Using Signature and Caching Techniques for Information Filtering in Wireless and Mobile Environments

Wang-chien Lee
Dept of Computer and Information Science
The Ohio State University
Columbus, Ohio 43210-1277, USA

wlee@cis.ohio-state.edu
FAX: 614-292-2911

Dik Lun Lee
Department of Computer Science
University of Science and Technology
Clear Water Bay, Hong Kong

dlee@cs.ust.hk
FAX: (852) 2358-1477

Abstract

This paper discusses the issue of power conservation on personal computing/communication systems (PCSs) and suggests that signature methods are suitable for realtime information filtering on PCSs. Three signature-based approaches, namely simple signature, integrated signature and multi-level signature schemes are presented. The cost models for the access time and tune-in time of these three approaches are developed. We have shown that the multi-level signature method is in general better than the other two methods. Based on the multi-level signature scheme, we further study four caching policies, namely, BA, BP, VA and VP. The cost models for tune-in time of the four policies are derived and their performance is compared based on a number of factors.

1 Introduction

Rapid advances in wireless data networks and personal computing have opened up new services to mobile users. Various commercial and experimental personal computing/communication systems (PCSs) have appeared recently [1,11]. It is envisioned that PCSs will be as popular as walkmans and portable TVs in the near future and promise to revolutionize the information service market. In a wireless communication environment, mobile users with PCSs are free to access information services and communicate with other users without geographical limitations. To meet the market needs, ubiquitous services, such as fax-oriented messengers, nomadic conferencing and computing, mail-triggered applications, and information broadcasting, will emerge as some of the most important research topics in the next decade [12]. In this paper, we consider the problems with information broadcast services for mobile users equipped with battery powered PCSs.

The application of information broadcast is numerous. An example is mobile shopping, where product information is broadcast continuously and PCS users can specify the products they are interested in, browse through the information provided (e.g., sample pictures, prices, and so on), and order the merchandises with a few key strokes. This example can be extended to shopping malls, where information such as product categories, locations of stores, special sales can be provided to customers on the air. Experimental shopping assistant systems based on these ideas have been proposed and developed [4].

We may consider two modes of operations in information broadcast applications. For PCSs with transmit capability, the PCSs will send the user queries to the central server, which collects the queries over a period of time and broadcasts the requested information on the channel. The PCSs are responsible for identifying the information they need from the channel. For PCSs without transmit capability, the central server will broadcast all of the information available on the channel in a certain order, the PCSs listen to the channel and select the information requested by the users. It is noted in either case that information from the central server is broadcast on a single channel because allocating a dedicated channel to each individual PCS is too expensive and that the PCSs are required to filter out unwanted information from the channel so as to reduce the amount of information presented to the users and to reduce battery consumption by the PCSs. In this paper, we investigate indexing and caching techniques for filtering broadcast information on the air with the assumption that the PCSs have no transmit capability.

1.1 Power Conservation and Information Broadcasting

A major problem with PCSs is their power supplies. Batteries are the main power source in most PCSs. In order to make PCSs portable, small batteries, such as AA or AAA batteries, are likely to be used [5]. However, these batteries have small capacity and need recharging or replacement after a short period of usage. Although processors and memories consuming less power have been developed (e.g., the Hobbit chip from AT&T and energy efficient chip designs at Berkeley [7]), new generations of faster chips with high clock frequency will continue to demand more and more power. Therefore, power conservation is an important issue for applications on mobile computing environment.

There are two factors affecting power consumption in PCSs: (1) PCSs can be switched between *active* mode (full power) and *doze* mode [10], and (2) receiving messages consumes less energy than sending messages. Methods have been proposed for energy efficient query optimization [3] and information broadcasting [13,14].

In the wireless environment, broadcasting is an attractive method of disseminating of information to mobile users because base stations are equipped with powerful communication equipments but mobile PCSs, restricted by cost, portability and power, can afford little or no transmission capability. Moreover, broadcasting can scale up to an arbitrary number of users. In contrast to accessing traditional storage media, the performance of accessing information on air does not degrade as the number of users increases. In this paper, we consider broadcasting-based applications.

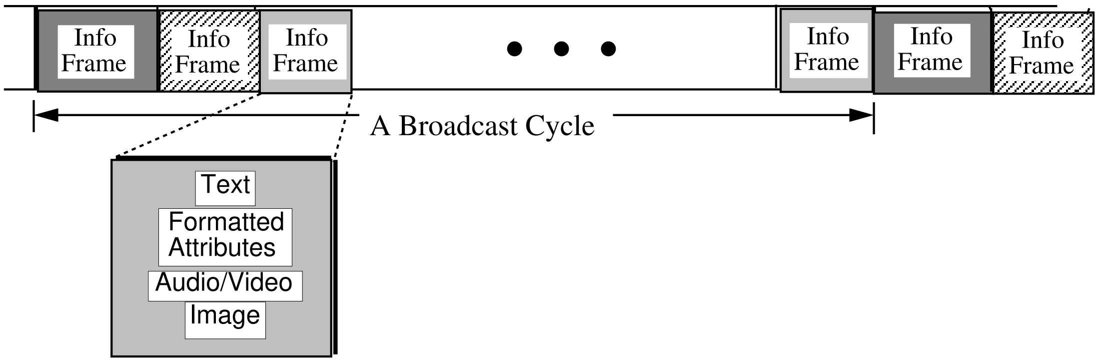

Figure 1: Information Stream.

In broadcasting, the base station sends out a series of *information frames.* (See Figure 1.) An information frame is a logical unit of information broadcasted on the air and may consist of multimedia information, including text, image, audio/video and other related data. Frames may vary in size; they consist of *packets* which are the physical units of the broadcasting. A frame contains a header (not shown in the figure) for synchronization as well as meta-information indicating the type and length of the frame. At the receiving end, users are allowed to specify conditions on the frames they are interested in. The PCSs will only present to the users frames matching the conditions. 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. In a broadcast cycle, some important information frames may be replicated (i.e., frames with the same contents but treated as different frames). Information frames may be inserted, deleted, and modified. The updates will be reflected in the subsequent broadcast cycles.

The duration that a PCS must stay in active mode to answer a query is called the *tune-in time*, which is proportional to power consumption. *Access time* is the time required to collect all qualified frames. Without any access aid, both the tune-in time and access time are equal to the length of the broadcast cycle, because it is necessary to scan through all of the frames in a broadcast cycle to pick up the qualified frames. This is very inefficient in power consumption, because typically only a few frames in a cycle satisfy the user request.

Access methods can be developed so that the PCS can be turned off when the frame being broadcasted is not qualified. By switching between active and doze modes, power consumption is reduced. In order to tell which frames would qualify ahead of time, auxiliary information about the contents of the frames must be added. Due to the limited number of broadcast channels available, we assume that only one channel is used for both primary and auxiliary information. With only one channel, the auxiliary information will increase the length of a broadcast cycle and thus increase access time. However, it will reduce tune-in time, because it allows the PCSs to avoid tuning into unwanted information frames. Thus, we must tradeoff between access time and tune-in time when we consider what auxiliary information is to be used and how it is going to be organized.

Two approaches, namely, hashing and indexing [13,14], have been proposed in the literature for encoding auxiliary access information for wireless broadcasting information services. Organizations and algorithms for disseminating and retrieving data on air were proposed. Tradeoffs between power consumption and access time were also considered. However, indexes based on one single key were assumed in these studies [13,14]. Therefore, they won't support general queries involving various attributes of the information frame.

A major difference between traditional disk-based indexing techniques and indexing broadcast data is that disk-based indexing allows random access to data whereas data on a broadcast channel must be accessed sequentially. This property significantly changes the cost factors of indexing techniques. For instance, tree structures are fast on disks because they allow random access from node to node, thus bypassing nodes containing irrelevant data. However, on a broadcast channel, tree structures lose this advantage because it takes time for the PCS to skip the irrelevant data. The focus of this paper is on the application of the signature file technique, to be described next, on indexing broadcast information. Signature file techniques are known to be slow compared to tree structures. However, in a broadcast channel, they become very attractive as tree structures lose their advantage in speed. On the other hand, the simplicity of the signature file makes it highly suitable for realtime information filtering under stringent processor speed and memory size.

1.2 The Signature Technique

Signature methods have been used extensively for text retrieval [9], image database [16], multimedia database [17,19] and other conventional database systems [8]. A signature is basically an abstraction of the information stored in a record or a file. By examining the signature only, we can estimate whether the record contains the desired information. Naturally, the signature technique is very suitable for filtering information frames in a wireless broadcasting environment.

A signature of a record is formed by first hashing each value in the record into a random bit string and then superimposing together all bit strings generated from the record into the record signature. During filtering, a query signature is constructed in the same way and then compared to the record signatures. There are three possible outcomes of the comparison: (1) the record matches the query; that is, for every bit set in the query signature, the corresponding bit in the record signature is also set (i.e. $S_Q \wedge S_i = S_Q$); (2) the record doesn't match the query (i.e. $S_Q \wedge S_i \neq S_Q$); and (3) the signature comparison indicates a match but the record in fact does not match the search criteria. The last case is called a *false drop.* To eliminate false drops, the record must be compared directly with the query after the record signature signifies a match. A signature failing to match the query signature guarantees that the corresponding record can be ignored. Signature techniques are good at screening out unqualified records. Figure 2 depicts the signature generation and comparison processes of a `company` record having two attributes, `name` and `type`.

In a mobile environment, the signature technique offers the following advantages for information filtering:

- Signature techniques may be generally applied to various types of information media.
- Signature techniques are particular good for multi-attribute retrieval, which is necessary for specifying precise filtering conditions.
- Signatures are very easy to generate and search; thus, they are suitable for PCSs where realtime searching with limited buffer space is required.
- A signature is very short compared to an information frame; therefore, the access time won't be increased drastically.
- A signature file is basically a sequential file structure. This makes it easy to "linearize" the signature file for broadcasting on air and scanning by a PCS, whereas a tree based access structure will lose the speed advantage because random access cannot be done on a broadcast channel.

The rest of the paper is organized as follows. In Section 2, three signature schemes for information indexing and filtering are presented; their performance is evaluated in terms of access time and tune-in time. Section 3 presents four caching policies for the multi-level signature scheme and provides performance analysis based on tune-in time. Section 4 is a review of related work. Finally, Section 5 concludes the paper.

2 Information Broadcasting Using the Signature Technique

Signatures are constructed from the information frames and broadcasted together with the information frames. The signatures may be broadcasted as a group before the information

record | IBM | Computer |

attribute signatures:	
IBM	001 000 110 010
Computer	000 010 101 001
record signature ($\vee$)	001 010 111 011

Queries	Query Signatures	Results
1) IBM	000 010 101 001	$\leftarrow$ match
2) GE	010 001 000 011	$\leftarrow$ no match
3) Ford	001 000 111 000	$\leftarrow$ false drop

Figure 2: Signature generation and comparison.

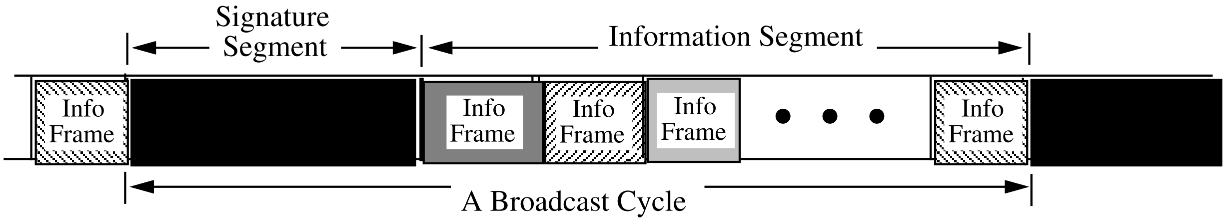

Figure 3: Non-interleaved Signatures.

