A Comparison of Indexing Methods for Data Broadcast on the Air

Qinglong Hu & Dik Lun Lee
Department of Computer Science
University of Science and Technology
Clear Water Bay, Hong Kong
qinglong@cs.ust.hk dlee@cs.ust.hk
FAX: (852) 2358-1477

Wang-Chien Lee
GTE Laboratories Incorporated
40 Sylvan Road
Waltham, MA 02254
wlee@gte.com
FAX: 617-466-3231

Abstract

Several indexing techniques for data broadcast on the air have been proposed for power conservation on mobile computers in the past few years. Indexing techniques for broadcast channels can save battery power (estimated by tune-in time) while incurring only limited overhead on access time. In this paper, we compare indexing techniques based on the index tree and the signature methods and find that both methods prevail under different circumstances.

1 Introduction

Efficient utilization of wireless bandwidth and battery power are two of the most important problems facing software designed for mobile computing. Broadcast channels are attractive in tackling these two problems in wireless data dissemination. Data disseminated through broadcast channels can be simultaneously accessed by an arbitrary number of mobile users, thus increasing the efficiency of bandwidth usage. In addition, mobile computers consume less battery power on monitoring broadcast channels to receive data than accessing data through point-to-point communications. To estimate the efficiency of the broadcast channels, two criteria are frequently used [1]:

- *Access Time*: The average time elapsed from the moment a client makes a query to the moment when all the requested data items are received by the client.
- *Tune-in Time*: The period of time that a mobile computer remains active in order to receive the requested data.

While the access time represents the efficiency of the access method and data organization for broadcast channels, the tune-in time correlates to a mobile computer's power consumption. How to save tune-in time with acceptable access time delay is the goal pursued in our research.

Various indexing techniques have been proposed in the literature to address the power conservation issue of broadcast channels [1, 2]. These indexing techniques are based on the common idea that, by adding some auxiliary information to data items on broadcast channels, mobile computers will be able to predict the forthcoming of the requested data items. By programming the mobile computers to stay in doze mode most of time and only wake up to receive data when it's coming, tune-in time may be significantly reduced. As a result, battery power is conserved.

Signature and index-tree methods represent two classes of index methods for broadcast channels. Although performance evaluation for each class of index methods has been conducted individually, there is no study in the literature comparing both classes of the index methods. In this paper, we propose a simple benchmark to compare these two indexing techniques.

The rest of the paper is organized as follows. In Section 2, we give an informal introduction of the broadcast channels, indexing techniques and system parameters used in performance evaluation and comparisons. In Section 3 and Section 4, indexing methods based on index-tree and signature techniques are introduced. In Section 5, we evaluate the indexing methods based on our benchmark and compare their efficiency in terms of tune-in time and access time. Finally, Section 6 concludes the paper.

2 Index on the Air

To support broadcast channels, the base station sends out a series of *information frames*. To receive the frames off the air, mobile users specify queries on the frames (e.g., by specifying a text pattern on a text field). The mobile computers will monitor the broadcast channels and present to the users frames matching the queries. Since the information frames are periodically broadcasted, a complete broadcast of the information frames is called a *broadcast cycle*. From the user's viewpoint, the broadcast information is perceived as a stream of frames flowing along the time axis. Logically, there is no specific start and end frames for a broadcast cycle; a broadcast cycle starts with any frame and ends when the frame appears again. Updates to the file are reflected only between successive broadcasts.

In this paper, we assume that all information to the clients, including data and auxiliary index information, is sent on a broadcast channel. We assume only one channel for broadcasting, because a channel with large bandwidth is logically the same as multiple channels with combined bandwidth of the same capacity. Moreover, we assume information broadcast is *clustered* on an attribute, in which all data items with the same value under the clustering attribute appear consecutively in each broadcast cycle.

As in the traditional disk-based environment, indexing tree techniques have been applied to data broadcast on a wireless channel. In addition to broadcasting data, the base station broadcasts index trees that indicate the point of time when a particular data item is broadcasted. Queries are answered by searching the index tree. Once the broadcast schedule of the desired data items is found, the clients will turn into doze mode until the required data items arrive on the broadcast channel.

