Indexing techniques for data broadcast on wireless channels

Dik Lun Lee & Qinglong Hu
Department of Computer Science,
The Hong Kong University of Science and Technology,
Clear Water Bay, Hong Kong
dlee@cs.ust.hk qinglong@cs.ust.hk

Wang-Chien Lee
GTE Laboratories Incorporated,
40 Sylvan Road, Waltham, MA 02454, U.S.A.
wlee@gte.com

Abstract

Indexing techniques have been developed as a means for clients to reduce power consumption and to select between broadcast and on-demand data services. In this paper, we provide an overview of our research on indexing techniques and revisit some related work in the literature. Our study incorporates two important techniques, clustering and scheduling, for improving data broadcast efficiency and explores the scenarios of single and multiple attribute query processing. Moreover, we apply two indexing methods, cache schedule and integrated signature, to a hierarchical data delivery system.

Keywords index, broadcast, on-demand, scheduling, clustering

1 Introduction

Wireless communication and mobile computing have gained much attention from the computer and communication research community. Wireless voice communication has already penetrated every modern society, and increasingly wireless data service is considered a standard feature of wireless communication systems.

Compared to voice communication, data services are more adaptive in the sense that they don't have a strict realtime requirement as voice communication, that they may be delivered purely by broadcast, purely by point-to-point communication or a mixed of both, and that they call for a set of distinct performance measures such as throughput, access time, and power consumption. Compared to wired networks, wireless communication opens up the possibility of truly simultaneous data broadcast (as opposed to multicast on Internet). Data access performance can be optimized by using and balancing broadcast and point-to-point communication methods. Despite these desirable features, however, a wireless system is limited in bandwidth, computational power, battery power, and physical size of the clients. These limitations have to be resolved in order for any application to be successful.

Data broadcast is efficient in terms of channel utilization when many users access to more or less the same set of data. It is power efficient because the clients consume much less energy when they are receiving than transmitting. Thus, many studies on data broadcast have appeared in the literature [IB93, AAFZ95, IV94, SRB97]. A weakness of data broadcast is that a client has to listen to the entire broadcast cycle to retrieve all data items it wants. Not only the access time, which depends on the length of the broadcast cycle, could be quite long, it also means that the client has to be active (i.e., consume battery power) in the entire cycle to get perhaps just a few data items from of the broadcast.

Indexing techniques for data broadcast have been introduced for reducing power consumption while keeping the access time small [IVB96, LL96, CYW97, SV96]. Among them, signature and index tree techniques [IVB96, LL96] are two representative indexing methods for broadcast channels. The basic idea behind these techniques is that, by including information about the arrival schedule of data frames in the broadcast channels, mobile computers are able to predict the arrival time of the requested data frames.

Three criteria are frequently used to evaluate the efficiency of wireless data services:

- *Access Time*: The average time elapsed from the moment a client makes a query to the moment when all the requested data frames are received by the client.

The 5th International Conference on Foundations of Data Organization (FODO'98), Kobe, Japan, November 1998.

- *Tune-in Time*: The time spent by a mobile computer staying active in order to acquire the requested data.
- *Indexing Efficiency*: The tune-in time saved per unit of access time overhead due to indexing.

While the access time measures the efficiency of data retrieval and organization methods for broadcast channels, the tune-in time is used to estimate the power consumption of a mobile computer. The objective of power conservative indexing is to reduce tune-in time, while maintaining an acceptable access time overhead, and the indexing efficiency measures how well an indexing method achieves that objective. When the power conservation is not a concern, the access time alone provides a convenient measurement of the efficiency of a wireless data service.

The rest of the paper is organized as follows. In Section 2, we give some background information regarding to the broadcast channel related techniques. In Section 3, we discuss various power efficient indexing techniques. In Section 4, we introduce a hierarchical data delivery system based on static hybrid channel allocation and the application of indexing techniques for the system. Finally, Section 5 concludes the paper.