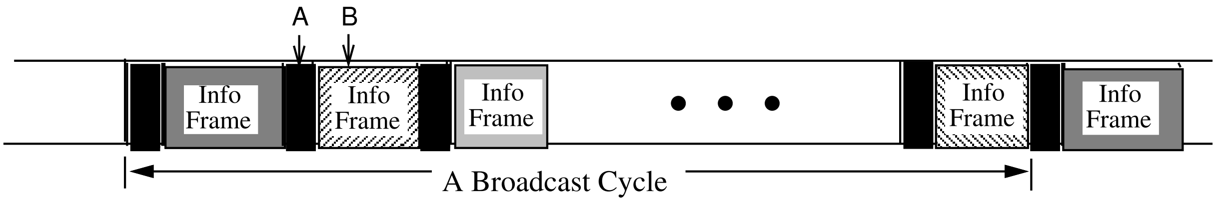

Figure 4: Interleaving Signatures.

frames or interleaved with the corresponding frames. Figures 3 and 4 illustrate these two approaches. For the non-interleaved signature approach, since the user may start monitoring the broadcast channels at any moment, missing the signature segment means the user has to wait until the next broadcast cycle to access the signatures. The period of time from the moment a user tunes in until the first signature is received is called *the initial probe time*. Obviously, the non-interleaved method is undesirable, because it results in a longer initial probe time. Thus, we only consider interleaving approaches for the signature schemes discussed in this paper. Noted that during the initial probe time, the user may choose to switch to doze mode until a signature is encountered or to remain in active mode to scan for qualified information frames without the help of signatures. The former will save energy, while the latter may return qualified information frames earlier. Since the focus of the paper is on energy saving, we will assume the PCS stays in doze mode during the initial probe time throughout this paper.

Different schemes may be used to organize signatures and information for broadcasting. In this paper, we discuss three signature methods based on interleaving.

2.1 Simple Signature Scheme

The most intuitive approach for interleaving signatures with information frames is to construct a *frame signature* for each information frame. The signature frame is broadcasted before the corresponding information frame (see Figure 4).

When a mobile user wants to retrieve information from the broadcast channel, she/he specifies a query on a PCS. A query signature S_Q is generated based on the specified query. Then the PCS tunes into the channel and uses S_Q to compare with the frame signatures received. When a match is found, the corresponding information frame is received by the

PCS for further checking in order to eliminate false drops. If the frame is not a false drop, it will be retained in the result set. When a received frame signature does not match with the query signature, the PCS will switch into doze mode until the next signature frame arrives. If most of the frame signatures don't match with the query signature, the PCS will stay in doze mode for the most part of a broadcast cycle, thus saving a lot of energy.

In this scheme, the average initial probe time is half of the average size of an information frame and its signature frame. The access time and tune-in time, however, are dependent on the positions of the initial probe:

position A: If the initial probe falls in the middle of a signature frame (point A in Fig. 4), the PCS has to stay active for the rest of the signature frame in order to detect the beginning of a new frame. Then, the PCS switches to doze mode. The initial probe time, access time and tune-in time are as follows.

> **Initial probe time** = length of the partial signature scanned + the length of the first information frame.
>
> **Access time** = initial probe time + a broadcast cycle.
>
> **Tune-in time** = length of the partially scanned signature + every signature in a broadcast cycle + false drop and true drop information frames in the cycle.

position B: If the PCS initially tunes into the middle of an information frame (point B in Fig. 4), it will stay active for the rest of the frame so that it can detect the beginning of the next signature.

> **Initial probe time** = the partial information frame.
>
> **Access time** = initial probe time + a broadcast cycle.
>
> **Tune-in time** = initial probe time + every signatures in a broadcast cycle + false drop and true drop frames in the cycle.

2.2 Integrated Signature Scheme

A generalization of the simple signature scheme is to generate a signature, called an *integrated signature*, for a group of one or more information frames, called a *frame group*. The integrated signature is broadcast before the frame group. Figure 5 shows the arrangement of the signatures and frame groups. In this scheme, a signature may index any number of information frames. As shown in the figure, the first signature indexes two information frames while the next signature indexes three information frames.

The filtering procedure of the integrated scheme is very similar to the simple signature scheme. When an integrated signature does not match with the query signature, the information frames it indexes can be skipped. Since an integrated signature indexes a large number of information frames, the PCS may stay in doze mode for a long period of time.

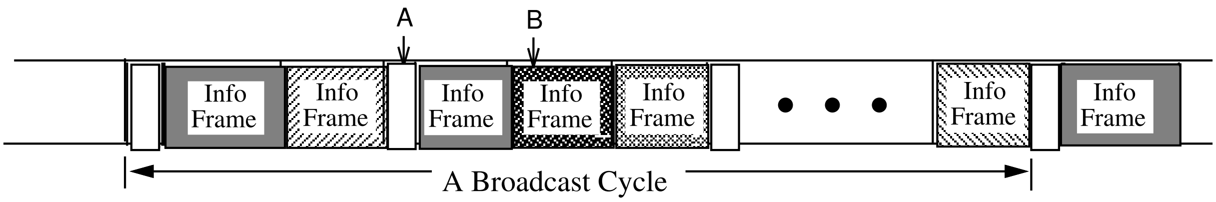

Figure 5: Integrated Signature Scheme.

When a signature match does occur, all of the information frames associated with the signature have to be checked for false drop elimination. Most likely, some of these information frames are not qualified for the query, resulting in unnecessary power consumption. This is a tradeoff for reducing access time.

An unmatched integrated signature allows the PCS to stay in doze mode for a longer period, thus avoiding frequent switching between modes. However, squeezing more information from multiple frames into a signature will increase the probability of false drops. Therefore, we have to properly adjust the size of signatures or reduce the number of bit strings superimposed into the integrated signatures in order to maintain the filtering capability. Grouping similar information frames (those having the same values for most of the indexed attributes) together to generate the integrated signature is virtually the same as reducing the number of bit strings superimposed. Thus, the adjustment will produce more concentrated hits and reduce false drops. This method is good when the order of the information frames is not important.

The average initial probe time for this scheme is half of the average size of the information frames grouped together and their integrated signature. Thus, the average initial probe time is longer than the simple signature scheme. As before, we assume the PCS stays in doze mode during the initial probe time. As in the simple scheme, the initial probe time, access time and tune-in time are dependent on the position of the initial probe.

position A: If the initial probe falls in the middle of a signature frame (point A in Fig. 5), the PCS has to stay active for the rest of the signature frame and then it switches to doze mode.

Initial probe time = length of the partial integrated signature scanned + length of the first group of the information frames.

Access time = initial probe time + a broadcast cycle.

Tune-in time = length of the partial integrated signature scanned + every integrated signature in a cycle + false drop and true drop frame groups.

Note that every information frame in the true match group has to be compared with the query, even though only one qualified frame within the group may exist.

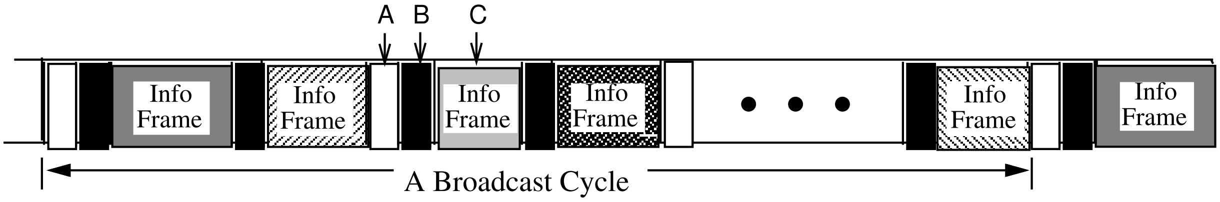

Figure 6: Multi-level Signature Scheme.

position B: If the PCS initially tunes into the middle of an information frame (point B in Fig. 5), it will stay active for the rest of the frame since this is the only way to reach the next frame. If the next frame is an information frame, it will switch to doze mode until the next signature frame arrives.

Initial probe time = the partial information frame scanned and the intervening information frames before the first signature frame arrives

Access time = initial probe time + a broadcast cycle.

Tune-in time = the partial information frame scanned + every integrated signature in a broadcast cycle + false drop and true drop frame groups.

2.3 Multi-level Signature Scheme

The multi-level scheme is a combination of the simple signature and integrated signature schemes [15]. It consists of multiple levels of signatures. Signatures at the upper levels are integrated signatures and those at the lowest level are simple signatures. Figure 6 illustrates a 2-level signature scheme. The white signatures in the figure are integrated signatures. An integrated signature indexes all of the information frames between itself and the next integrated signature of the same or at a higher level. (In the figure, an integrated signature indexes two information frames.) The black signatures are frame signatures for the corresponding information frames. To reduce the false drop probability, the hashing functions used in generating the integrated signatures and simple signatures are different.

To answer a query, a query signature is generated for each level of the signatures. After tuning into the broadcast channel, the corresponding query signatures are used to compare with different levels of signatures. If a signature fails in the comparison, the PCS switches to doze mode until a signature at the same or upper levels arrives. Otherwise, the PCS stays in active mode and continues the filtering process. Take the 2-level signatures in Figure 6 as an example. Query signatures S_Q and $S_Q{}'$ are constructed for the integrated and simple signature levels, respectively. When an integrated signature is received, S_Q is used to match with the signature. If the match fails, the PCS will go into doze mode until the next integrated signature arrives. If the match is successful, $S_Q{}'$ is used to match with the following simple signature. If they match, the corresponding information frame is received; if not, the PCS may go into doze mode until the next signature arrives.

Compared to the other schemes, the multi-level scheme may achieve better tune-in time. The initial probe time, access time and tune-in time are dependent on the position of the initial probe.

position A: If the initial probe falls on an integrated signature frame (point A in Fig. 6), the PCS has to be active for the rest of the integrated signature frame. The filtering process will start afterwards.

Initial probe time = the partial integrated signature scanned.

Access time = initial probe time + a broadcast cycle − the initial integrated signature.

Tune-in time = initial probe time + every simple signature in the first frame group + all but the initial integrated signature in a cycle + simple signatures following the qualified integrated signatures + false drop and true drop frames associated with the qualified simple signatures.

position B: If the initial probe falls on a simple signature frame (point B in Fig. 6), the PCS has to be active for the rest of the signature frame and then it goes into doze mode as assumed before. The filtering process starts after the next signature arrives.

Initial probe time = the partial simple signature frame scanned and its associated information frame.

Access time = initial probe time + a complete broadcast cycle.

Tune-in time = the partial simple signature scanned + every simple signature before the first integrated signature arrives + all of the integrated signature in a cycle + simple signatures associated with the matched integrated signatures + (if the last integrated signature matches) the simple signatures before the initial probe + false drop and true drop information frames associated with the matched simple signatures.

position C: If the PCS initially tunes into the middle of an information frame (point C in Fig. 6), it will stay active for the rest of the frame. Then, filtering starts.

Initial probe time = the partial information frame scanned.

Access time = initial probe time + a broadcast cycle.

Tune-in time = the partial information frame scanned + every simple signature before the first integrated signature + all of the integrated signatures in a cycle + simple signatures associated with the qualified integrated signatures + (if the last integrated signature matches) the simple signatures before the initial probe + false drop and true drop information frames associated with the matched simple signatures.

2.4 Performance Analysis

There are several factors affecting the tune-in time and access time of the signature schemes. For example, we must consider the number and the size of the signatures, the filtering capability of the signatures, the false drop probability of the signatures, and the initial probe time. A performance evaluation has to take these factors into account. The filtering capability and false drop probability may be controlled by the size of the signatures. On the other hand, the initial probe time is related to the number of signatures interleaved with the information frames, and the access time and tune-in time are dependent on the number, size and false drop probability of the signatures.

