested object has a unique list of children (next accessible objects) and the number of children for a particular object is limited to the number of channels. Thus, if the information is not duplicated, the trees can be represented in O(number of elements * number of channels) size. Furthermore, each node (object) has a unique best branch. This branch can be no longer than the total number of data elements, because no objects can appear more than once in a branch. So, assuming no branch is duplicated, the cost of calculating the best branch is limited to $O(K^2)$ where K is the number of objects requested. The simulator takes advantage of these observations to reduce the size of the retrieval tree and the calculation time without sacrificing accuracy.

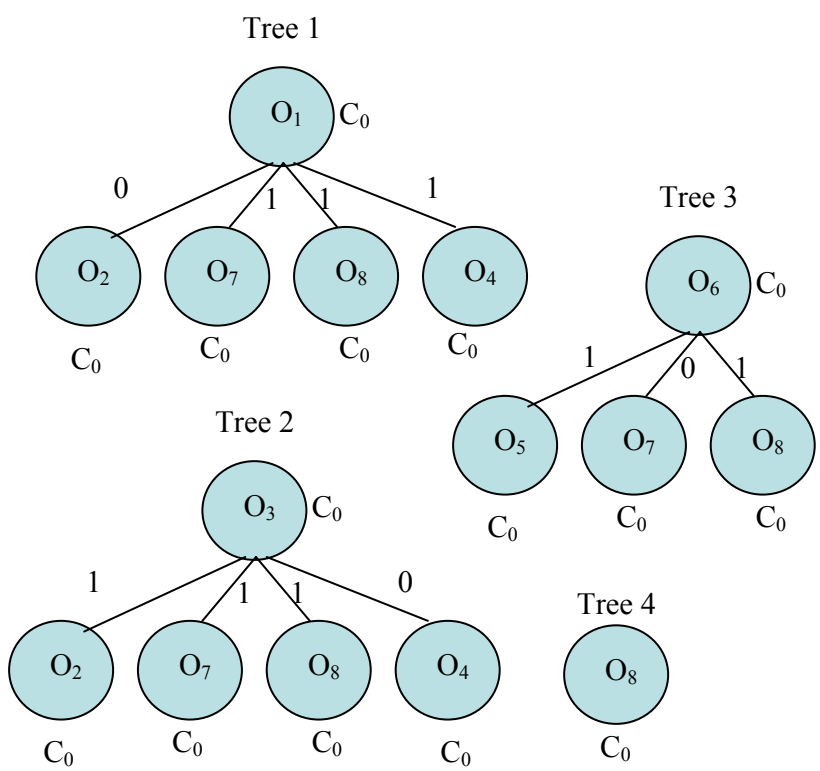

Figure 4. Children of each root.

As will be discussed in the next section, the simulation of the *retrieval scheme1* showed that it offers a better response time and power consumption than *Row Scan* and *next-object* schemes when a small percentage of the total broadcast data is requested by the user. However, as the number of requested data elements increases, the energy consumption of the mobile unit becomes worse than *Row Scan* due to the excessive channel switches occurring during the data retrieval. This motivated us to develop the *retrieval scheme2* that is the subject of the next section.

3.3 Retrieval Scheme2 ─ Reducing the number of channel switches

Retrieval scheme2 attempts to overcome the problem associated with *retrieval scheme1* through the use of artificial intelligence techniques to generate the access patterns with the goal of reducing the number of channel switches while attemting to reduce access latency.

3.3.1 Revised Heuristics

Retrieval scheme1 generates excessive channel switches to pull a large number of data elements. This in turn increases the energy consumption at the mobile unit. To further reduce energy consumption, the priority list of heuristics was reordered as follows:

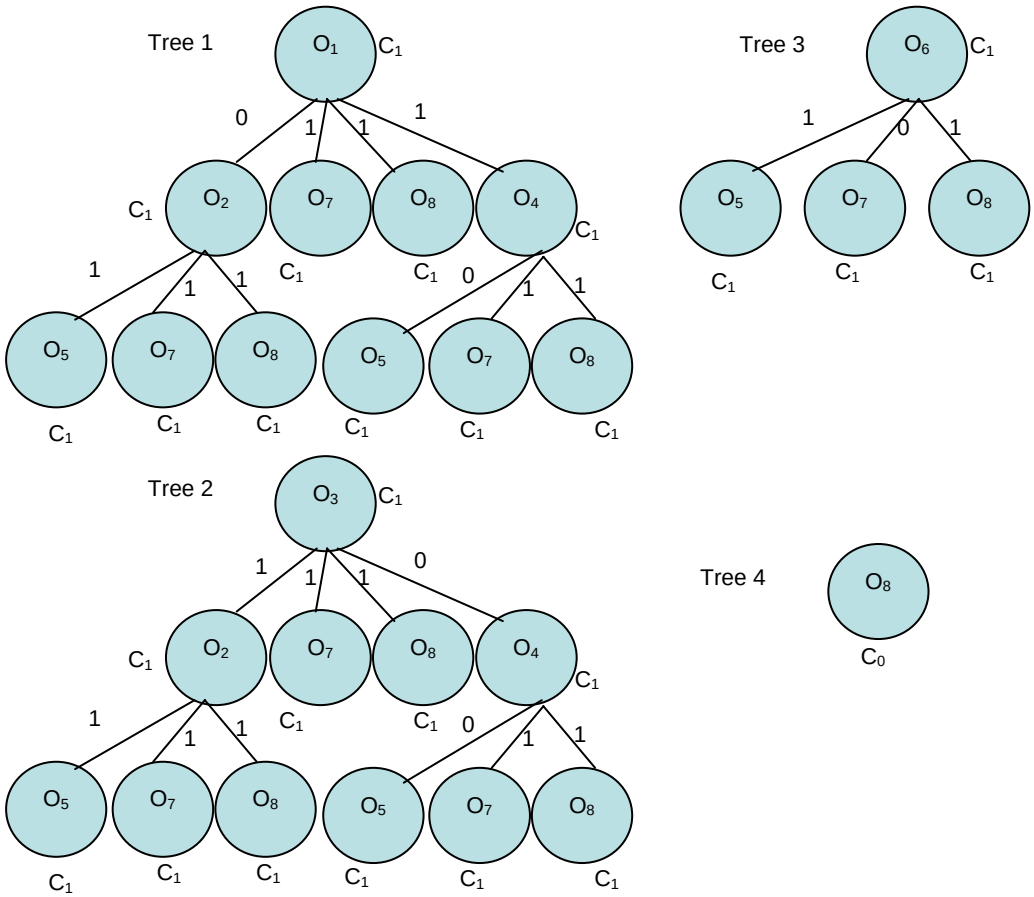

Figure 5. Final state of the access forest.

1) *Eliminate the maximum number of conflicts.*

2) *Minimize the number of channel switches.*
3) *Retrieve the maximum number of data elements.*

3.3.2 Access Protocol – Least-Cost Path Retrieval Method

Similar to *Retrieval Scheme1*, *Retrieval Scheme2* makes use of the temporal and spatial information provided by the index to plot an access pattern for the requested data elements. The recursive method used in *Retrieval Scheme1* requires an entire access forest to be constructed prior to path selection, and then the trees were individually searched in a bottom-up fashion to find the best path. By the nature of the bottom-up design, every single branch was traversed even though some branches would obviously produce a non-ideal path.

To improve the efficiency and effectiveness of *Retrieval Scheme2*, it employs a top-down "best-first search" method. The general philosophy of this method is to use heuristic information to assess the cost of every search avenue, and then continue the search down the path with the lowest cost. This method can also be referred to as the "Least-Cost Path" method. By the nature of the search, nodes on the same level in a tree that possess a higher cost will produce non-ideal patterns. Thus, higher cost nodes are eliminated, and only those nodes that provide a potential least-cost path are expanded and searched further. For user requests where K is large, the least-cost path approach provides a means for reducing the number of branches to be searched, thus having the potential to reduce the overall time spent searching for an ideal access pattern. Our previous running example is used to clarify the method.

1) **Probe the index:** The index is retrieved and used to determine the channel number and offset of each requested data element.
2) **Generate Access Forest:** For each channel, find the first requested data element to be broadcast (if there is one). The root nodes in the running example are O_1, O_3, O_6, and O_8.
3) **Root Node Assignment:** For each root node from step 2, temporarily assign a tag of "C_0" and create an access tree with the root.
4) **Child Node Assignment:** For each node, find the closest non-conflicting data elements that may be accessed later on in the broadcast. Label the arc for each child respectively – "1" if the child is on a different channel, "0" if the child is on the same channel. All child nodes are temporarily assigned a tag of "C_0". Figure 4 shows the state of the access trees after generating the children of the roots.

It is at this point where *retrieval scheme2* differs from *retrieval scheme1*.

5) **Node Label Update:** After the generation of the children, they are examined and assessed for conflicts. If the parents of the children are the root nodes (i.e. the search just started) the root nodes are also examined for conflicts. A node that is in conflict with at least one other node is labeled as "C_1", and a node that is not in conflict with any node is labeled as "C_0".
6) **Cost Evaluation:** At this stage of the retrieval method, the children are examined to determine the least-cost path of the tree. Using the weights assigned to the arcs of each child, the switching cost of each node is calculated. The node with the lowest switching cost (arc with label "0") will be expanded in step 7. If more than one node has the lowest switching cost, then all paths are to be expanded.
7) **Expansion:** Using the node with the lowest switching cost found in step 6, the children of that node are determined to expand the least-cost path. The children

are added as defined in step 4. Step 5 is repeated to further define the least-cost path.

8) **Repeat:** Step 7 is repeated until all least-cost paths can no longer be expanded. The fully expanded least-cost paths for the running example are depicted in figure 6.
9) **Compare:** If more than one least-cost path is determined for each tree, the heuristics are used to determine the most ideal path to follow. The least-cost path to use will be the access pattern that:
    1. Eliminates the most conflicts (has the highest number of "C_1"nodes)
    2. Uses the least channel switches (the path whose arcs are weighted with the most "0"s).
    3. Retrieves the most data elements (the highest node count)

Referring to the running example, the least-cost path selected by the best-first search is depicted in figure 6. After the access pattern for the first broadcast pass is determined, steps 2-9 are repeated to generate a new updated forest until every data element in the user's request is scheduled for retrieval. Once all data elements are covered by access patterns, the algorithm is complete, and the data elements may be retrieved from the broadcast using the determined order. Figure 7 shows the complete access patterns of our running example.

By comparing Figures 5 and 6 one can conclude that the *retrieval scheme2* generates a smaller access forest relative to *retrieval scheme1*. The smaller forest results in fewer branches to be searched, thus, reducing the overhead of the algorithm.