2 Background Information

2.1 Mobile Computing Model

A geographical area covered by the wireless system is divided into *cells*. Wireless communication in a cell is supported by a *mobile support station* (MSS). MSSs are stationary computers connected to each other on a fixed network. An MSS is equipped with a wireless interface for communicating with the mobile computers in the cell[1]. In addition, the MSS also provides various data services to the mobile users. Logically, it maintains a database and provides data to its mobile clients. Each data item in the database is uniquely identified by its primary key which is a distinct number assigned by the MSS. The client access pattern is initially unknown to the MSS and is assumed to change with time.

We assume that the *basic fixed channel assignment strategy* [TJ91] is used. That is, a fixed set of channels is permanently assigned to each cell and a service request from a cell can only be served by unoccupied channels in that cell. Otherwise, the request is blocked. The communication is asymmetric (i.e., the uplink bandwidth is much less than that of downlink). The channels in the cell can work in one of the two modes:

- **On-demand mode:** the channel is bi-directional and asymmetric. The client explicitly makes a pull request to the server via uplink channel. Upon receipt of the request, the server makes connection with the client and return the required data through the connection. The connection is point-to-point. That is, once it is established between a client and the server, the client occupies the channel until the connection ends.
- **Broadcast mode:** the channel is uni-directional. The MSS periodically broadcasts data of popular demands to a large population of clients at the cell site. A complete broadcast of the set of data frames is called a *broadcast cycle*. To retrieve data item, the clients passively monitor the channel and download the data when they arrive.

On-demand and broadcast data services incur different access latency. On-demand service time is determined by the waiting time for the connection and the transmission time of the data. Since the queries are served on a request-by-request basis and the on-demand channels are occupied by one query service a time, a significant amounts of wireless bandwidth is required and it only performs well when the cell is lightly loaded. On the contrary, broadcast service is determined by the volume of data broadcast and the broadcast scheduling. Since data broadcast are shared among all the clients in the cell, wireless bandwidth is more efficiently utilized and the workload of the server is lower compared with on-demand data service. Broadcast service has a good performance when the cell is heavily loaded and there are common interest data set to the clients in the cell [HLL98d].

2.2 Channel Allocation Methods

Since broadcast and on-demand modes have different advantages for different system workloads, they can be used together to handle the variation of communication load over time and space due to the mobility of the clients. Generally, there are four channel allocation methods [LHL97, HLL99, HLL98d]. The *exclusive on-demand* and *exclusive broadcast* methods, respectively, assigned all channels in the cell to on-demand or broadcast mode. The *static hybrid* method assigns some of the channels to broadcast mode and some to on-demand mode, and the assignment is fixed in advanced. The *dynamic hybrid* method is an extension of the *static hybrid* method. It dynamically changes the assignment according to system workload and data access patterns. In this paper, we assume that a client gets data through either broadcast channels or on-demand channels but not both. That is, at any time the client is either monitoring the broadcast channels for the desired data to appear

[1] In this paper, we use 'mobile computer', 'mobile client', or 'client' to refer to a user with a mobile computer.

or making a pull request to the server for a point-to-point connection.

2.3 Techniques for Efficient Data Broadcast

In addition to indexing techniques, *scheduling* and *clustering* are two important data organizing techniques for improving data access efficiency. Data clustering refers to the consecutive placement of data items with the same value for a specific attribute in a broadcast cycle [IVB94, IVB96, HLL99]. By monitoring the arrival of the first data item with the desired attribute value, the client can retrieve all of the successive data items with the same attribute value instead of filtering for each arrival of the desired data items. However, a broadcast cycle can only be clustered based on one attribute. Although the other attributes are non-clustered in the cycle, a second attribute can be chosen to cluster the data items within a data cluster of the first attribute. Likewise, a third attribute can be chosen to cluster the data items within a data cluster of the second attribute. We call the first attribute the *clustered attribute* and the other attributes the *non-clustered attributes*.