Frame is the the logical unit of information in the broadcasting, while packet is the physical unit of the broadcasting. Therefore, in our analysis, the performance is estimated in terms of the number of packets.

2.4.1 Symbols and Parameters

- A: number of information frames in a broadcast cycle.
- A_f: number of information frames received due to false drops.
- A_t: number of information frames received due to true drops.
- I: number of integrated signatures in a broadcast cycle.
- I_f: number of integrated signatures matched as false drops.
- I_t: number of integrated signatures matched as true drops.
- P_f: false drop probability.
- P_f^s: false drop probability for simple signatures.
- P_f^i: false drop probability for integrated signatures.
- P_s: selectivity of a query.
- $P(x, y)$: probability that a particular set of x bits are set to 1's in a frame signature which is superimposed from y attribute bit strings.
- k: the number of information frames indexed by an integrated signature.
- l: locality of true drops (average number of true drops in a frame group).
- m: length of a signature in bits.
- n: the average number of packets in an information frame.
- p: the number of bits in a packet.
- r: the number of packets in a signature ($r = \lceil m/p \rceil$).
- s: the number of bit strings which are superimposed into a signature.
- w_b: number of 1's in a bit string generated from hashing.
- w_f: average number of 1's in a frame signature ($\bar{w}_f = m - w_f$).
- $\bar{w}_f$: average number of 0's in a frame signature.

2.4.2 False Drop Probability

The false drop probability is an important factor for the estimation of access time and tune-in time. The false drop probability P_f is defined as:

$$P_f = \frac{A_f}{A - A_t}.$$

In the following, we derive P_f assuming an unsuccessful search of a single value query. Multiple value queries which are Boolean combinations of single value queries may be derived similarly. Based on P_f, the estimation for the access time and tune-in time can be derived.

Assume that a good hash function is used so that each of the potential bit strings has the same probability of being used in generating frame signatures. Given m and w_b, the probability that a particular set of x bits is set to 1's in a frame signature by superimposing y bit strings is:

$$P(x, y) = \sum_{i=0}^{y} (-1)^i \binom{y}{i} \binom{m-i}{w_b}^x \binom{m}{w_b}^{-x}.$$

If x is sufficiently large and $w_b \ll m$,

$$P(x, y) \approx (1 - (1 - w_b/m)^x)^y.$$

A frame signature is generated by superimposing bit strings hashed from the key values in a frame. Therefore, $P(s, 1)$ represents the probability of a bit position β to be set to 1 in the frame signature, where s is the number of distinct key values in an information frame. There are m bits in a signature, so the average number of 1's in a frame signature is:

$$w_f = mP(s, 1) = m(1 - (1 - w_b/m)^s).$$

As a result, the average number of 0's set in a frame signature is:

$$\bar{w}_f = m - w_f = m(1 - w_b/m)^s \approx me^{-w_b s/m}.$$

A false drop occurs when each of the bits in the frame signature corresponding to the 1's in the query signature is set to 1. In other words, a false drop occurs when the following condition holds:

$$\text{if} \quad \alpha_i = 1 \quad \text{then} \quad \beta_i = 1, \quad 1 \leq i \leq m$$

where α_i and β_i are the i-th bit of the query signature and frame signature, respectively. Therefore, a false drop occurs when the following condition holds:

$$\text{if} \quad \beta_i = 0 \quad \text{then} \quad \alpha_i = 0, \quad 1 \leq i \leq m$$

For a query with single key value, the false drops probability is:

$$\begin{aligned}
P_f &= Probability[\alpha_1 = 0 \wedge \alpha_2 = 0 \wedge \cdots \wedge \alpha_{\bar{w}_f} = 0] \\
&= \binom{m - \bar{w}_f}{w_b} / \binom{m}{w_b} \\
&= \frac{(m - w_b)!/(m - \bar{w}_f - w_b)!}{m!/(m - \bar{w}_f)!} \\
&= \frac{m - w_b}{m} \cdot \frac{m - w_b - 1}{m - 1} \cdots \frac{m - w_b - \bar{w}_f + 1}{m - \bar{w}_f + 1} \\
&= (1 - \frac{w_b}{m})(1 - \frac{w_b}{m - 1}) \cdots (1 - \frac{w_b}{m - \bar{w}_f + 1}) \\
&\approx (1 - \frac{1}{m})^{w_b}(1 - \frac{1}{m - 1})^{w_b} \cdots (1 - \frac{1}{m - \bar{w}_f + 1})^{w_b} \\
&= \left(\frac{(m - 1)(m - 2)...(m - \bar{w}_f)}{m(m - 1)...(m - \bar{w}_f + 1)}\right)^{w_b} \\
&= (\frac{m - \bar{w}_f}{m})^{w_b} \\
&= (1 - \frac{\bar{w}_f}{m})^{w_b} \\
&\approx (1 - e^{-w_b s/m})^{w_b}.
\end{aligned}$$

Based on [18], the above formula is optimal when:

$$w_b = w_{opt} = m \cdot ln2/s.$$

Consequently, the optimal false drop probability is:

$$P_f \approx 0.5^{w_{opt}}.$$

2.4.3 Cost Models

In this section, we develop the cost models for the initial probe time, access time and tune-in time for the three signature schemes we described. We use the number of packets as the unit for time estimation. To simplify our discussion, we assume that every information frame has the same number of packets. Therefore, the total number of packets for the data part is:

$$DATA = A \cdot n.$$

Simple Signature Scheme

The total number of packets for the simple signatures in a cycle is:

$$SIG_s = A \cdot \lceil m/p \rceil = A \cdot r.$$

We use $CYCLE_s$ to denote the length of a complete cycle for the simple signature scheme:

$$CYCLE_s = SIG_s + DATA.$$

The initial probe time is the period of time before the next signature arrives. Therefore, the average initial probe time is:

$$PROBE_s = (r + n)/2.$$

After the initial probe period, the filtering process will last for a complete broadcast cycle. Therefore, the average access time is the sum of the initial probe time and the broadcast cycle.

$$\begin{aligned} ACCESS_s &= PROBE_s + CYCLE_s \\ &= (A + 0.5) \cdot (r + n). \end{aligned}$$

Let PT_s denote the period in which the PCS is active during the initial probe time.

$$\begin{aligned} PT_s &= \frac{r \cdot 1/2 \cdot r + n \cdot 1/2 \cdot n}{r + n} \\ &= \frac{r^2 + n^2}{2(r + n)}. \end{aligned}$$

To estimate the tune-in time, we have to first estimate the number of true drops. Let P_s denote the selectivity of a query, the number of true drops is:

$$A_t = A \cdot P_s.$$

In the filtering process, the PCS has to tune in for all of the signature frames. In addition, it has to tune in for the true drop and false drop information frames as well. Therefore, the tune-in time is:

$$\begin{aligned} TUNE_s &= PT_s + SIG_s + A_t \cdot n + A_f \cdot n \\ &= PT_s + SIG_s + A_t \cdot n + P_f^s \cdot A \cdot n - P_f^s \cdot A_t \cdot n \\ &= PT_s + SIG_s + A \cdot n \cdot P_s + A \cdot n \cdot P_f^s - A \cdot n \cdot P_s \cdot P_f^s \\ &= PT_s + SIG_s + DATA \cdot P_s + DATA \cdot P_f^s - DATA \cdot P_s \cdot P_f^s. \end{aligned}$$

Integrated Signature Scheme

Assume that k information frames are grouped together in generating an integrated signature. The total number of integrated signatures is:

$$I = \lceil A/k \rceil.$$

The total number of packets for the integrated signatures in a cycle is:

$$SIG_i = I \cdot r = \lceil A/k \rceil \cdot r.$$

Therefore, the length of a complete broadcast cycle for the integrated signature scheme is:

$$CYCLE_i = SIG_i + DATA.$$

The average initial probe time is:

$$PROBE_i = (r + k \cdot n)/2.$$

Similar to the simple scheme, the access time for the integrated scheme is:

$$ACCESS_i \;=\; PROBE_i + CYCLE_i.$$

Let PT_i denote the duration in which the PCS is kept active during the initial probe time.

$$\begin{aligned} PT_i &= \frac{r \cdot 1/2 \cdot r + n \cdot k \cdot 1/2 \cdot n}{r + n \cdot k} \\ &= \frac{r^2 + n^2 \cdot k}{2(r + n \cdot k)}. \end{aligned}$$

The integrated scheme is good for broadcast with similarity among information frames, because similar information frames can be grouped together in generating the integrated signatures. Consequently, when a true match occurs, the frame group is likely to contain more than one qualified information frame. Let l denote the average number of qualified frames corresponding to a true drop integrated signature. The tune-in time for the integrated signature scheme is:

$$\begin{aligned} TUNE_i &= PT_i + SIG_i + I_t \cdot n \cdot k + I_f \cdot n \cdot k \\ &= PT_i + SIG_i + \lceil \frac{A \cdot P_s}{l} \rceil \cdot n \cdot k + P_f^i \cdot I \cdot n \cdot k \\ &\approx PT_i + SIG_i + \frac{P_s \cdot k}{l} \cdot DATA + P_f^i \cdot DATA. \end{aligned}$$

Note that P_f^i is a function of the number of bit strings superimposed, which we expect to be smaller than $s \cdot k$.

Multi-level Signature Scheme

In this section, we assume a two-level signature scheme, which consists of integrated signatures at the higher level and simple signatures at the lower level. Thus, the total number of packets occupied by the signatures is:

$$SIG_m = SIG_i + SIG_s.$$

The length of a complete cycle is:

$$CYCLE_m = SIG_m + DATA.$$

The average initial probe time is derived based on the probability of the initial probe location and the corresponding probe time:

$$\begin{aligned} PROBE_m &= \frac{k \cdot n}{k \cdot n + k \cdot r + r} \cdot \frac{n}{2} + \frac{k \cdot r}{k \cdot n + k \cdot r + r} \cdot (\frac{r}{2} + n) + \frac{r}{k \cdot n + k \cdot r + r} \cdot \frac{r}{2} \\ &= \frac{k \cdot n^2 + (k+1)r^2 + 2 \cdot k \cdot nr}{2(k \cdot n + (k+1)r)}. \end{aligned}$$

The access time can be approximated as follows.

$$ACCESS_m = PROBE_m + CYCLE_m.$$

The tune-in time in the initial probe period is:

$$\begin{aligned} PT_m &= \frac{k \cdot n}{k \cdot n + k \cdot r + r} \cdot \frac{n}{2} + \frac{k \cdot r}{k \cdot n + k \cdot r + r} \cdot \frac{r}{2} + \frac{r}{k \cdot n + k \cdot r + r} \cdot \frac{r}{2} \\ &= \frac{k \cdot n^2 + (k+1)r^2}{2(k \cdot n + (k+1)r)}. \end{aligned}$$

To simplify the formula for the estimate of the tune-in time, we assume that the integrated signature for the group of the initial probe is a true drop. Therefore, the tune-in time can be approximated as follows.

$$\begin{aligned} TUNE_m = {} & PT_m + SIG_i + k \cdot r + k \cdot P_s \cdot n \\ & + k \cdot P_f^s \cdot n + (I - \lceil P_s \cdot A/l \rceil - 1) \cdot P_f^i \cdot (k \cdot r + k \cdot P_f^s \cdot n) \\ & + \lceil P_s \cdot A/l \rceil (k \cdot r + l \cdot n + (k-l) \cdot P_f^s \cdot n). \end{aligned}$$

2.4.4 Comparisons

Using the formulae developed above, we compare the access time and tune-in time of the three schemes. Table 1 lists the parameter values used in the comparisons. In the comparisons, we assume that the selectivity of a query is 1%. For the integrated and multi-level

Table 1: Parameters of the cost models.

p	=	128	k	=	4
n	=	1000	l	=	3
s_s	=	100	A	=	10000
s_i	=	280	P_s	=	0.01

schemes, four information frames are grouped together to generate an integrated signature. The false drop probability for the simple and integrated signatures can be calculated based on m and the number of bit strings superimposed, s_s and s_i. Since the integrated scheme is good for broadcast with similar information frames, we assume that 30% of the superimposed bit strings are overlapped. Therefore, $s_i = 70\% \cdot s_s \cdot 4$. Also, we assume that the locality of frames is three. In other words, for the truely qualified integrated signatures, three out of its associated information frames are truely qualified. The same assumption is also applied for the multi-level scheme. We vary the signatures length, m, from 1 to 35 physical frames to observe the changes on access time and tune-in time.