Signature methods have been widely used for information retrieval. A signature S_i of a data item is basically a bit vector generated by first hashing each data value into a bit string and then superimposing the bit strings together [2]. By examining the signature only, the client can estimate whether the data item contains the desired information. During filtering, a query signature, S_Q, is constructed in the same way as the data signature and then compared to the data signatures. The three possible results are:

- The data item matches the query;
- The data item doesn't match the query;
- The data item doesn't match the query but the signature comparison indicates a match.

In the first and last cases, the query and data signatures match (i.e., $S_Q \wedge S_i = S_Q$). We call the first case *true match* and the last case *false drop*. To eliminate false drops, the data item must be compared directly with the query after the data signature signifies a match. A data signature failing to match the query signature (i.e., $S_Q \wedge S_i \neq S_Q$) guarantees that the corresponding data item can be ignored. In that case, the client can remain in doze mode and wake up again when the next signature arrives.

In a mobile environment, both signature and index tree techniques provide the following advantages for information filtering:

- They can improve the tune-in time, with limited delay on the access time.
- They are very easy to generate and search; thus, they are suitable for mobile computers with limited computing power.

Comparing the design principles behind the index tree and signature methods, we find that the index tree maintains a global view of the data items it indexes. The index tree can be efficiently traversed to find out the arrival time of the data items. Thus, the mobile computer can accurately predict the arrival time of the desired data items and thus stay in doze mode most of the time. On the other hand, the signature method does not maintain a global location information of data items. It helps the mobile computer to make quick decisions on whether a data item is relevant to a given query. Also, the linear structure of signatures is suitable for broadcast channel.

In the following, we define the parameters to be used in the cost models of the index tree and signature methods.

Definition of Symbols and Parameters

F: the number of information frames in a broadcast cycle.
P: the number of packets in an information frame.
S: the selectivity of the attribute, defined as average number of frames containing the same attribute value.
h: the height of the whole index tree.
t: the number of top levels of an index tree, e.g., part of the index tree which is replicated.
n: the number of search keys and pointers a frame holds.
X: size of the full index tree (number of frames).
$X[t]$: size of the top t levels of the index tree.
$L[t]$: the number of nodes on the t th level of the index tree.
P_f: false drop probability.
P_f^s: false drop probability for simple signatures.
P_s: selectivity of a query ($S = P_s \cdot F$).
p: the number of bits in a packet.
R_s: the number of packets in a simple signature.
s: the number of attributes in a record.

Frame is the logical unit of information in broadcasting, while packet is the physical unit of broadcasting. Frame size will be equal to some multiple of the packet size. To facilitate our analysis and evaluation, we assume the information frames have the same size and use it for measurement of both access time and tune-in time.

3 Filtering with Index Tree Techniques

The access protocol for retrieving data items with index tree technique involves the following steps,

- Initial probe: The client tunes into the broadcast channel and finds out when the next index tree frame for the data item will be broadcasted.
- Search: Access index tree to find out when to tune into the broadcast channel to get the required data. The number of pointers accessed will be equal to the height of the index tree.
- Retrieve: Tune into the data frame containing the required data and download all the matched data.

When only data is broadcast (hereafter denoted as *non-index*), the optimal access time can be achieved. We use this special case as a benchmark for comparisons between various indexing methods. In [1], two index tree methods, distributed index and (1,m) index, were proposed and compared. In the following section, we briefly introduce the distributed index method and present the cost model. Interested readers can refer to [1] for more details.

The distributed index is an index replication method in which only the relevant portion of the index tree is interleaved with the data in every fraction ($\frac{1}{m}$) of the broadcast cycle. It replicates some upper levels of the index tree to provide a global view of the divided fractions of the broadcast cycle and use lower level subtrees for navigation within the fractions.