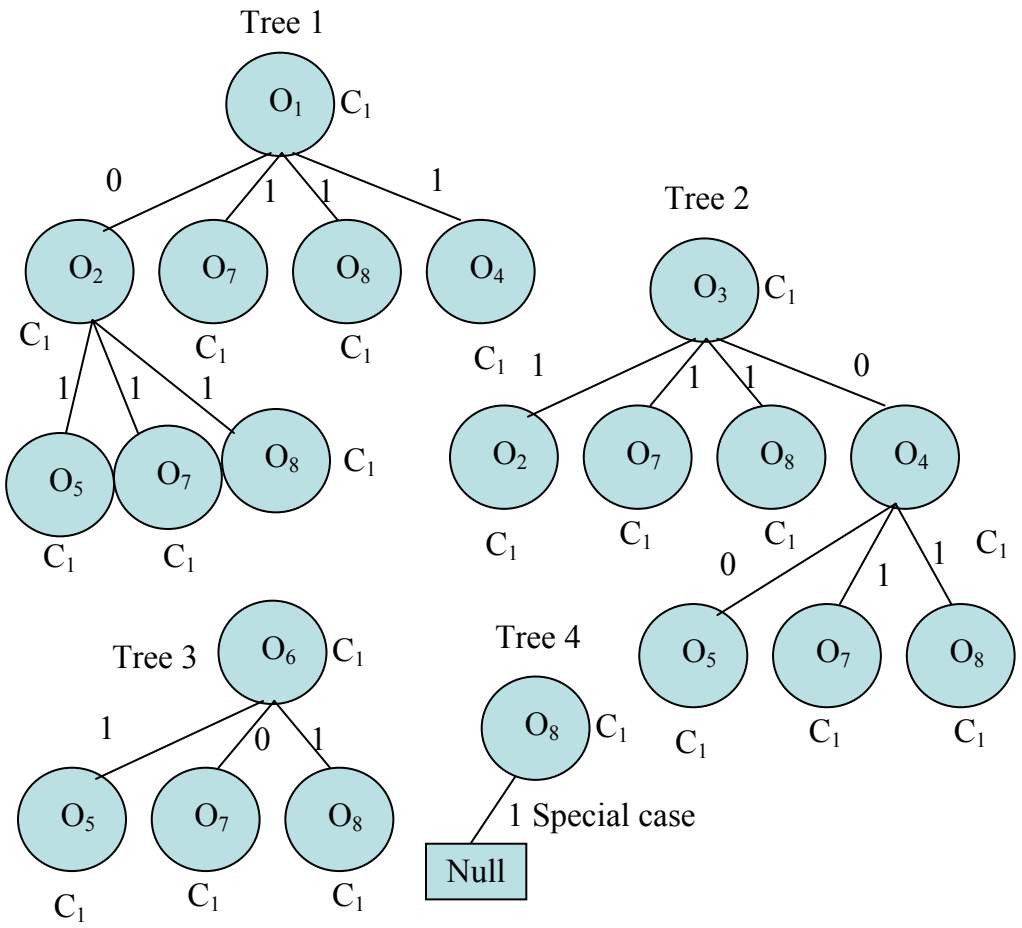

Figure 6: Final Access Forest Using AI Method.

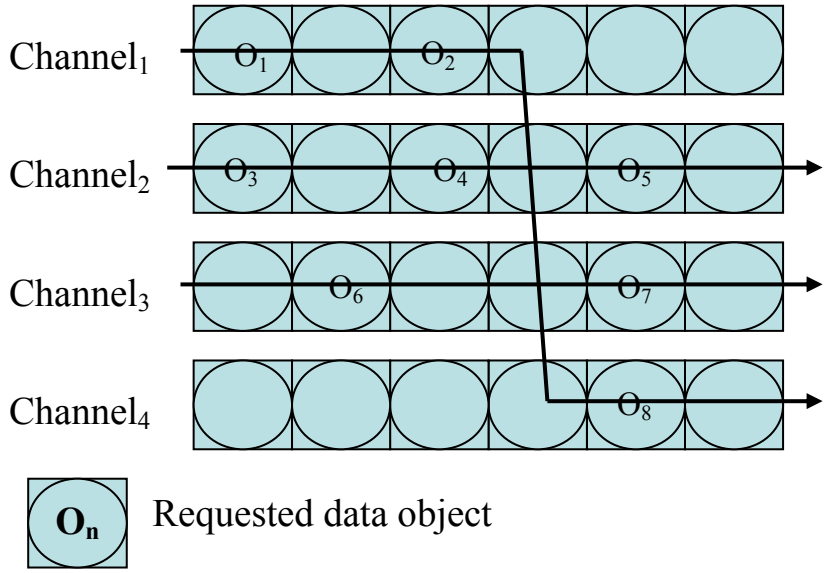

Figure 7: Complete access patterns of our running example.

4. Simulation and Results

To validate the feasibility of the proposed scheduling protocols, to compare and contrast them against each other, and to show the trade-off between access latency and computational complexity the scope of our simulator [7] was extended. The response time and energy consumption of the retrieval schemes presented were used as the performance metrics. The extended simulator emulates the process of accessing data from a hierarchical indexing scheme in parallel air channels. Moreover, the simulator also analyzes the effect of conflicts on aforementioned performance metrics. This section presents and analyzes the results of the simulations developed.

4.1 Description of Simulator

We employed a hierarchical indexing scheme to organize data elements on parallel air channels [2]. The index structure can be transmitted in different manners including:

- A complete index is transmitted at the beginning of each broadcast in the first channel before the data,
- Index is distributed among the data elements,
- Index is replicated and interleaved with the data elements, or
- A dedicated channel is used to exclusively and continuously transmit the index in a cyclic manner.

The simulation results indicated that repeated transmission of the index on a separate channel offers the best response time. Therefore, in the rest of this paper, for the sake of space, the simulation results are based on a configuration in which a dedicated channel is used to exclusively transmit the index. The simulator models a mobile unit retrieving data from indexed parallel broadcast channels represented as an $N \times M$ array with N data elements per channel on M channels. User requests are generated randomly to represent a distribution of K data elements in the broadcast. Throughout the simulation, the value of K was varied from 1 to $N \times M$ to analyze the performance of the algorithm for the full range of possible user requests. The performance of the retrieval protocols were evaluated using several metrics reported for each simulation run. Times associated with the retrieval operation are reported for three different parameters: total response time, active time (tune-in time), and doze time. In addition to timing measurements, the simulator reports the number of broadcast passes per user request, and the number of channel switches used during the retrieval. The power is the amount of energy consumed per unit time; using the collected information, the energy consumption of the retrieval process is calculated using equation 1 (As noted in the literature, the switching cost for power consumption is 10% of the active mode power consumption [7]).

$$\begin{aligned} &Energy\ Consumption = (AccessTime - Tune\text{-}inTime)*DozeModePower \\ &+ (Tune\text{-}inTime)*ActiveModePower + TheNumberOfSwitching *10\% *ActiveModePower \end{aligned} \quad (1)$$

To generate the mentioned timing and energy consumption measurements, and to take future technological advances into account, parameters such as transmission rate and power consumption in different modes of operation were fed to the simulator as variable entities. A summary of the parameters used for simulation appears in table 1. Similar to [2,9], the data being broadcast was based on a NASDAQ [10] database of 4290

securities. The size of each data element is 512 bytes of text-based securities information.