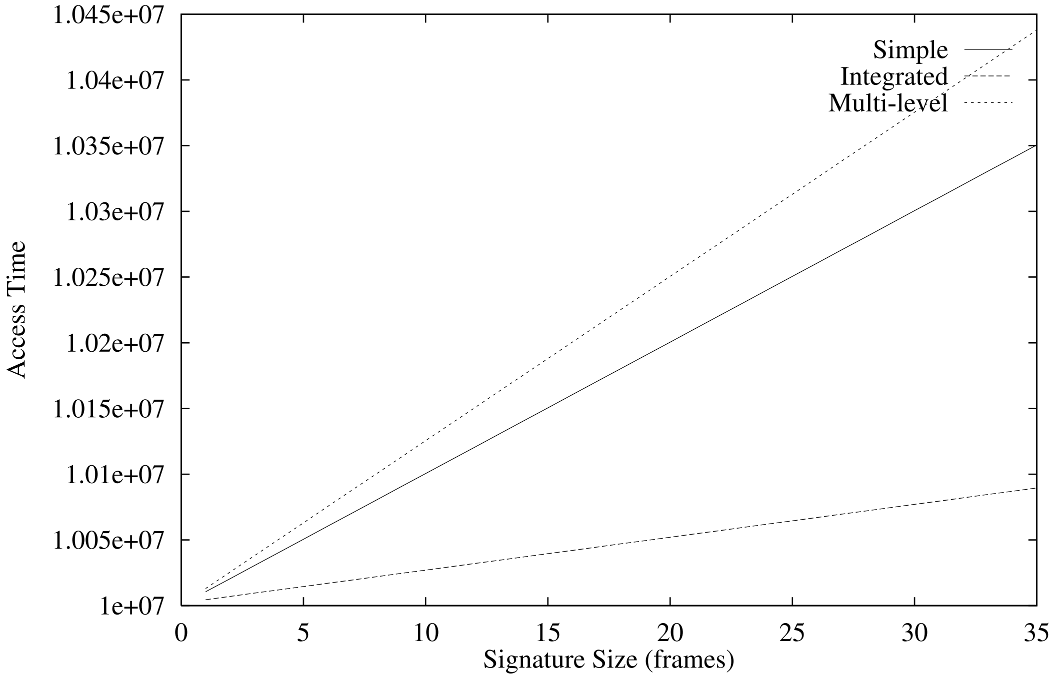

Figure 7: Access time vs Signature size.

Figure 7 shows that the access times of the three schemes are linearly proportional to the size of the signatures. Since the overall size of the information frames is fixed at 10^7 physical frames, the increase in access time over the signature size represents the overheads of the signature schemes. From the figure, we also find that the overhead of the multi-level scheme is close to the sum of the overheads for the other two schemes. This is attributed to the fact that the multi-level scheme is a combination of the other two schemes and that the overall size of signatures plays an important role to the increased access time.

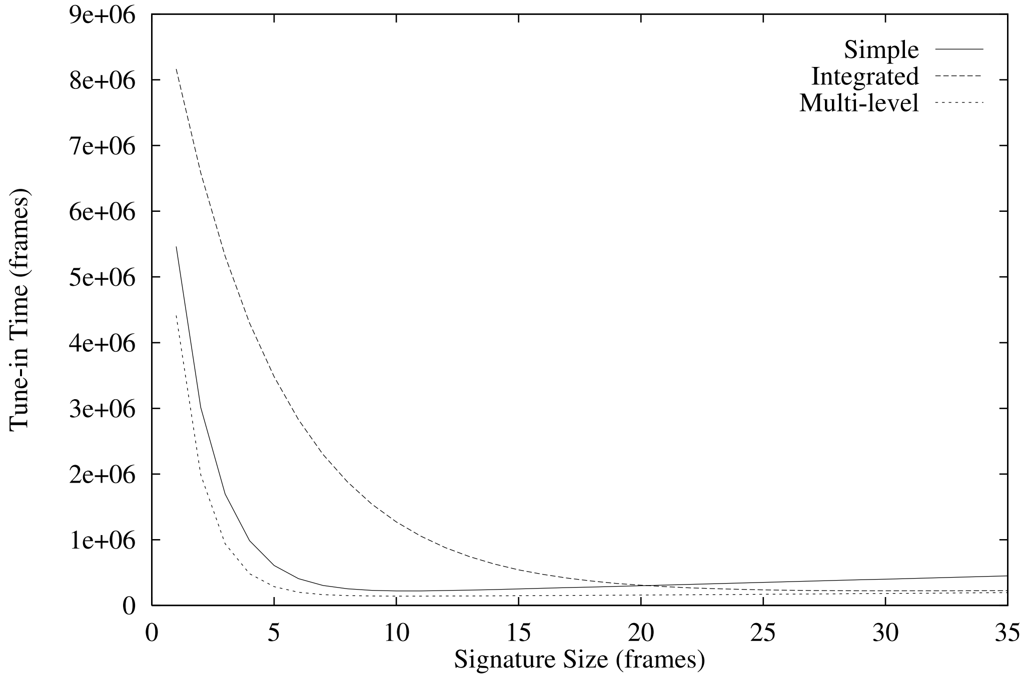

Figure 8: Tune-in time vs Signature size.

Figure 8 shows the tune-in time for the three schemes. From the figure, we may observe that the tune-in time decreases to the minimal and then increases again as the size of the signatures increase. From the data collected, we observe that at those minimal points the false drop probability is so low that false drops become insignificant to the tune-in time. Increasing the signature length further would only increase access time and tune-in time. The signature sizes and access time corresponding to the minimal tune-in time of the three schemes are listed in Table 2. The table shows the best signature sizes for the parameters used. From the table, we may estimate that the access delays are less than 1.4% for all three schemes, while the tune-in time save is more than 97.7%.

Although Table 2 gives us the best choice of the signature sizes for the three schemes, it doesn't give us a fair comparison for the performance among the three schemes. Figure 9 compares the tune-in times in terms of access time. With a given access time, the figure tells the best choice among the three schemes. From the figure, we may come to the conclusion that the integrated scheme and multi-level scheme are better than the simple scheme, while the integrated scheme is the best when access time is small and the multi-level scheme prevails otherwise. However, the performance of the integrated scheme is dependent on the locality in the frame groups and the overlaps among the superimposed bit strings. If the

Table 2: Signature Size, Access Time w.r.t. Minimal Tune time

	Minimal Tune Time	Access Time	Sig. Size
Simple	221620.82	10100505.00	10
Integrated	222370.82	10077015.00	31
Multi-level	142319.26	10138007.58	11

locality is 1 and there is no overlap, the integrated scheme will have a worse tune-in time than the simple scheme due to longer initial probe time and heavy overhead for the true drop frames (there is no overhead for true drops in simple scheme). In other words, the performance of the integrated scheme is highly dependent on the contents of the information frames and their clustering.

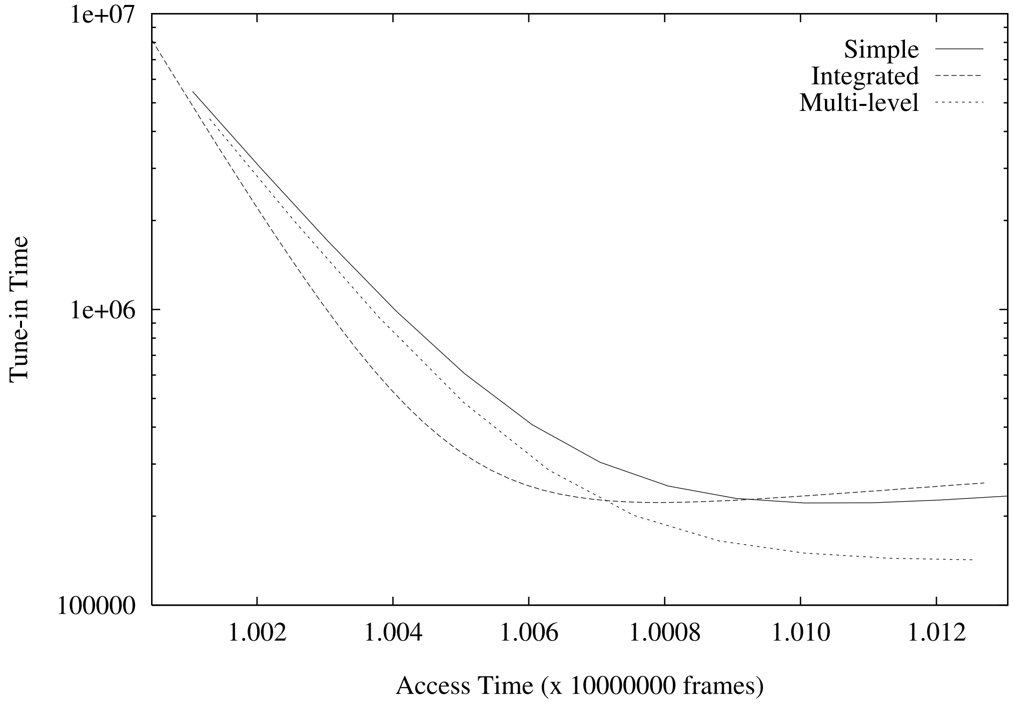

Figure 9: Tune-in time vs Access Time ($l = 3; s_i = 270$).

For the multi-level scheme, however, the signatures at the lowest level are indeed simple signatures. We expect the tune-in time of this scheme to be at least as good as the simple scheme with some access time delay for upper level signatures. Therefore, we compare the worst case of the integrated and multi-level schemes (i.e., the locality is 1 and there is no overlap bit strings in a frame group) to the simple scheme. Figure 10 shows that with small access delay (i.e., less than 0.6% of the optimal access time) the tune-in time of the simple scheme and the multi-level scheme is roughly the same. For access delays of more than 0.6% of the optimal access time, the multi-level scheme has a much better tune-in time than the simple scheme.