For each non-clustered attribute, the broadcast cycle can be partitioned into a number of segments called *meta segments* [IVB96], each of which holds a sequence of frames with non-decreasing (or non-increasing) values of that attribute. Thus, when we look at each individual meta segment, the data frames are clustered on that attribute and indexing techniques designed for clustered data broadcast can still be applied within the meta segment. To facilitate our study, we define the number of meta segments in the broadcast cycle for an attribute as the *scattering factor* of the attribute. For multiple attributes, the broadcast cycle is partitioned into meta segments for each attribute in the order that frequently accessed attributes first. The scattering factors of an attribute increases as the importance of the attribute decreases. Organizing data broadcast with the above discussed clustering structure can improve retrieval efficiency.

Data scheduling determines the contents of the broadcast cycle and the broadcast frequency of the data frames [SRB97, AAFZ95, ST97] (i.e., the hot data set for broadcast and the relative broadcast frequency of each frame). A simple broadcast schedule, called *flat broadcast*, is to broadcast each data frame once every cycle [HLL98d]. With the flat broadcast, the expected access time for the data item on the broadcast channel is the same for all data items (i.e., half a broadcast period). In the real world, client access usually follows a skewed distribution (e.g., Zipf or Gaussian). It is unfair to have all data items the same access time regardless of their relative importance to the clients. Consequently, another scheduling method, called *broadcast disks*, was proposed in [AAFZ95].

Broadcast disks broadcast important data more often than the other data to reduce the average response time for queries based on that attribute value. The broadcast disk method has better access time when the data frames with the same attribute values are clustered in one of the minor cycles. By receiving the cluster of data frames together, the mobile computer can answer the query without continue to monitor the rest of broadcast cycle. This can be achieved by placing all of the data frames with the same indexed attribute value as a cluster on the same broadcast disk. The whole cluster of data frames are brought to the broadcast channel as a unit. Depending on speed of broadcast disk where this cluster is located, these data frames may appear several times in minor cycles. Thus, the resulting broadcast cycle is different from the completely clustered broadcast cycle. For broadcast scheduling adopting broadcast disks without using clustering, we simply consider the resulting broadcast cycle as non-clustered, in that case, broadcast disks loses its advantages compared with flat broadcast. Thus, when we consider index techniques for broadcast disks in the later sections of this paper, we only consider clustered case. However, broadcast disks increase the length of a broadcast cycle and the client has to spend more time retrieving the less commonly accessed data.

For multi-attribute data files, a cycle can be organized in broadcast disks based on one of attributes (i.e., the cycle is not clustered on the other attributes). Due to data frame duplication, queries other than that attribute may take longer to answer. Therefore, broadcast disks are not efficient for data files with multiple attributes. Thus, the flat broadcast scheduling is used for multi-attribute data files.

3 Power Conservative Indexing

It is expected that the lifetime of batteries will increase only 20% over the next 10 years [SCB92]. Thus, power conservation is a key issue for battery-powered mobile computers. Wireless data broadcast is efficient for disseminating data of common interest to a massive number of mobile users. Furthermore, indexing techniques can be employed to predict the arrival time of a requested data item so that the client can slip into *doze mode* and switch back to *active mode* only when the data of interest arrives, thus substantially reducing battery consumption.

Most of studies have focused on indexing techniques based on a single-attribute. In the real world applications, data items usually contain multiple attributes. Since broadcast channels are linear medium, comparing to single-attribute indexing and querying, data organization and access methods for multiple attribute indexes and queries are much more complicated. Thus, it's important to consider both of the single and multiple attribute queries when we adopt an indexing technique for the broadcast channels.

In the following, we present the indexing methods we proposed and the related techniques[2] existing in the literature. Our separate studies [HLL98c, HLL98e] have taken the single and multiple attributes queries into consideration. Due to the space limitation, we only illustrate the application of indexing techniques in the single attribute query centered environments.