Distributed index takes an index tree and multiplexes it with the data by subdividing the tree into two parts: replicated part and non-replicated part. The replicated part constitutes the top t levels of the index tree and each node in that part is replicated a number of times equal to the number of children it has, while the non-replicated part consists of the bottom $(h-t)$ levels and each node in this part will appear only once in a given broadcast cycle. Since the low levels of an index tree will take up much more space than the upper part, not replicating that part will reduce access time. Based on the above definition of the distributed index tree, the access time and tune-in time can be obtained as:

$$\begin{aligned} ACCESS &= \frac{1}{2} * (\frac{X - X[t]}{L[t+1]} + \frac{F}{L[t+1]} + X + F \\ &\quad + L[t+1] - 1) + S \\ TUNE &= 2 + h + S \end{aligned}$$

In order to minimize the access time, the optimal number of replicated levels for a balanced index tree, denoted by $\hat{t}$, can be set as:

$$\hat{t} = \lfloor \frac{1}{2} * (log_n(\frac{F}{S} + \frac{n^{h+1}}{n-1}) - 1) \rfloor + 1$$

4 Filtering with Signature Techniques

The access protocol for signature scheme has the following steps:

- Initial probe: The client tunes into the broadcast channel and determines when the first signature for data frame will be broadcasted.
- Search: Access the successive signature and data frames to find the required data. On the average, the client will take half the broadcast cycle to get the first data with the required attribute.
- Retrieve: The client tunes in to get the successive desired data items from the channel.

The number and the size of the signatures and the false drop probability of the signatures will affect the tune-in time and access time of the signature scheme. The false drop probability may be controlled by the size of the signatures. The initial probe time is related to the number of signatures interleaved with the information frames.

Generally the length of a signature is very short compared to an information frame, the period in which the mobile client is active during the initial probe time is very small compared with the tune-in time for filtering of false drops and true matches, it can be approximated as half of the frame size. Lemma 1 gives the false drop probability in the signature technique [2].

LEMMA 1 (optimal false drop probability) *Given the number of packets in a signature, R_s, and the number of bit strings superimposed into the signature, s, the false drop probability for the signature is: $P_f = 2^{-R_s \cdot (p \cdot ln2)/s}$*

In [2], three signature algorithms, namely simple, integrated and multi-level signature schemes, were proposed and their cost models for the initial probe time, access time and tune-in time were given. In this paper, we compare the simple signature scheme for clustered data with the distributed index scheme. We assume the clients will always give priority to achieving optimal tune-in time. In the following part, we give a brief introduction to the simple signature scheme and develop an approximate method to obtain the optimal value of R_s of the signature scheme based on the cost model in [2].

In the signature method, the signature frame is broadcasted before the corresponding information frame, as shown in Figure 1.

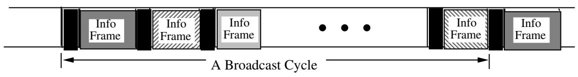

Figure 1. Simple Signature Technique.

The access time and tune-in time are:

$$ACCESS = (F+1) \cdot (R_s/P + 1)/2 + S$$

$$TUNE \approx \frac{1}{2} + \frac{F \cdot R_s}{2P} + \frac{F \cdot (1 - P_s) \cdot P_f^s}{2} + S \quad (1)$$

According to Lemma 1, we have, $P_f^s \approx 2^{-(p \cdot ln2/s) \cdot R_s}$. In order to evaluate the minimum tune-in time of the simple signature scheme, we differentiate (1) with respect to R_s and let $\partial TUNE_s / \partial R_s$ equal zero. The optimal $\hat{R}_s$ can be computed as:

$$\hat{R}_s = s/(p \cdot ln2) \cdot log_2(((1 - P_s) \cdot P \cdot p \cdot ln^2 2)/s) \quad (2)$$

5 Comparison of Signature and Index Tree

In this section, we compare the discussed indexing methods in terms of their access time and tune-in time. Our comparisons are based on the formulae given in the previous sections. Table 1 lists the parameter values used in the comparisons. Several assumptions are made: information frames are clustered, there is only one attribute in the data records, the selectivity of query is one, the index tree is balanced (all leaves are on the same level), and each node has the same number of children.