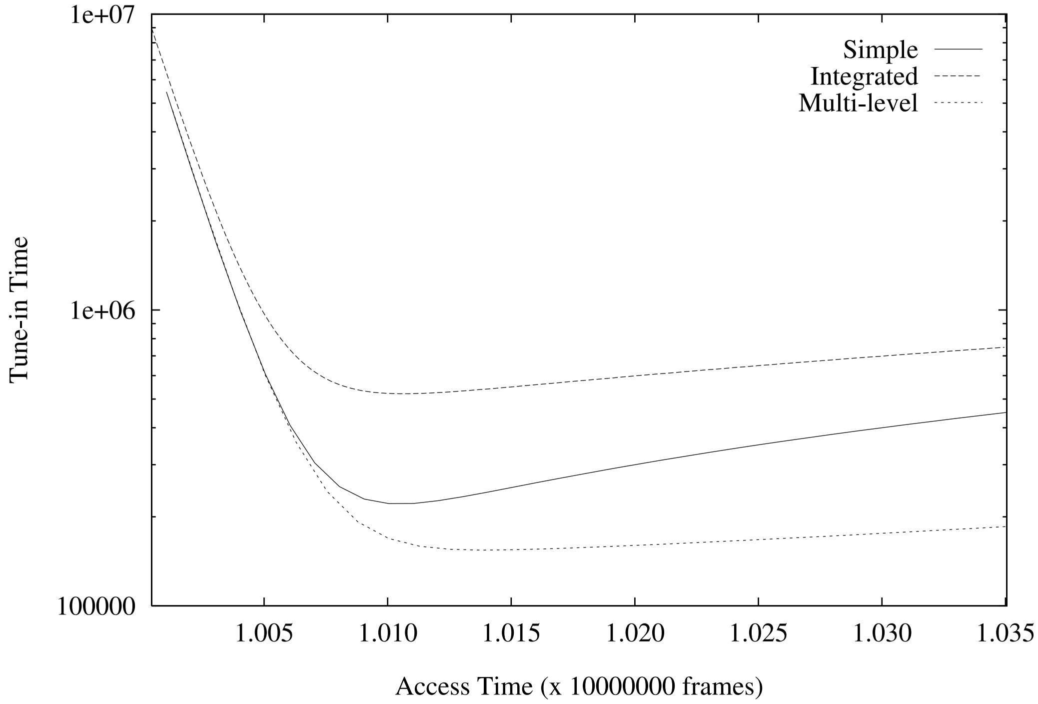

Figure 10: Tune-in time vs Access Time ($l = 1; s_i = 400$).

2.5 Optimization of Signatures

In the previous section, we fixed the size of the signatures used in the multi-level signature scheme. Since the integrated signatures index more information frames than the simple signatures do, the false drop probability of the integrated signatures is higher than that of the simple signatures. On one hand, since the integrated signatures are more frequently used in filtering than the simple signatures, it is desirable to lower their false drop probability by increasing the signature length in order to reduce further matching of the associated simple signatures. On the other hand, lengthening the integrated signatures will increase the tune-in time.

In this section, we fix the total overhead of the signatures and adjust the ratio of the storage occupied by the simple and integrated signatures in order to observe the optimal tune-in time for the multi-level signature scheme. To be consistent with our discussion in previous session, we assume a two-level signature scheme. In the following, we use subscripts i and s to distinguish the symbols used for integrated signatures and simple signatures, respectively.

Assume that the size of the overall signature overhead is SIG_m. The portion of the overhead used for integrated signatures is q. Therefore,

$$SIG_i = q \cdot SIG_m$$

and

$$SIG_s = (1 - q) \cdot SIG_m.$$

As a result, the numbers of packets allocated for each integrated signature and simple signature are: $r_i = \lfloor SIG_i/I \rfloor$ and $r_s = \lfloor SIG_s/A \rfloor$.

The cost formulae for the multi-level signature scheme are reformulated as follows. We only list the formulae which are different from that in Section 2.4.3.

$$\begin{aligned} PROBE_m &= \frac{k \cdot n}{k \cdot n + k \cdot r_s + r_i} \cdot \frac{n}{2} + \frac{k \cdot r_s}{k \cdot n + k \cdot r_s + r_i} \cdot (\frac{r_s}{2} + n) + \frac{r_i}{k \cdot n + k \cdot r_s + r_i} \cdot \frac{r_i}{2} \\ &= \frac{k \cdot (n + r_s)^2 + {r_i}^2}{2(k \cdot n + k \cdot r_s + r_i)}. \end{aligned}$$

The tune-in time in the initial probe period is:

$$\begin{aligned} PT_m &= \frac{k \cdot n}{k \cdot n + k \cdot r_s + r_i} \cdot \frac{n}{2} + \frac{k \cdot r_s}{k \cdot n + k \cdot r_s + r_i} \cdot \frac{r_s}{2} + \frac{r_i}{k \cdot n + k \cdot r_s + r_i} \cdot \frac{r_i}{2} \\ &= \frac{k \cdot n^2 + k \cdot {r_s}^2 + {r_i}^2}{2(k \cdot n + k \cdot r_s + r_i)}. \end{aligned}$$

The tune-in time can be approximated as follows.

$$\begin{aligned} TUNE_m &= PT_m + SIG_i + k \cdot r_s + k \cdot P_s \cdot n + k \cdot P_f^s \cdot n \\ &\quad +(I - \lceil P_s \cdot A/l \rceil - 1) \cdot P_f^i \cdot (k \cdot r_s + k \cdot P_f^s \cdot n) \\ &\quad +\lceil P_s \cdot A/l \rceil (k \cdot r_s + l \cdot n + (k - l) \cdot P_f^s \cdot n). \end{aligned}$$

2.5.1 Comparison

Based on the cost models we developed above, we like to see the change in tune-in time based on different ratios between the integrated and simple signatures. Figure 11 shows the tune-in time with respect to the share of the total signature overhead that the integrated signatures occupy, q, given that the total signature overhead is $100,000$ physical frames. We increase q from 0 to 1 by 1% each time. From the figure, optimal tune-in time is at $q = 0.3$. In other words, when 30% of the total signature overhead are used for the integrated signatures, the tune-in time is the lowest. In this case, the sizes of the integrated and simple signatures are 12 and 7 packets per signature, respectively. However, the optimal point may

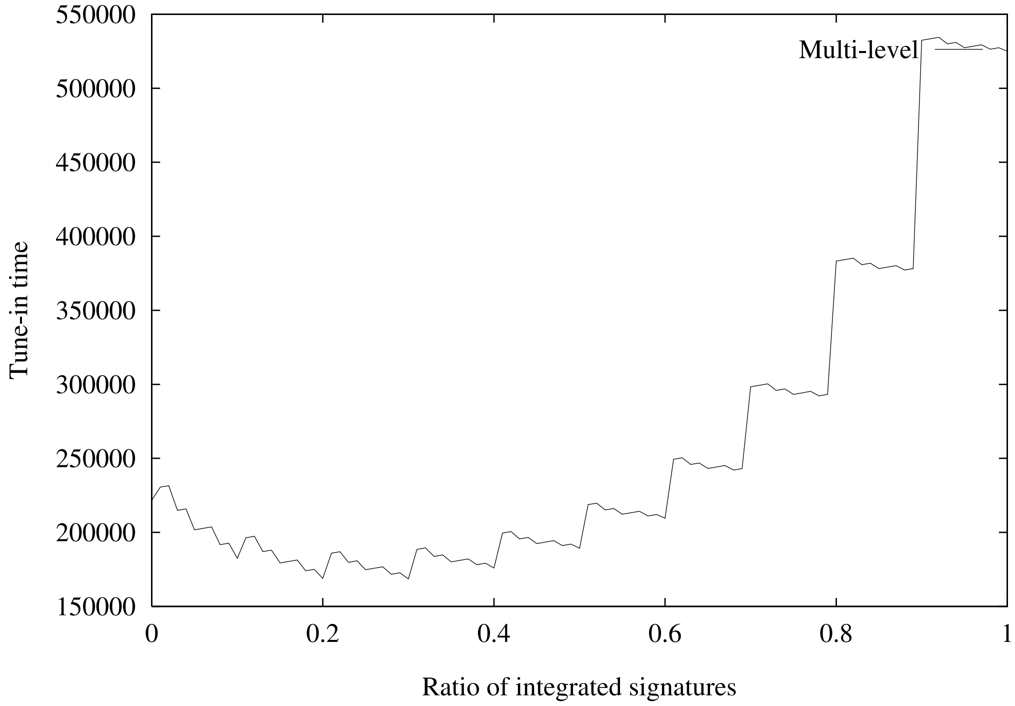

Figure 11: Tune-in time vs integrated signature share.

change when a different total signature overhead is used. Since packets have fixed size, the allocated signature space must be truncated to fit into an integral number of packets. This situation can also be observed from the zig-zag behavior in Figure 11.

Figure 12 compares, based on the same access time, the tune-in time of the "optimal" multi-level signature scheme and the original multi-level signature scheme (which has the same signature length for the integrated and simple signatures). In this experiment, we set the size of frame group to 15. Other experiments with different frame group sizes have similar results. We observe that although the performance of the suboptimal configuration is not as good as the optimal one, the gap is quite insignificant. Therefore, one may choose to adopt the suboptimal configuration for the simplicity of signature generation and comparison.

3 Caching of Signatures

Caching of frequently accessed information in the PCSs can reduce tune-in time, and thus power consumption, because information can be fetched from the cache without tuning into the communication channel. Previous studies [2,6] have discussed cache management policies and invalidation strategies for information dissemination based on broadcast. They have considered caching the information frames in the PCSs. However, due to physical

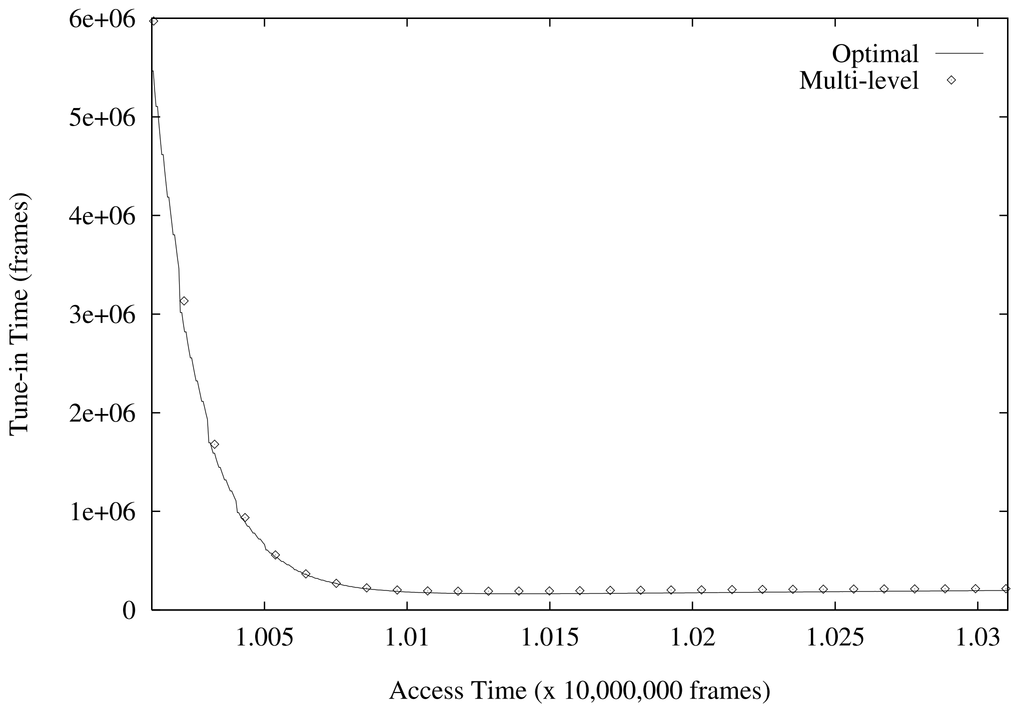

Figure 12: Optimal tune-in time vs access time.

constraints, PCSs usually have relatively small memory. Caching large chunks of multi-media information frames is infeasible. Comparing to the information frames, signatures need a rather small space overhead and they contain critical information to support various kinds of queries. Thus, they are very suitable as the caching entity. In the following, we consider the signature caching schemes for information filtering for mobile users.

To simplify our discussion, we consider the two-level signature scheme in our discussion. There are two factors affecting signature maintenance in cache:

- Invalidation notice: An invalidation notice indicates which cached signatures are stale.