When no index is used, a broadcast cycle consists of data frames only (called *non-index*). As such, the length of the broadcast cycle and hence the access time are minimum. Since every arriving frame must be retrieved into the client to check against the attribute values specified in the query, the tune-in time is very long and is equal to the access time.

3.1 The Hashing Technique

As mentioned previously, there is a tradeoff between the access time and the tune-in time of the system which can vary for different applications. Thus, we need flexible data organization methods capable of accommodating different classes of users. Among these indexing techniques, hashing based schemes and flexible indexing method were discussed in [IVB94].

In hashing based schemes, instead of broadcasting a separate directory with the data, the hashing parameters are included in the frames. Each frame has two parts: the *data* part and the *control* part. The control part is the "investment" which helps guide searches to minimize the access time and the tune-in time. It consists of a hash function and a shift function. The hash function hashes the query key attribute to compute the arrival of the desired frame; the shift function is necessary since most often the hash function is not perfect. In such a case there can be collisions and the colliding data items are stored immediately following the frame assigned to them by the hashing function.

Flexible indexing first sorts the data in ascending (or descending) order and then divides the file into p segments numbered 1 through p. The first frame in each of the data segments contains a control part consisting of the control index. The control index is a binary index which, for a given key K, helps to locate the frame which contains that key. In this way, we can reduce the tune-in time. The parameter p makes the indexing method flexible since depending on its value we can either get a very good tune-in time or a very good access time.

On selecting between the hashing scheme and the flexible indexing method, the former should be used when the tune-in time requirements are not rigid and the key size is relatively large compared to the record size. Otherwise, the latter should be used.

3.2 The Index Tree Technique

As with a traditional disk-based environment, index-tree methods [IVB96] have been applied to data broadcasts on wireless channels. Instead of storing the locations of disk records, an index tree stores the arrival time of information frames. The access protocol for retrieving information frames with an index tree technique involves the following steps:

- Initial probe: The client tunes into the broadcast channel and determines when the next index tree is broadcast.
- Search: The client accesses a list of pointers to find out when to tune into the broadcast channel to get the required frames.
- Retrieve: The client downloads all the information frames of interests.

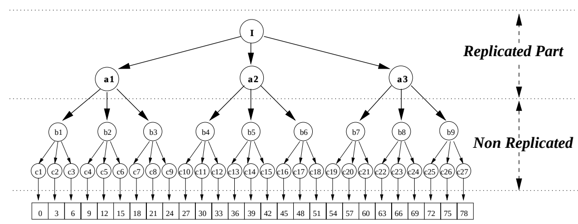

Figure 1: A Full Index Tree

Figure 1 depicts an example of an index tree for a broadcast cycle which consists of 81 information frames. The lowest level consists of square boxes which represent a collection of three information frames. The index tree is shown above the information frames. Each index node has three pointers[3].

To reduce access time while maintaining a similar tune-in time for the client, the index tree can be replicated and interleaved with the information. *Distributed indexing* is actually one index replication and interleaving method. The index tree is broadcast every $\frac{1}{d}$ of the file during a broadcast cycle. However instead of interleaving the entire index tree d times, only the part of the index tree which indexes the data block immediately following it is broadcast. The whole index tree is divided into two parts: replicated and non-replicated parts. The replicated part constitutes the upper levels of the index tree and each node in that part is

[2] Many of the techniques presented here are actually extended from their original proposals to co-operate the clustering and scheduling techniques.

[3] For simplicity, the three pointers of each index node in the lower most index tree level is represented by just one arrow.

replicated a number of times equal to the number of children it has, while the non-replicated part consists of the lower levels and each node in this part appears only once in a given broadcast cycle. Since the lower levels of an index tree take up much more space than the upper part (i.e., the replicated part of the index tree), the index overheads can be greatly reduced if the lower levels of the index tree are not replicated. In this way, access time can be improved significantly without much deterioration in tune-in time.