For the sake of simplicity and without loss of generality, we assume that each data item spans only one page and is not fragmented across pages ─ A page can be considered to carry the minimum unit of data. Larger objects can be split across multiple pages and without loss of generality, each such sub-object can be treated as different objects all of which must be retrieved.

The size of the index structure is dependent on the number of distinct keys for data elements. For these simulations in particular, the size of the index was 13.52% of the size of the data elements (not including the index). The number of channels varied from 1 to 16 to measure the relationship between the number of channels, switching frequency among channels, conflicts, power consumption, and response time.

Table 1: Input Parameters

Parameter	Value (Default/Range)
Number of data elements on Broadcast	4290
Number of Channels	1-16
Size of Air-Channel Page	512 Bytes
Broadcast Data Rate	1 Mbit/sec
Power Consumption (Active Mode)	130 mW
Power Consumption (Doze Mode)	6.6 mW
Power Consumption (Switching channels)	13 mW

4.2 Simulation Results

For each simulation run, a request of *K* data elements was randomly generated. A set of input parameters including the number of parallel air channels, the broadcast transmission rate, and the power consumption in different modes of operation was passed to the simulator. The simulator was run 1000 times and the averages of the performance metrics were calculated.

The retrieval protocols, discussed in section 3, are intended to reduce the number of passes over parallel channels by scheduling data retrieval. This by default should reduce the average response time. To show this fact, in a configuration composed of 2, 4, 8, and 16 channels *retrieval scheme1* and the *Row Scan* algorithm were simulated when the number of requested data elements was varied between 5 and 50. The simulation results showed that, regardless of the number of parallel air channels, *retrieval scheme1* reduces both the number of passes and the response time compared to the *Row Scan* algorithm. Moreover, the energy consumption was also reduced, but only when the number of data elements retrieved was approximately 15 or less. For example, in an environment composed of 16 parallel air channels when requesting 10 data elements (see Table 2), the *retrieval scheme1* performed the retrieval with:

- 72% fewer passes,
- 41% reduced response time, and
- 3% less energy

than that of the *Row Scan* algorithm.

Table 2: Improvement of *retrieval scheme1* vs. *Row Scan* (10 objects)

# of Channels	# of Passes	Response time	Energy
2	48.0%	28.0%	2.7%
4	68.0%	43.6%	3.1%
8	72.3%	46.5%	3.3%
16	71.8%	40.8%	3.4%

However, relative to the *Row Scan* algorithm, one should also consider the expected overhead of the proposed retrieval schemes. The simulation results showed that in the worst case, the overhead of *retrieval scheme1* was slightly less than the time required to transmit one data page. To minimize the impact of this overhead, one may propose a delay gap of one page between the transmission of the index and data objects. This would allow the mobile unit to retrieve any sequence of objects without the risk of missing any object.

a) Response Time

As expected, the simulation results showed that, regardless of the underlying retrieval protocol, the response time decreases as the number of channels increases. In addition, the response time increases as the number of requested data elements increases. Finally, for all cases, *retrieval scheme2* (re-ordered sequence of heuristics), relative to *retrieval scheme1* compromises the response time (Figure 8). In other words, the use of the revised sequence of heuristics produces access patterns that required more time to retrieve the data elements requested ─ the additional time requirement can be associated to the goal of reducing channel switches during the retrieval, thus incurring more broadcast passes that inherently increase the access latency. From Figure 8 it can be observed that the least-cost path technique is more efficient in generating the access patterns than the technique used in *retrieval scheme1*.

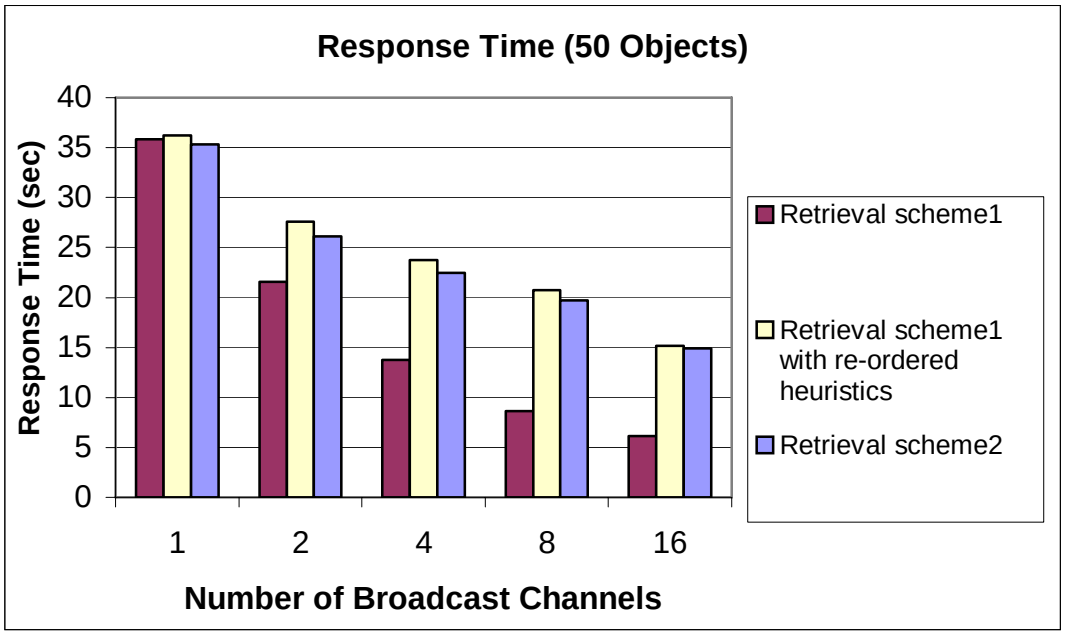

Figure 8: Response times comparison for different retrieval protocols.