  In order to support information filtering, the signatures in the cache have to be accurate. Therefore, the information server has to provide invalidation information to the PCS. If some information frames and heir corresponding signatures are changed, the information server will indicate the changes in the invalidation notice embedded in the broadcast. There are two kinds of information notices, aiming at different situations.

  - Bit tags: For each information frame, a 1-bit tag is allocated to indicate whether the corresponding information frame has been changed since the last broadcast cycle. The overhead of the bit tags is very low. However, the invalidation information only indicates changes with respect to the immediate preceding cycle. If a PCS has stopped tracking the invalidation information, it cannot tell if the signatures in the cache is valid or not, even though the invalidation notice indicates no changes. Therefore, when a mobile client tunes into a channel, it has to reload the signatures into its cache memory in accordance with the caching policies used.

  - Version numbers: To reduce the cost of reloading the signatures into the cache every time when a PCS tunes into a channel, multiple-bit version numbers may be used to serve as the invalidation notice. In the broadcast, a version number is assigned to each information frame. If the information frame is modified, its version number is incremented by 1 (and reset to 0 when it reaches the maximum number representable).

    The size of the version number is dependent on the frequency of updates on the information frames. However, it shouldn't exceed that of the simple signatures, because, if it does, the mobile clients may simply listen to the signatures. Due to the limited length allowed for the version numbers, the information server has to decide a period when the same version number won't appear twice. The mobile clients will assume that the signatures they cached are expired after they lost track of the channel for the specific period of time and reload the signatures in accordance with the caching policy used. If the mobile users listen to the broadcast channels before the version numbers expired, they only need to reload the signatures which are changed during the off period.

- Refresh strategy: The refresh strategy determines which signatures to maintain in the cache.

  - Active refresh: Active refresh maintains all of the signatures in the cache. In order to maintain the accuracy of the signatures, the PCS has to load the updated signatures into the cache based on the invalidation notices. As a result, the performance of the active refresh strategy is influenced by the number of updates on the information frames.
  - Passive refresh: Passive refresh only keeps the previously accessed signatures in the cache. Instead of maintaining all of the signatures in the cache, only the signatures received in the previous filtering process are kept in the cache. According to the invalidation notices, invalid signatures are cleared from the cache without refresh. Therefore, the performance of information filtering using passive refresh caching policies is not affected by the number of updates on the information frames.

3.1 Analysis of the Caching Policies

In the following, we propose four signature caching policies based on the invalidation notice and cache refreshing strategies. The cost models for tune-in time and access time are derived and their performance is compared. We assume that a two-level signature scheme is used in information broadcasting and filtering. The caching policies may be easily applied to other signature schemes. In the analysis, we assume that the average percentage of information frames modified is P_u. Similar to the locality of qualified information frames, we define locality, u, as the number of modified information frames in a frame group.

3.1.1 Bit Tags and Active Refresh (BA)

The BA policy maintains all of the integrated signatures and simple signatures in the cache memory. For each frame group, we use bit tags in front of the integrated signature to indicate the update status of the information frames in the group. When a bit tag signals that the corresponding information frame is changed, the integrated and simple signatures for the frame have to be loaded into the cache memory. The query signatures are then matched with the signatures maintained in the cache to decide which information frames may be skipped and which frames have to be brought into the PCSs for false drop elimination. k bits are necessary for a group of k frames. The total broadcast overhead for invalidation information is:

$$II_{BA} = \lceil k/p \rceil \cdot \lceil A/k \rceil.$$

Note that the invalidation information may be combined with the integrated signatures to make more compact use of the packets. Here we estimate its upper bound overhead.

For new users or those who lost track of the invalidation information, the tune-in time for loading signatures into the cache in the initial cycle is the number of frames for bit tags plus the number of signatures in a cycle:

$$Init_{BA} = II_{BA} + SIG_m.$$

Since the cache loading process may be combined with the filtering process, a query may be answered while the signatures are being loaded into the cache. Therefore, the actual tune-in time is the sum of the initial probe time, the signature loading time, and the time to load the qualified information frame for false drop verification. Thus, the initial tune-in time is:

$$\begin{aligned} Tune_{BA}^{init} &= PT_m + Init_{BA} + (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k \cdot P_f^s \cdot n \\ &\quad + \lceil P_s \cdot A/l \rceil (l \cdot n + (k-l) \cdot P_f^s \cdot n). \end{aligned}$$

For the subsequent queries, the tune-in time is consumed for loading the modified signatures and listening to information frames for false drop elimination. The average numbers of the modified integrated and simple signatures are $P_u \cdot A$ and $\lceil P_u \cdot A/u \rceil$ respectively. The tune-in time for the subsequent queries is:

$$\begin{aligned} Tune_{BA}^{next} &= PT_m + II_{BA} + P_u \cdot A \cdot r + \lceil P_u \cdot A/u \rceil \cdot r + (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k \cdot P_f^s \cdot n \\ &\quad + \lceil P_s \cdot A/l \rceil (l \cdot n + (k-l) \cdot P_f^s \cdot n). \end{aligned}$$

3.1.2 Version Numbers and Active Refresh (VA)

The VA scheme uses the version numbers of the information frames to inform mobile users of changes on the frames. Similar to bit tags, the version numbers for the information frames in a frame group are broadcasted before the integrated signatures. If the version number of a frame in the current broadcast is different from that of the frame in cache memory, we know that the signatures in the cache is not valid. Therefore, the updated signatures have to be brought into the cache. Due to the constraint on the size of the version numbers, the version numbers are valid only for a certain period of time. If a PCS loses track of the broadcast channel over that period of time, we will consider it as a new client for the channel. A new client has no prior knowledge of the channel and thus has to load the signatures into its cache in accordance with the caching policy. For an old client, however, only out-dated signatures will be deleted and reloaded.

Assume that the size of a version number is v bits. The period of time in which the version number is guaranteed valid is $2^v - 1$ broadcast cycles.

The total broadcasting overhead for the invalidation information is:

$$II_{VA} = \lceil k \cdot v/p \rceil \cdot \lceil A/k \rceil.$$

For a new user to load signatures into the cache, the tune-in time is:

$$Init_{VA}^{new} = II_{VA} + SIG_m.$$

For an old user who lost track of the broadcast channel for t cycles, where $t < 2^v - 1$, the average number of information frames which have not been changed during this period of time is $A \cdot (1 - P_u)^t$. Therefore, the number of simple signatures which have to be reloaded is $A - A \cdot (1 - P_u)^t$. Similarly, the number of integrated signatures which have to be reloaded is $I - I \cdot (1 - \frac{\lceil A \cdot P_u/u \rceil}{I})^t$. Therefore, the signature loading time for an old user is:

$$\begin{aligned} Init_{VA}^{old} &= II_{VA} + (A - A \cdot (1 - P_u)^t) \cdot r + (I - I \cdot (1 - \frac{\lceil A \cdot P_u/u \rceil}{I})^t) \cdot r \\ &= II_{VA} + SIG_m - SIG_s \cdot (1 - P_u)^t - SIG_i \cdot (1 - \frac{\lceil A \cdot P_u/u \rceil}{I})^t. \end{aligned}$$

If the initial cache loading process of a client is combined with its first query evaluation, the total tune-in time is:

$$\begin{aligned} Tune_{VA}^{init} = \; & PT_m + Init_{VA}^{*} + (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k \cdot P_f^s \cdot n \\ & + \lceil P_s \cdot A/l \rceil (l \cdot n + (k - l) \cdot P_f^s \cdot n), \end{aligned}$$

where $Init_{VA}^{*}$ is either $Init_{VA}^{new}$ or $Init_{VA}^{old}$ depending on whether the user is new to the channel or not.

For the subsequent queries, the tune-in time is consumed for loading the modified signatures and listening to information frames for false drop elimination. The tune-in time for the subsequent queries is:

$$\begin{aligned} Tune^{next}_{VA} = \quad & PT_m + II_{VA} + P_u \cdot A \cdot r + \lceil P_u \cdot A/u \rceil \cdot r + (I - \lceil P_s \cdot A/l \rceil) \cdot P^i_f \cdot k \cdot P^s_f \cdot n \\ & + \lceil P_s \cdot A/l \rceil (l \cdot n + (k-l) \cdot P^s_f \cdot n). \end{aligned}$$

3.1.3 Bit Tag and Passive Refresh (BP)

In the BP policy, instead of caching all of the signatures in the PCS, only the signatures received in previous queries are kept in the memory. Instead of actively updating all of the signatures in the cache memory to maintain their accuracy, the outdated signatures are flushed out of the cache in accordance with the bit tags. New signatures are brought into the cache only when they are needed in the current filtering process. Like the BA policy, the bit tags corresponding to a frame group are broadcasted before the integrated signature. If the bit tags show no change in the frame group, it will use the integrated signature in cache for processing the query. If the bit tags indicate changes in some information frames, the corresponding integrated and simple signatures in cache are deleted and the updated integrated signature is loaded into the cache for comparison with the integrated query signature. If the comparison between the query signature and the integrated signature is a match, the simple signatures not residing in the cache will be loaded into the cache and compared to the simple query signature. The subsequent steps for checking information frames are based on the result of the comparison. If the comparison between the integrated signatures failed, we simply tune off without refilling the cache for the deleted signatures.

The total number of frames needed for the BP scheme is:

$$II_{BP} = \lceil k/p \rceil \cdot \lceil A/k \rceil.$$

Because of the passive manner on signature caching adopted by the BP policy, we don't consider the tune-in time needed for only loading signatures. Since we don't bring all of the signatures into the cache, the tune-in time for the initial query is the same as answering a query without caching plus the overhead for the bit tags.

$$\begin{aligned} Tune^{init}_{BP} \quad = \quad & PT_m + SIG_i + (I - \lceil P_s \cdot A/l \rceil) \cdot (P^i_f \cdot k \cdot r + P^i_f \cdot k \cdot P^s_f \cdot n) \\ & + \lceil P_s \cdot A/l \rceil (k \cdot r + l \cdot n + (k-l) \cdot P^s_f \cdot n). \end{aligned}$$

In the initial cache loading cycle, the average number of signatures received and maintained by the PCS is $I + (I - \lceil P_s \cdot A/l \rceil) \cdot P^i_f \cdot k + \lceil P_s \cdot A/l \rceil \cdot k$.

Let I_{cache} and A_{cache} be the number of integrated and simple signatures in cache respectively. Then,

$$A_{cache} = (I - \lceil P_s \cdot A/l \rceil) \cdot P^i_f \cdot k + \lceil P_s \cdot A/l \rceil \cdot k.$$

In this scheme, every integrated signatures has be maintained in cache because every integrated signature has to be checked in order to decide whether the corresponding simple signatures and frame groups may be skipped. Thus,

$$I_{cache} = I.$$

The number of integrated signatures deleted from and brought into the cache are the same as the number of integrated signatures modified: $\lceil P_u \cdot A/u \rceil$. The average number of simple signatures purged from the cache is the number of simple signatures in cache times the percentage of updates: $P_u \cdot A_{cache}$. The simple signatures brought into the cache are those corresponding to successful matches between the integrated signatures and integrated query signature but not residing in the cache: $((I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k + \lceil P_s \cdot A/l \rceil \cdot k) \cdot (1 - \frac{A_{cache}}{A} \cdot (1 - P_u))$.