To support distributed indexing, every frame has an *offset* to the beginning of the root of the next index tree. The first node of each distributed index tree contains a tuple, with the first field as the primary key of the record that was broadcast last and the second field as the offset at the beginning of the next broadcast cycle. This is to guide the clients that have missed the required record in the current cycle to tune to the next broadcast cycle. There is a *control index* at the beginning of every replicated index to direct the client to a proper branch in the index tree. This additional index information for navigation together with the sparse index tree provides the same function as the complete index tree.

3.3 The Signature Techniques

Signature methods have been widely used for information retrieval. A signature of an information frame is basically a bit vector generated by first hashing the values in the information frame into bit strings and then superimposing them together [LL96]. They are broadcast together with the information frames in every broadcast cycle. A query signature is generated in a similar way based on the query specified by the user. To answer a query, the client simply retrieves information signatures from the broadcast channel and then matches the signatures with the query signature by performing a bitwise *AND* operation. When the result is not the same as the query signature, the corresponding information frame can be ignored. When the result is the same as the query signature, the information frame still need to be checked against the query. This step is to distinguish a *true match* from a *false drop*.

The signature technique interleaves signatures with their associated information frames. By checking a signature, the mobile computer can decide whether an information frame contains the desired information, if it doesn't, the mobile computer turns into doze mode and wakes up again for the next signature. The primary issue with different signature methods is the size and the number of levels of the signatures. The access protocol for a signature scheme involves the following steps:

- Initial probe: The client tunes into the broadcast channel for the first received signature.
- Filtering: The client accesses the successive signatures and data frames to find the required data.
- Retrieve: The client tunes in to get the successive desired data frames from the channel.

In [LL96], three signature algorithms, namely *simple signature*, *integrated signature*, and *multilevel signature*, were proposed and their cost models for access time and tune-in time were given. For simple signatures, the signature frame is broadcast before the corresponding information frame. Therefore, the number of signatures is equal to the number of information frames in a cycle. An integrated signature is constructed for a group of consecutive frames, called a *frame group*. The multilevel signature is a combination of the simple signature and the integrated signature methods, in which the upper level signatures are integrated signatures and the lowest level signatures are simple signatures.

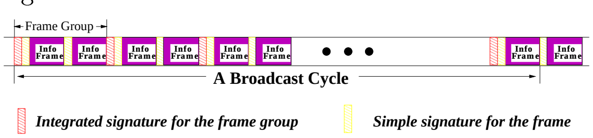

Figure 2: The Multi-level Signature Technique

Figure 2 illustrates a two-level signature scheme. The white signatures in the figure are integrated signatures. An integrated signature indexes all data frames between itself and the next integrated signature (i.e., two data frames). The black signatures are simple signatures for the corresponding data frames. In the case of non-clustered data frames, the number of data frames indexed by an integrated signature is quite small in order to maintain the filtering capability of the integrated signature. However, if similar data frames are grouped together in each meta segment (as in the case of clustered frames, we can regard the whole file as one meta segment), the integrated signature generated in this file organization has the same effect as reducing the number of bit strings superimposed. Therefore, the number of frames indexed by the integrated signature can be large.

3.4 The Hybrid Index Approach

Both the signature and the index tree techniques have advantages and disadvantages in one aspect or the other. For example, the index tree method is good for random data access, while the signature method is good for sequentially structured media such as broadcast channels. The index tree technique is very efficient for a clustered broadcast cycle, but the signature method is not affected much by clustering factor. While the signature method is particularly good for multi-attribute retrieval, the index tree provides a more accurate and complete global view of the data frames based on its indexed value. Since the clients can quickly search in the index tree to find out the arrival time of the desired data, the tune-in time is normally very short. Since a signature does not contain global information about the data frames; it can only help the clients to make a quick decision on whether the current frame (or a group of frames) is relevant to the query or not. The filtering efficiency heavily depends on the false drop probability of the signature. As a result, the tune-in time is normally high and is proportional to the length of the broadcast cycle.