The retrieval protocols proposed in Section 3 attempt primarily to reduce the conflicts in each pass of the broadcast; however, when the number of potential conflicts increases considerably, some conflicts become unavoidable causing an increase in the number of passes and hence an increase in the response time. The increase in the number of passes makes the retrieval scheme inadequate when the percentage of data elements requested is large. The simulation results also showed that when the percentage of requested data elements approaches 100%, the response time reduces. This shows the validity of the proposed scheduling protocols since, when a relatively large number of data elements are requested, they generate the same retrieval sequence as the *Row Scan* method would.

b) Channel Switching Frequency

In general, with respect to *retrieval scheme1* as the number of requested data elements increases, up to a threshold value, one should observe more frequent channel switches. This is due to the increase in the number of conflicts. As the number of conflicts increases, *retrieval scheme1* attempts to reduce their effect on the response time through the use of channel switches. The number of channel switches reaches a maximum and begins to decrease when retrieving more than 50% of the broadcast. This decrease occurs because of the heuristic rule employed that attempts "to maximize the number of data elements retrieved during each broadcast cycle". As the density of requested objects on a broadcast channel increases, *retrieval scheme1* does not have to switch channels as often to retrieve the maximum number of requests (Figure 9).

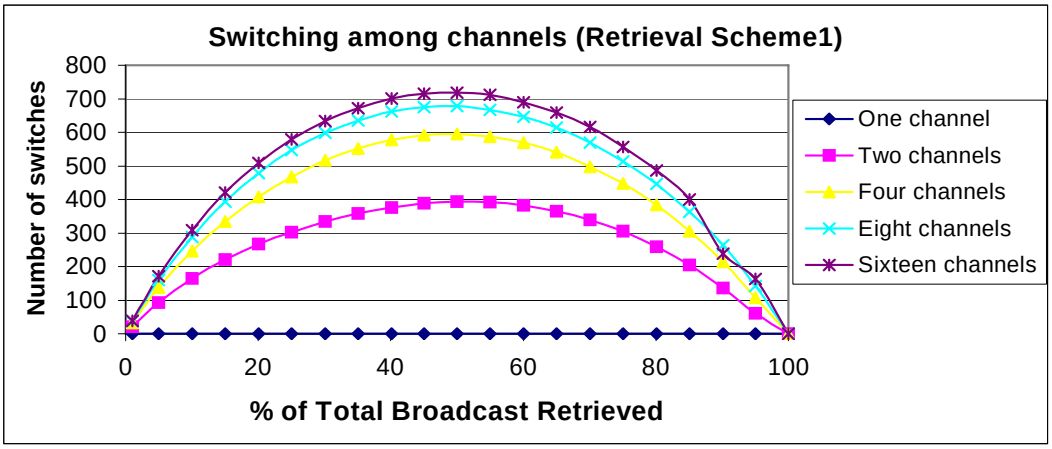

Figure 9. Channel Switching Frequency (Retrieval Scheme 1).

Retrieval scheme2 compromises the response time in order to minimize the number of channel switches. Figure 10 depicts the channel switching frequency when *retrieval scheme2* is employed. The maximum number of channel switches is reached for when size of a user query is less than 5%. At this point, the number of broadcast passes approaches the number of channels, thus mimicking the access pattern of the *Row Scan* Method. In short, *retrieval scheme2* does not allow the number of broadcast passes to exceed the number of channels, and thus as the number of requested data elements increases, the number of channel switches decreases.

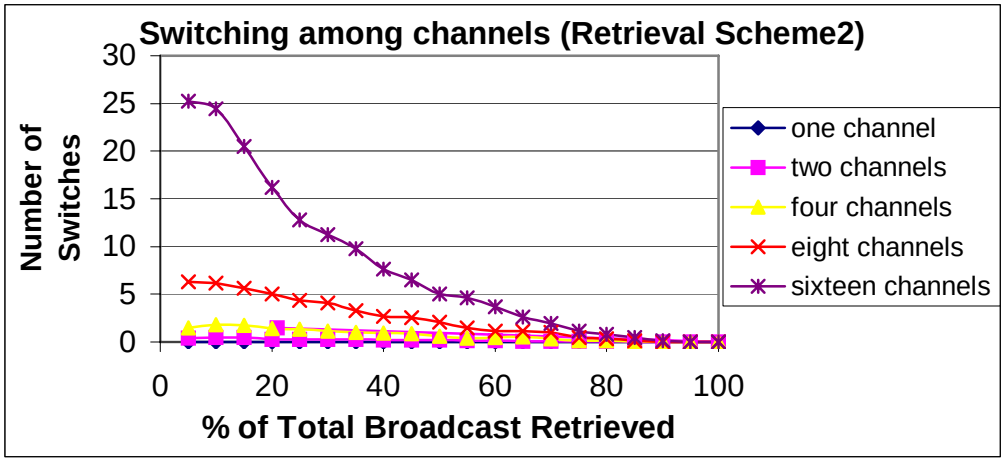

Figure 10: Channel Switching Frequency (Retrieval Scheme2).

c) Energy Consumption

In general, the energy consumption follows the same pattern as the channel switching frequency. For retrieval scheme1, the energy consumption is dominated by the number of channel switches. As a result, the energy consumption increases as the number of channels increases. In addition, the energy consumption increases, up to a threshold point, as the number of requested data elements increases, and then it decreases as the number of requested data elements continues to increases (Figure 11).