For the subsequent queries, the average tune-in time is as follows:

$$\begin{aligned} Tune_{BP}^{next} &= PT_m + II_{BP} + \lceil P_u \cdot A/u \rceil \cdot r + ((I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k + \lceil P_s \cdot A/l \rceil \cdot k) \\ &\quad \cdot (1 - \frac{A_{cache}}{A} \cdot (1 - P_u)) \cdot r + (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k \cdot P_f^s \cdot n \\ &\quad + \lceil P_s \cdot A/l \rceil (l \cdot n + (k - l) \cdot P_f^s \cdot n). \end{aligned}$$

3.1.4 Version Numbers and Passive Refresh (VP)

The VP scheme uses the version numbers of the information frames to inform PCSs of changes on the frames and to refresh the cache in a passive manner. Assume that the size of a version number is v bits. The total broadcast overhead for the invalidation information is:

$$II_{VP} = \lceil k \cdot v/p \rceil \cdot \lceil A/k \rceil.$$

Similar to the BP policy, new signatures are brought into the cache only when they are needed in the current filtering process. The tune-in time for the initial query is the same as answering a query without caching plus the overhead for version numbers. However, depending on whether the PCS is new to the channel, the tune-in time for the initial query will be different. For a new client, the initial tune-in time is:

$$\begin{aligned} Tune_{VP}^{init,new} &= PT_m + II_{VP} + SIG_i + (I - \lceil P_s \cdot A/l \rceil) \cdot (P_f^i \cdot k \cdot r + P_f^i \cdot k \cdot P_f^s \cdot n) \\ &\quad + \lceil P_s \cdot A/l \rceil (k \cdot r + l \cdot n + (k - l) \cdot P_f^s \cdot n). \end{aligned}$$

Assume that the number of integrated and simple signatures cached in a mobile client before it tunes off is I_{cache} and A_{cache}, respectively. In the two-level signature scheme, since every integrated signature will be examined, the PCS will keep a complete set of the integrated signatures in the cache. Therefore, $I_{cache} = I$. For an old client which tuned off for t cycles, the number of valid simple signatures in cache is $A_{cache}(1 - P_u)^t$ and the number of valid integrated signatures in cache is $I(1 - \frac{\lceil A \cdot P_u/u \rceil}{I})^t$. As a result, the tune-in time for an old user's initial query is:

$$
\begin{aligned}
Tune_{VP}^{init,old} &= PT_m + II_{VP} + (I - I(1 - \frac{\lceil A \cdot P_u/u \rceil}{I})^t) \cdot r \\
&+ (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k \cdot r(1 - \frac{A_{cache}}{A}(1 - P_u)^t) \\
&+ (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k \cdot P_f^s \cdot n + \lceil P_s \cdot A/l \rceil \cdot k \cdot r(1 - \frac{A_{cache}}{A}(1 - P_u)^t) \\
&+ \lceil P_s \cdot A/l \rceil (l \cdot n + (k - l) \cdot P_f^s \cdot n).
\end{aligned}
$$

After the initial cycle, there is no difference between new and old clients. Therefore, the average tune-in time for the subsequent queries is:

$$
\begin{aligned}
Tune_{VP}^{next} &= PT_m + II_{VP} + \lceil P_u \cdot A/u \rceil \cdot r \\
&+ (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i (1 - \frac{A_{cache}}{A} \cdot (1 - P_u)) \cdot k \cdot r \\
&+ \lceil P_s \cdot A/l \rceil (1 - \frac{A_{cache}}{A} \cdot (1 - P_u)) \cdot k \cdot r + (I - \lceil P_s \cdot A/l \rceil) \cdot P_f^i \cdot k \cdot P_f^s \cdot n \\
&+ \lceil P_s \cdot A/l \rceil (l \cdot n + (k - l) \cdot P_f^s \cdot n).
\end{aligned}
$$

3.2 Comparison of the Caching Policies

There are several factors, such as the percentage of updates, size of signatures, and number of information frames in a group, affecting the performance of the tune-in time for the caching policies. In the following, we vary these factors to observe their influence on the caching policies.

First, we compare the tune-in time of the caching policies for the first ten queries after a mobile user connects to a broadcast channel.The parameter values used in the comparison are the same as that in our experiments in Section 2.4.4. We fix the signature size to 10 packets and the average percentage of frames updated in each cycle to 1%. The locality of the modified frames in a group is 1. We use the tune-in time of the multi-level scheme without caching as a reference. In Figure 13, the symbols for different policies are self-explanatory except that we add N and O in front of VA and VP to distinguish the cases for new and old users. New users have no signatures in cache while old users have some signatures obtained before they were switched off. For the old users, we assume that four broadcast cycles have passed since the old user disconnected. Since the bit tag methods use only one bit per frame, it would be unfair to assume that the bit tags occupy an entire packet. Therefore, we combine the bit tags with the integrated signatures in the experiment. On the other hand, the length of a version number is set to 10 bits.

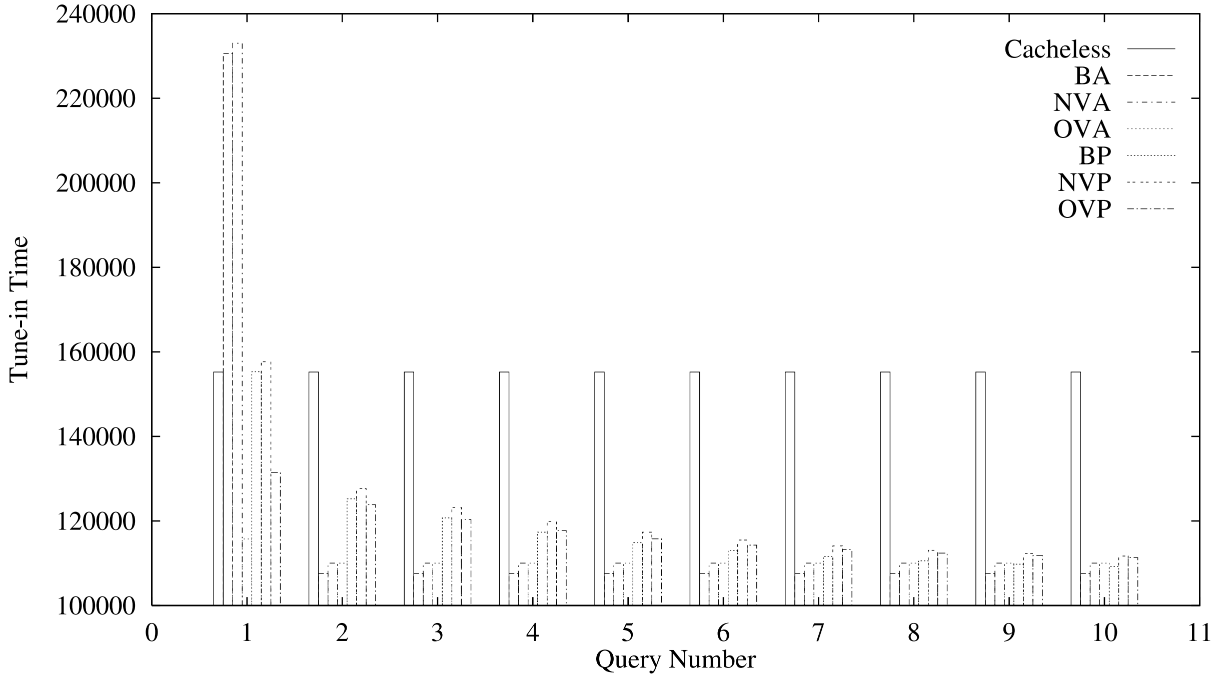

Figure 13: Tune-in time vs query number.

Figure 13 shows the tune-in time of the policies for the initial and subsequent queries. In general, the tune-in time of the initial query is much higher than that of the subsequent queries, because signatures have to be loaded into the caches in the initial query. Furthermore, only the tune-in time of the BA and NVA policies is dramatically higher than that of the cacheless multi-level scheme, because they have to load all of the integrated and simple signatures into the cache. OVA and OVP have the best tune-in time, because they have signatures left in the cache from the previous connection. OVA is better than OVP, because OVA has more signatures left in the cache. On the other hand, the tune-in time of the passive policies for new users is close to the cacheless scheme, because the passive policies are dependent on the signature filtering process.

For the subsequent queries, the BA policy has the best performance, while NVA and OVA, which have identical tune-in time, are closely behind it. The difference between the BA and VA policies is due to the overhead of the invalidation information. In this experiment, the active policies outperform the passive policies. However, as time goes by, the passive policies gradually catch up. Also, note that, the update rate in this experiment is set to 1%, which is rather low. As we will show in the following experiments, the update rate plays an important role in the difference between the active and passive policies.

From the data we collected in this experiment, all of the caching policies have better average tune-in time than the cacheless multi-level method after only listening to the channel for three broadcast cycles. Thus, we conclude that the signature schemes in general will benefit from the caching techniques.

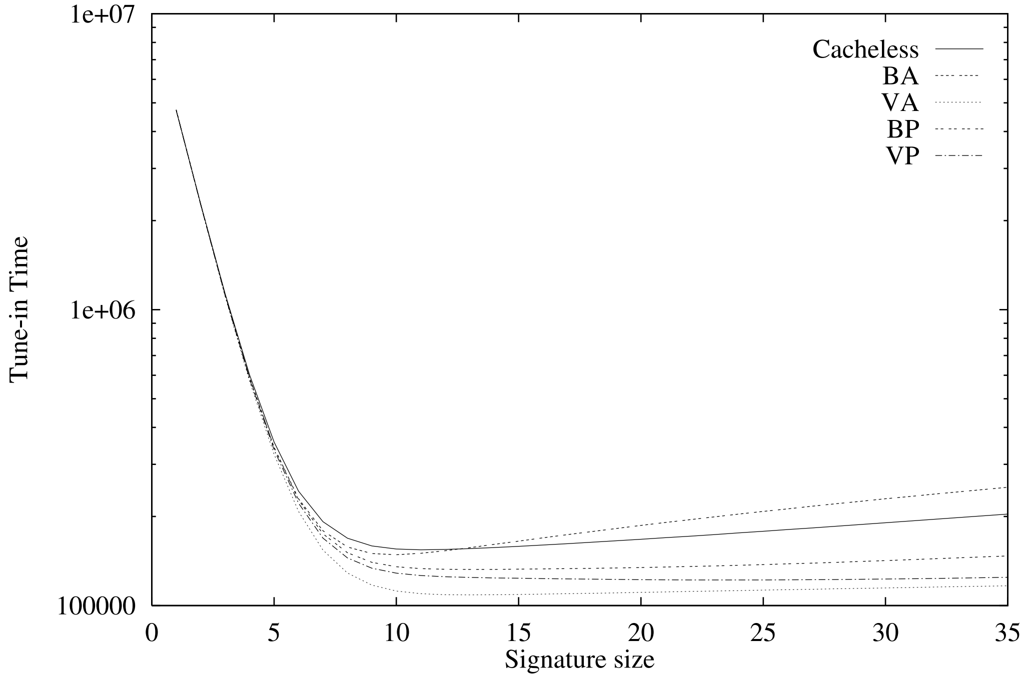

Figure 14: Tune-in time vs signature size.

Next, we compare the tune-in time of the caching policies by changing the size of the signatures. For this experiment, we calculate the average tune-in time for the initial query and two subsequent queries. To simplify our comparison, we use the average tune-in time for old users in the version number policies.

Figure 14 shows that VA $<$ VP $<$ BP $<$ BA. In the figure, the tune-in time of BA is higher than that of the cacheless approach as the frame size increases. In practice, however, after the best performance of a signature scheme is reached, we won't increase the size of the signatures. In other words, the comparison should emphasize on the minimal points of the policies.