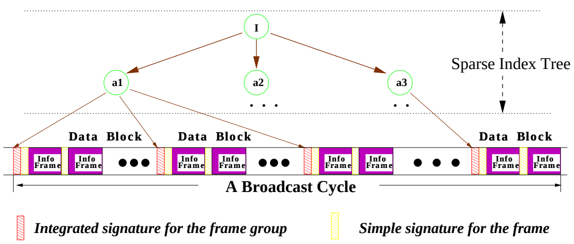

Figure 3: The Hybrid of Index Tree and Signature

In this section, we develop a new index method, called the *hybrid index*, which builds on top of the signatures a *sparse* index tree to provided a global view for the data frames and their corresponding signatures. The index tree is called sparse because only the upper t levels of the index tree (the replicated part in distributed index) are constructed. The *key-search pointer* node in the t-th level points to a *data block* which is a group of consecutive frames following their corresponding signatures. Figure 3 illustrates a hybrid index. To retrieve a data item, the client can search the sparse index tree to obtain the approximate location information about the desired data frames. Since the size of the upper t levels of an index tree is usually small, the overhead for this additional index is very small.

Since the hybrid index technique is built on top of signature method, it retains all of the advantages that a signature method has. However, the global information provided by the sparse index tree improves tune-in time considerably. The general access method for retrieving data with this technique now becomes:

- Initial probe: The client tunes into the broadcast channel and determines when the next index tree arrives.
- Search: Upon receipt of the index tree, the client accesses a list of pointers in the index tree to find out when to tune into the broadcast channel to get to the nearest location where the required data frames can be found.
- Filtering: At the nearest location, a successive signature filtering is carried out until the desired data frames are found.
- Retrieval: The client tunes into the channel and downloads all the required data frames.

4 Access Efficient Indexing

Mobile computing systems based on hybrid channel allocation methods facilitate a good balance of broadcast and on-demand channels. In addition, it allows the mobile users to use both kinds of channels to maximize their access efficiency. A hierarchical data delivery system has been proposed by the authors [HLL98b]. In this section, we introduce the indexing techniques studied for the system and related issues.

4.1 Hierarchical Data Delivery System

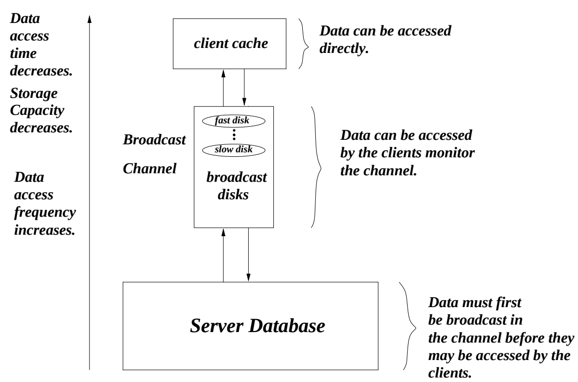

Figure 4: Hierarchical Data Distribution.

To minimize the client access time for a large population of clients, data caching, data broadcasting and pull-based requests can be used together. Logically we can assume that data are stored in a hierarchical medium (Figure 4) such that the most frequently accessed data are cached in the client, the commonly interesting data subset is temporarily stored in the air (broadcast on the channel), and the rest of the data can be pulled from server via explicit client requests. Data caching and push-based data dissemination can alleviate pull-based request considerably, as most frequently access data can be retrieved either from the client's cache or the broadcast channel.

The hierarchical data delivery system dynamically updates its broadcast schedule based on the client access patterns collected through a bit vector mechanism [HLL98a]. Based on the data access patterns, a medium sub-set of frequently accessed data is selected for broadcast on the channels and the broadcast program (i.e., broadcast disks) is designed accordingly.