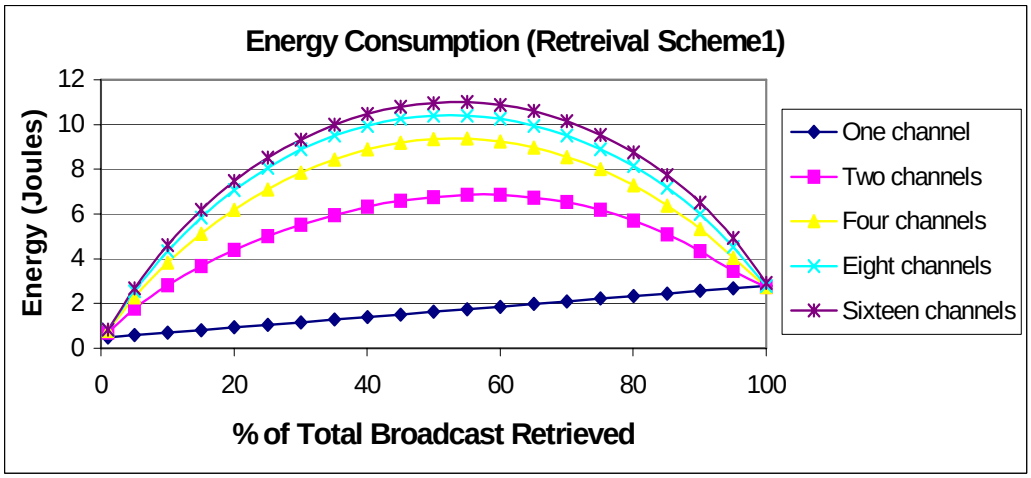

Figure 11: Energy consumption (Retrieval Scheme1).

Figure 12 depicts the energy consumption of *retrieval scheme2* retrieving data elements from the entire range of the broadcast. The figure illustrates that the energy consumption increases as the number of data elements being retrieved increases. The increase in energy consumption follows a linear trend that is directly related to the increased of data elements requested. This is because the retrieval of a data element implies the active operational mode. In addition, *retrieval scheme2* attempts to minimize the channel switching frequency. As a result, the energy consumption is dominated by the number of requested data elements.

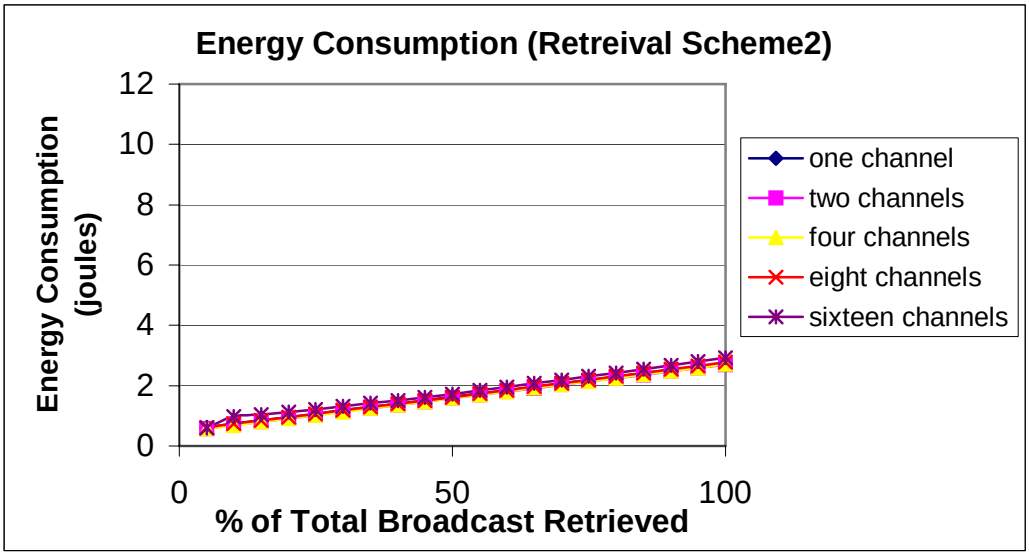

Figure 12: Energy consumption (Retrieval Scheme2).

d) Number of passes

The number of passes directly relates to the number of conflicts. As a result, as the number of requested data elements and/or the number of channels increases, one should expect to observe more passes over the parallel air channels. This by default implies higher access latency. An increase in the number of channels implies an increase in the number of conflicts, and hence the higher possibility of unavoidable conflicts, resulting in an increase in the number of passes. Figure 13 demonstrates these facts. Interestingly, when the number of requested data elements is large, the number of passes exceeds the number of channels available. This is due to the priority order of the heuristics used in *retrieval scheme1* — the proposed method tries to reduce the amount of conflicts first. However, it is not likely that a query of public data would involve a substantial percentage of the total data available, hence it can be concluded that in general, *retrieval scheme1* reduces the number of passes.