$p = 128$	$P = 1000$	$s_s = 1$
$s_i = 280$	$k = 4$	$l = 3$
$P_s = 0.01$	$S = 1$	$n = 10$

Table 1. Parameters of the cost models.

Figure 2 compares the access time of the two index methods with the non-index method. We can see that the signature method has similar performance as non-index method. That means the signature method does give us very good access time. On the other hand, the index tree method has worse performance. The access time of each scheme is proportional to the length of the broadcast cycle, with the index tree method increasing much faster than the signature method.

In Figure 3, the tune-in time of the non-index, index tree and signature method is shown. We can see that the tune-in time of non-indexed method is obviously much higher than any of the index schemes. It is certain that indexing methods can improve tune-in time considerably. If we only consider the signature scheme in Figure 3, we find that they are proportional to the length of the broadcast cycle. On the other hand, the indexing tree scheme shows almost the same performance regardless of the length of the broadcast cycle, which is much better than signature method. This is because the size of the index tree can be adjust automatically according to the length of the broadcast cycle F and the height of the index tree h, the only factor affecting the tune-in time, increases very slowly ($n^h \approx F$). As a result, the length of the broadcast cycle has a very small influence on tune-in time and is negligible when compared to that of the signature method.

Generally index tree techniques do not support multi-attributes index well and are good for random data access which is not the characteristics of a broadcast channel where data can only be accessed sequentially. That is the reason that the access time is higher than that of the signature techniques. Also, index tree techniques are good for clustered data file. For non-clustered data file with high scattering factor, they don't perform well.

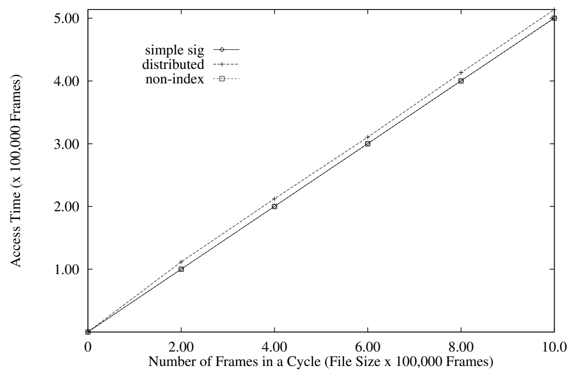

Figure 2. Access Time Comparison.

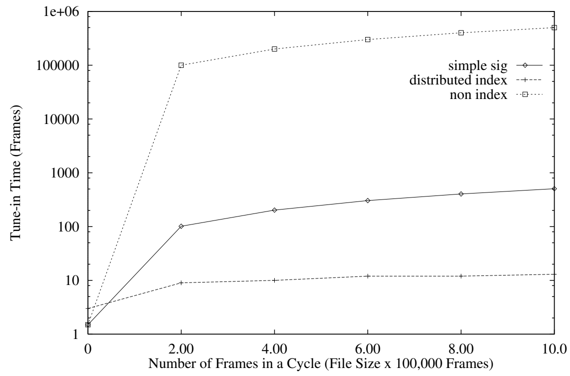

Figure 3. Tune-in Time Comparison.

6 Conclusion

The ideal index technique should seek optimum in both access time and tune-in time. However, these two optimum goals are contradictory to each other. We found that index techniques can improve tune-in time considerably with acceptable access time overhead. Both the index tree and signature techniques have some advantages over the other. For example, index tree gives more precise global location information of the data items and the tune-in time is very short. The signature techniques, on the other hand, can be applied to various types of information media, the tune-in time is not as good as the index tree methods, but the access time overhead is small. To achieve a given tune-in time, the access time overhead for index tree methods is much greater than the signature methods. Therefore, if the tune-in time is not the most important factor, signature techniques are better than index tree techniques.

As for the future work, a study for the performance of indexing built upon non-clustered data file will be conducted. New index techniques combining index tree and signature methods can be developed. Furthermore, we can apply caching strategies to improve the performance of indexing on data file.

References

[1] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Energy efficiency indexing on air. In *Proceedings of the International Conference on SIGMOD*, pages 25–36, 1994.

[2] 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.