When a user issues a query to the client, the client first search the cache. If there is a valid copy in the cache, an answer is replied immediately. Otherwise, the client attempts to obtain the data item from the server site. For a static hybrid channel allocation, the data access method depends on the existence of indexing methods.

Indexing mechanisms can facilitate the mobile clients of the hierarchical data delivery system to determine when to switch from the push-based broadcast channels to the pull-based on-demand channels. In the following, we discuss two indexing methods, namely, *cache schedule* and integrated signature, and their application in the hierarchical data delivery system. Though these methods are based on very simple ideas, we expect them to stimulate more studies on the indexing issues of the hierarchical data delivery systems.

4.2 Indexing Methods for Hierarchical Data Delivery

When no index is used, the client doesn't know which data frame is to appear next on the broadcast channels. Thus, pre-scheduling data access for broadcast mode and on-demand mode is impossible. The client first monitors the next *Threshold* number of data frames on the broadcast channels. When the desired data items are encountered, they are retrieved into the client cache. Otherwise, after monitoring *Threshold* data frames, the client turns to on-demand data service and issues a pull request to the server like in the exclusive on-demand channel allocation.

For cached schedule method, the complete broadcast schedule is broadcast at the beginning of each cycle, so that each active client keeps on monitoring the schedules and retrieves them into its cache. Based on the schedules, the client can make selection between broadcast and on-demand services. That is, if the desired data items will appear within the next *Threshold* number of data frames on the broadcast channels, the client monitors the broadcast channels. Otherwise, the client issues a pull request to the server immediately. Broadcasting schedule will take up bandwidth. The biggest drawbacks for this index method are that monitoring broadcast schedule is not energy efficient to the client and the requirement of all clients remaining active is not flexible.

When signature (i.e., integrated signature) is broadcast together with the data, the client first monitors the broadcast channels for the data items and the signatures. If the signatures retrieved indicate that the desired data item will be broadcast in the cycle and the number of data frames before the desired data to appear in the broadcasting cycle is less than *Threshold*, then the client continue monitoring the broadcast channels and retrieves the items into the cache when they arrive. Otherwise, the client issues a pull request to the server. Although, signature incurs some what index overhead, such overhead is low since the signature size is usually very small.

5 Conclusion

In a mobile environment, power conservation of the mobile clients is a critical issue to be addressed. An efficient power conservative indexing method should introduce low access time overhead, low tune-in time, and produce high indexing efficiency. As another application, indexing makes selection between push-based data dissemination and pulled-based data delivery possible. An ideal index method should perform well under both clustered and non-clustered broadcast cycle, with different broadcast scheduling policies, such as flat broadcast and broadcast disks.

Combining strengths of the signature and the index tree techniques, a hybrid indexing method is introduced in this paper. This method has the advantages of both the index tree method and the signature method and a better performance than the index tree method. A variant of the hybrid indexing method has been demonstrated to be the best choice for multiple attributes indexing organization in wireless broadcast environments [HLL98c].

Our studies of the indexing methods takes into consideration the clustering and scheduling factors which may be employed in wireless data broadcast. Access time, tune-in time, and indexing efficiency are the evaluation criteria for our comparisons. For both single attribute and multi-attribute index methods, cost models for access time and tune-in time of the three indexing methods were developed and numerical comparisons under various broadcast organizations based on the formulae were produced in [HLL98e, HLL98c].

In the future, we plan to incorporate the index schemes with data caching algorithms to achieve an improved system performance and obtain a better understanding of the wireless broadcast systems.

References

[AAFZ95] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: Data management for asymmetric communications environments. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, San Jose, California, 1995.

[CYW97] M.-S. Chen, P. S. Yu, and K.-L. Wu. Indexed sequential data broadcasting in wireless mobile computing. In *The 17th International Conference on Distributed Computing Systems (ICDCS97)*, Baltimore, Maryland, USA, May 27 - 30 1997.