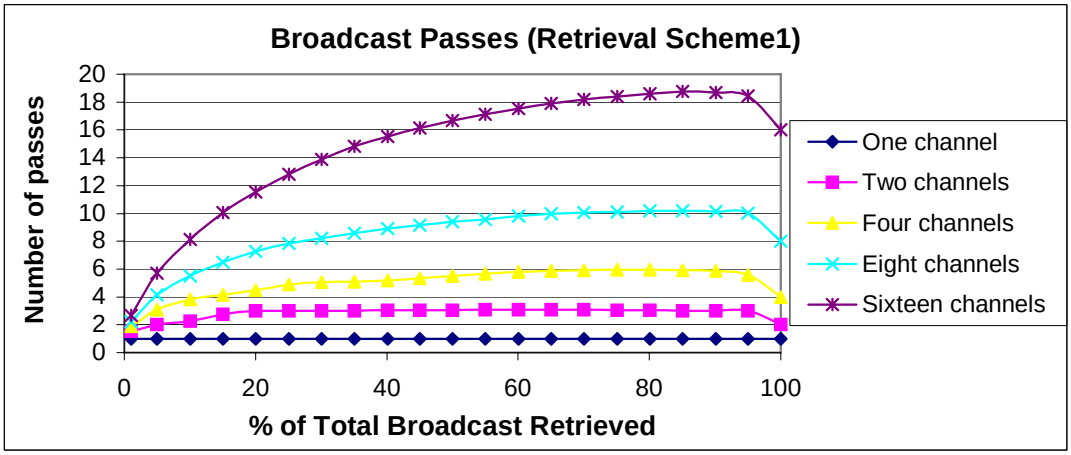

Figure 13: Number of passes (Retrieval Scheme1).

Retrieval scheme2 sacrifices response time and hence the number of passes, for lower power consumption. Our experience showed that the number of passes increases greatly

for a small number of requests and reaches a maximum where the number of passes equals the number of channels. This is mainly due to the intelligence of the least-cost path method that only expands the path that has the smallest number of channel switches. As the number of user requests increases, the algorithm generates the access patterns according to the *Row Scan* method (Figure 14).

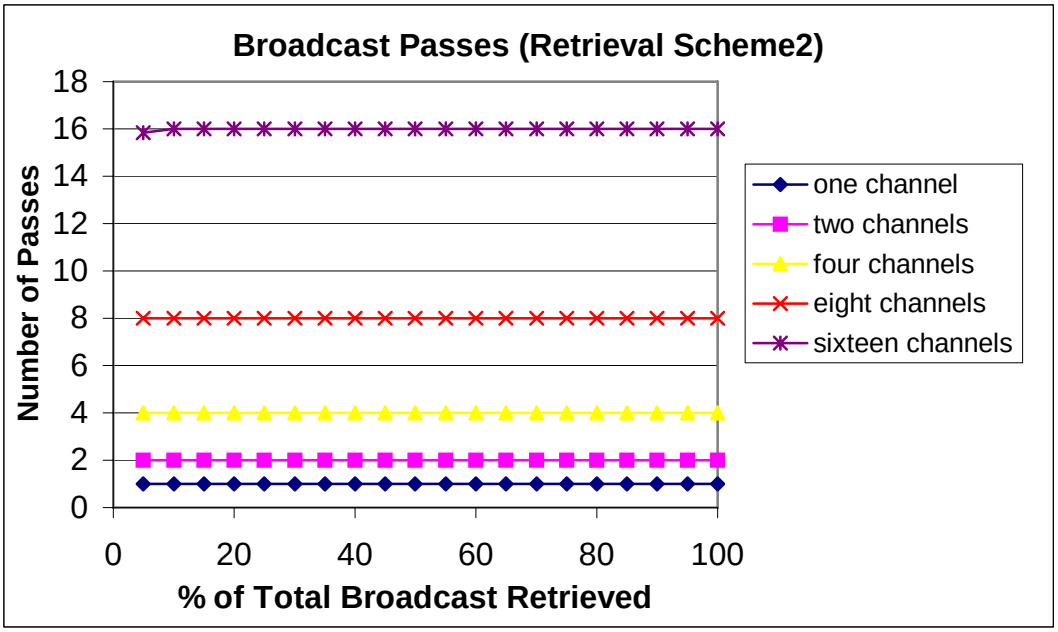

Figure 14: Number of Broadcast Passes (Retrieval Scheme2).

5. Conclusions and Future Directions

In this paper, we examined the problem of retrieving data elements from a parallel broadcast channel to make them available to mobile devices in pervasive computing environment. We proposed two retrieval schemes that utilize temporal and spatial information given by the index to reduce the energy consumption while pulling data elements.

We found that *Retrieval Scheme1* reduces the response time and the number of passes compared with the case in which *Row Scan* with index is used. Moreover the energy consumption was also reduced when a reasonable number of data elements was requested. The results also showed that the response time of a query decreases as the number of channels increases.

Retrieval scheme1 reduces the number of passes over the broadcast channels as the number of channels increases; however, this produced an increase in the number of conflicts and consequently, an increase in the switching frequency among channels and hence, higher energy consumption.

Retrieval Scheme2 aimed at an improved reduction of energy consumption, as well as a continued effort to improve the response time. The protocol makes use of a revised heuristics sequence that gives higher priority to the reduction of channel switches. In addition, the retrieval method makes use of the least-cost path search algorithm. The use of these features allow for only ideal paths to be expanded that maintain the minimal

amount of channel switches, and consequently require minimal energy consumption at the mobile device. The results of the simulations revealed that the introduction of the revised heuristics sequence had an adverse affect on the response time of the retrieval operation. The increased response time occurred as a result of an increased number of broadcast passes. The number of broadcast passes may be reduced through the use of channel switching; however, the revised heuristics prioritized the minimization of channel switches, thereby minimizing the ability to reduce the number of broadcast passes. It should be noted that in spite of the increased response time, application of the new heuristic sequence combined with the least-cost path algorithm drastically reduced the number of channel switches (see Figures 11 and 12) which results in much lower power consumption at the mobile unit.