[HLL98a] Q. L. Hu, D. L. Lee, and W.-C. Lee. Dynamic data delivery in wireless communication environments. In *Workshop on Mobile Data Access, in conjunction with 17th International Conference on Conceptual Modeling (ER'98)*, Singapore, November 1998.

[HLL98b] Q. L. Hu, D. L. Lee, and W.-C. Lee. Hierarchical data caching in asymmetric communication environments. In *Under Preparation*, 1998.

[HLL98c] Q. L. Hu, D. L. Lee, and W.-C. Lee. Multi-attribute indexing techniques for data on air. In *Submitted to the 6th International Conference on Database Systems for Advanced Applications (DASFAA'99)*, Taiwan, October 1998.

[HLL98d] Q. L. Hu, D. L. Lee, and W.-C. Lee. Optimal channel allocation for data dissemination in mobile computing environments. In *Proceedings of the 18th International Conference on Distributed Computing Systems (ICDCS98)*, pages 480–487, Amsterdam, The Netherlands, May 1998.

[HLL98e] Q. L. Hu, W.-C. Lee, and D. L. Lee. Power conservative indexing techniques for wireless data broadcast. *Submitted to Distributed and Parallel Databases: Special Issue on Mobile Data Management and Applications Wireless Data Broadcast (DPDB)*, October 1998.

[HLL99] Q. L. Hu, W.-C. Lee, and D. L. Lee. Indexing techniques for wireless data broadcast. In *Submitted to The Proceedings of the 19th International Conference on Distributed Computing Systems (ICDCS'99)*, Austin, Texas, USA, May 1999.

[IB93] T. Imielinski and B. R. Badrinath. Mobile wireless computing: Solution and challenges in data management. Technical Report DCS-TR-296, Detartment of Computer Science, Rutgers University, 1993.

[IV94] T. Imielinski and S. Viswanathan. Adaptive wireless information systems. In *proceedings of SIGDBS (Special Interest Group in DataBase Systems) Conference*, Tokyo - Japan, October 1994.

[IVB94] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Power efficiency filtering of data on air. In *Proceedings of the International Conference on Extending Database Technology*, pages 245–258, 1994.

[IVB96] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Data on the air - organization and access. *IEEE Transactions of Data and Knowledge Engineering*, July 1996.

[LHL97] W.-C. Lee, Q. L. Hu, and D. L. Lee. Channel allocation methods for data dissemination in mobile computing environments. In *Proceedings of the 6th IEEE International Symposium on High Performance Distributed Computing*, pages 274–281, August 1997.

[LL96] W.-C. Lee and D. L. Lee. Using signature techniques for information filtering in wireless and mobile environments. *Special Issue on Database and Mobile Computing,Journal on Distributed and Parallel Databases*, 4(3):205–227, July 1996.

[SCB92] S. Sheng, A. Chandrasekaran, and R.W. Broderson. A portable multimedia terminal for personal communications. *IEEE Communications Magazine*, pages 64–75, December 1992.

[SRB97] K. Stathatos, N. Roussopoulos, and J. S. Baras. Adaptive data broadcast in hybrid networks. In *Proceedings of the 23rd VLDB Conference*, pages 326–335, Athens, Greece, 1997.

[ST97] C.J. Su and L. Tassiulas. Broadcast scheduling for information distribution. In *Proceedings of IEEE INFOCOM'97*, April 1997.

[SV96] N. Shivakumar and S. Venkatasubramanian. Engery-efficient indexing for information dissemination in wireless systems. *ACM-Baltzer Journal of Mobile Networks and Nomadic Applications (MONET)*, December 1996.

[TJ91] S. Tekinay and B. Jabbari. Handover and channel assignment in mobile cellular networks. *IEEE Communications Magazine*, pages 42–46, November 1991.