For a large number of requests, the least-cost path search algorithm was able to successfully reduce the response time in comparison to the technique used in *retrieval scheme1*. For large user queries, the least-cost path search algorithm limits the number of broadcast passes so that it cannot exceed the number of channels being used. Limiting the number of passes also creates a limit on the increase in response time. Furthermore, since it allows only the ideal branches to be expanded, the least-cost path technique reduces the overall size of the access forest. As the number of data requests increases, the size of the possible access forest increases exponentially; the minimal tree size created by the least-cost path algorithm reduces the search time needed to determine the proper access patterns.

As a final note, it should be reminded that the proposed schemes are heuristic based and hence in some instances they may not generate efficient access patterns. For example, as depicted in figure 15, application of *retrieval scheme1* will requires four passes to retrieve the eight requested data elements. However, application of the *Row Scan* algorithm will pull the very same data elements by three passes over the parallel air channels. Our future research in this area will attempt to develop protocols that will always generate the minimum number of passes and channel switches [16].

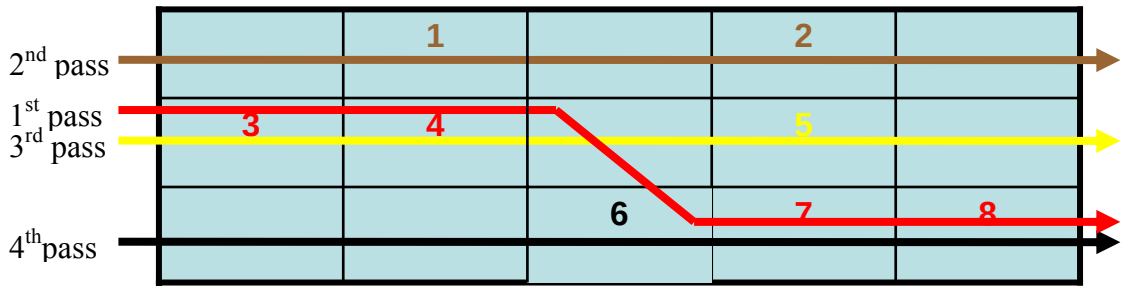

Figure 15: An example that shows inefficiency of the heuristic method.

References

[1] M.W. Bright, A.R. Hurson, S. Pakzad, "Automated Resolution of Semantic Heterogeneity in Multidatabases," *ACM Transactions on Database Systems*, Vol. 19, No. 2, 1994, pp. 212-253.

[2] Y. C Chehadeh, A. R. Hurson, L.L Miller, "Energy-Efficient Indexing on a Broadcast Channel in a Mobile Database Access System," *In The International Conference on Information Technology: Coding and Computing (ITCC'00)*, 2000.

[3] I.-R. Chen, F.B. Bastani, "Effect of artificial-intelligence planning procedures on system reliability," *IEEE Transactions on Reliability,* Vol. 40, No. 3, 1991, pp. 364-369.

[4] M.-S. Chen, K.-L. Wu, and P. S. Yu, "Optimizing Index Allocation for Sequential Data Broadcasting in Wireless Mobile Computing," IEEE Transactions on Knowledge and Data Engineering, Vol. 15, No. 1, 2003, pp. 161-173.

[5] Q. L. Hu, and D. L. Lee, "A Hybrid Index Technique for Power Efficient Data Broadcast," *In Distributed and Parallel Databases Journal,* Vol. 9, No. 2, 2001, pp. 151-177.

[6] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Data on Air: Organization and Access," *IEEE Transactions on Computers*, Vol. 9, No. 3, 1997, pp. 353-372.

[7] J. Juran, A.R. Hurson, and N. Vijaykrishnan, "Data Organization and Retrieval on Parallel Air Channels: Performance and Energy Issues," ACM Journal of WINET, Vol. 10, No. 2, 2004, pp. 184-195.

[8] J. B. Lim, A. R. Hurson, "Heterogeneous Data Access in a Mobile Environment- Issues and Solutions," *Advances in Computers*, Vol. 48, 1999, pp.119-178.

[9] A. Munoz-Avila and. A.R. Hurson, "Energy-Aware retrieval From Indexed Broadcast Parallel Channels," *Advanced Simulation Technology* Conference, 2003, pp. 3-8.

[10] NASDAQ World Wide Web Home Page, *http:\\www.nasdaq.com*, February 2002.

[11] N. Orchowski, and A.R. Hurson, "Energy-Aware Object Retrieval from Parallel Broadcast Channels," *Proceedings of the International Database Engineering and Applications Symposium,* IDEAS 2004.

[12] W.-C. Peng, J.-L. Haung, and M.-S. Chen, "Dynamic Leveling: Adaptive Data Broadcasting in a Mobile Computing Environment," Mobile Networks and Applications, 2003, pp. 355-364.

[13] E. Pitoura, and G. Samaras, "Data Management for Mobile Computing," Kluwer Academic Publishers, 1998.

[14] H. Qinglong, D.L. Lee, L. Wang-Chien, "A comparison of indexing methods for data broadcast on the air," *Twelfth International Conference on Information Networking (ICOIN-12),* 1998, pp. 656-659.

[15] M. Stemm and R.H. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices," *IEICE Transactions on Communications*, Vol. E80-B, No. 8, 1997, pp. 1125-1131.

[16] B. Sun, A.R. Hurson, and J. Hannan, "Energy-Efficient Scheduling Algorithms of Object Retrieval on Indexed Parallel Broadcast Channels", International Conference on Parallel Processing, 2004.

[17] R.W. Webster, "Useful AI tools – a review of heuristic search methods," *IEEE Potentials,* Vol. 10, No 3, October 1991, pp. 51-54.

[18] G.F. Wilber, "Strategic route planning using informed best-first search," *National Aerospace and Electronics Conference (NAECON 1988),* Vol. 3, 1988, pp. 23-27.

[19] R. Vlach, J. Lana, J. Marek, and D. Navara, "MDBAS – a prototype of a multidatabase management system based on mobile agents," In *Proceedings of SOFEM'00*, Springer-Verlag, 2000.