The above comparison shows the best performance of the policies. However, it didn't show their overhead on access time. In Figure 15, we show the tune-in time against the access time for the proposed caching policies. As shown in the figure, the tune-in time will benefit from the caching policies with about 75,000 packets of access time delay. Also, as the number of subsequent queries increase, the tune-in time for the caching policies will decrease. Therefore, the required access delay for the caching policies to be effective will be lower. However, due to the broadcast overhead on the invalidation information of the caching policies, signature filtering without caching will still be better when access time

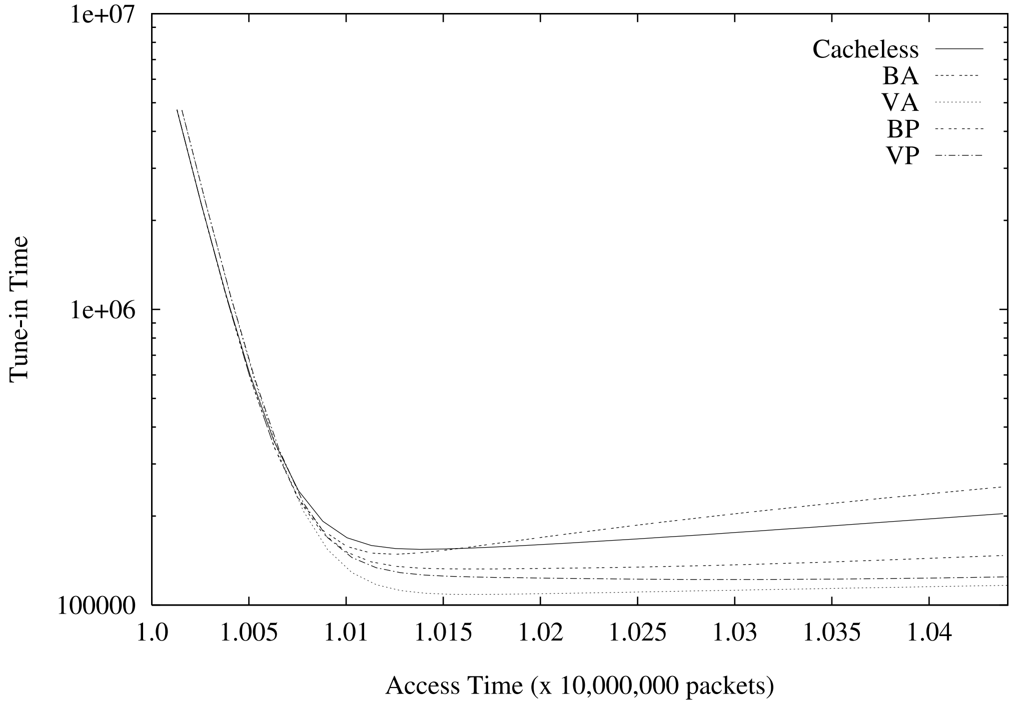

Figure 15: Tune-in time vs access time.

delay is really restricted.

In addition to the factors we discussed above, the number of information frames grouped together to generate an integrated signature also affects tune-in time performance. Its impact is two fold: on one hand, increasing the number of frames grouped together allows more compact storage of the invalidation information in packets and thus reduce its overhead; on the other hand, the information abstracted and stored into the integrated signatures will increase, thus resulting in higher false drop rates. Figure 16 compares the caching methods based on the group size. In order to more easily observe the impact of the group size on fitting invalidation information into packets, we don't combine the bit tags with integrated signatures as we did in the previous experiments. Also, we change the size of the packet to 4 bytes and the size of signatures to 32 packets. In the figure, as the group size increases, we find that the tune-in time of the bit tag policies drops initially but becomes stable later. The initial drop is due to the more compact storage of the bit tags and the reduced number of integrated signatures. However, the factor of false drops dominates later on. The version number policies perform better mainly because they have inherited cached signatures from previous connections.

Finally, we increase the percentage of the updated frames to observe its effect on caching policies. In this experiment, we set the locality of the modified frames in a frame group to 2 in order to allow the percentage of the updated frames to go up to 50%. Figure 17 shows the

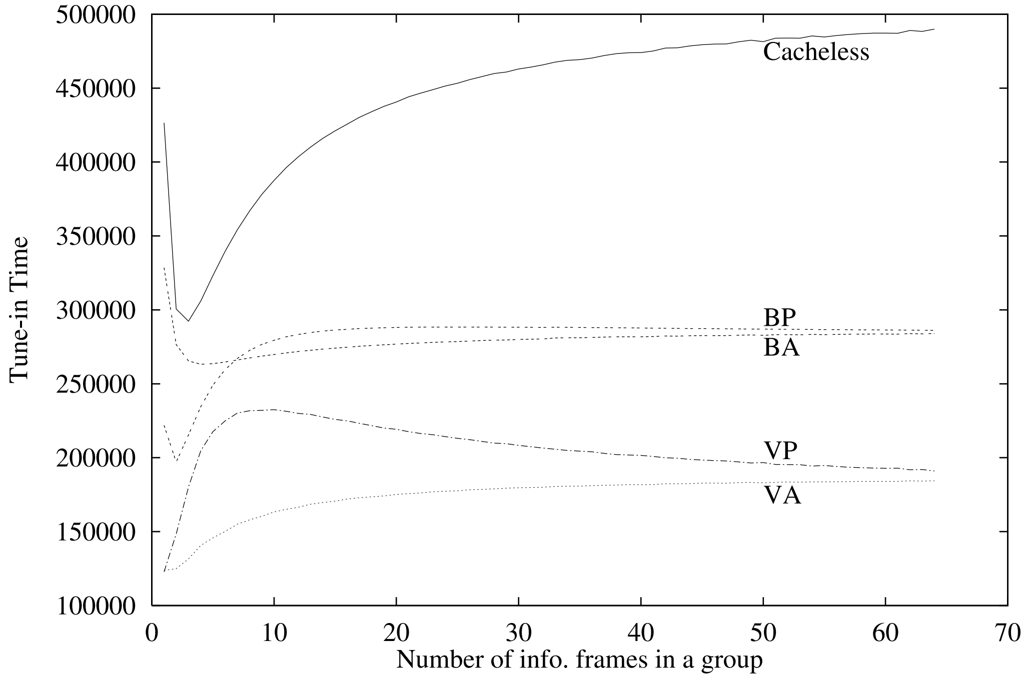

Figure 16: Tune-in time vs group size

tune-in time performance. As the update percentage increases, the caching methods using passive refreshing strategy perform better than the methods using active strategy. That's because the passive strategies only load those signatures needed for query processing but not residing in the cache, while the active policies try to maintain all of the signatures in cache.

4 Related Works

4.1 Hashing Technique

The problem of using wireless data broadcasting as a way of disseminating information to a massive number of battery powered palmtops was discussed in [13], where two hashing schemes and one flexible indexing method for organizing and accessing broadcast data were discussed.

The hash-based schemes embedded control information with data buckets, so there is no extra buckets allocated for the control information. The data buckets containing hashing function and shift pointers to the beginning of the logical buckets are called directory buckets. The first hashing scheme puts the directory buckets at the beginning of a broadcast. Missing the directory will delay the filtering process to the next broadcast. The second

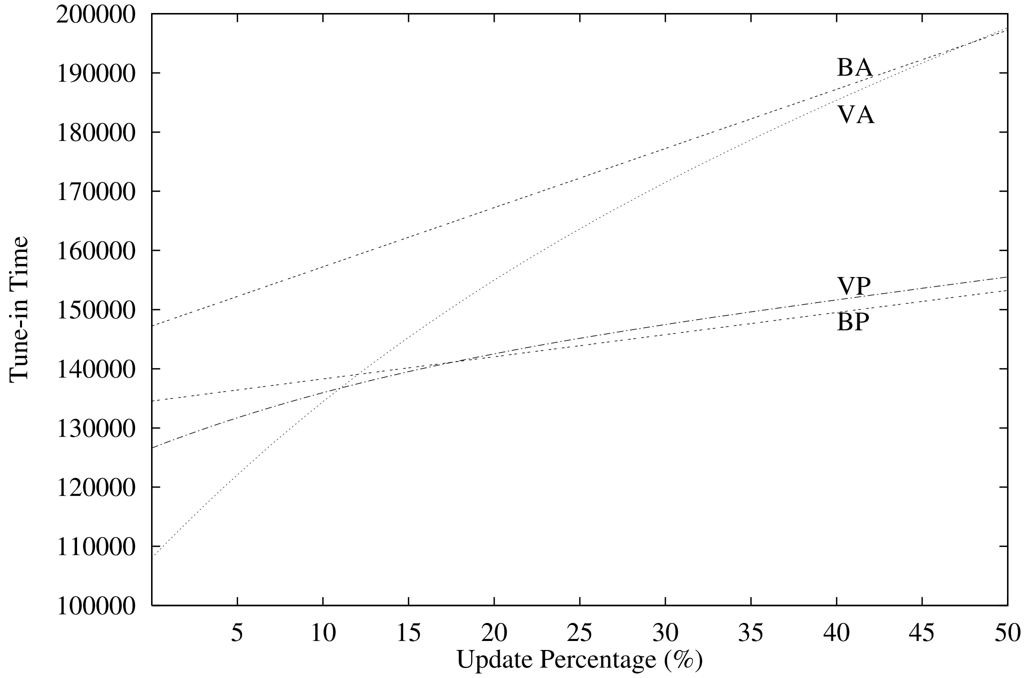

Figure 17: Tune-in time vs update percentage.

hashing scheme remedies the problem by modifying the hashing function to interleave control buckets with the other data buckets. Given the same broadcast file and the same search key, the probability of missing the directory bucket for the second scheme is much smaller than that for the first scheme. The cost model for access time has been derived for the hashing schemes. However, the tune-in time is not analysed. There is a table of data for access time and tune-in time. However, it seems that the initial probe time is not considered in the calculation of the tune-in time.

The flexible indexing method divides the sorted data records into several segments and indexes. The term 'flexible' refers to the size of segments which may be adjusted to get good access and tune-in time. The first bucket of each segment will contain the control index, which consists of binary control index and local index. The binary control index is a partial binary tree which indexes the upcoming data segments in the broadcast cycle. The local index is a linear list which indexes the data buckets in the local segment. The cost models for the tune-in time and access time are derived for the flexible index.

On selecting between the hashing scheme and the indexing method, the author suggested that hashing schemes 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, indexing method should be used.

4.2 Indexing Technique

Two methods, *(1,m) indexing* and *distributed indexing*, for organizing and accessing broadcast data were described in [14]. The paper assumes that the queries are based on the primary key. The data frames are sorted by the primary key. Also, the paper assumes that the size of the data is small enough to fit into a frame.

In the (1, m) indexing method, a tree-structured index, e.g., B^+-Tree, is created for the data frames in the broadcast cycle. The data frames are divided into m segments. A replication of the index, called the index segment, is created for each data segment. After accessing the index, the PCS may enter doze mode and wake up when the data frame arrives.

The distributed indexing is based on the observation that there is no need to replicate the entire index between successive data segments. For each data segment, there is an index segment which consists of two parts: the replicated part indexes different data segments and the non-replicated part indexes the data buckets in a data segment.

The cost models for the tune-in time and the access time are derived for both (1, m) indexing and distributed indexing method.

4.3 Caching Techniques

Acharya et. al. [2] proposed *broadcast disks* for mobile clients. The broadcast disk superimposes multiple information streams into a super-stream for broadcasting on one single channel. The idea is to interleave the information frames of different streams at some specified frequencies. The information frames from the more important streams are broadcasted more often than those from less important streams. Thus, the frequencies of various streams may be adjusted to reflect the user demands.

In addition to showing how to superimpose information streams into a broadcast channel, the paper discussed caching techniques for broadcast disks. Due to the serial and shared nature of information broadcasting, two new caching policies, *PIX* and *LIX*, are proposed. Instead of caching the hottest pages, the policies are based on the ratio of local access probability to broadcast frequency. In other words, if the hottest frames may easily be obtained on the air, they need not be cached. The best candidates for caching are those accessed frequently but difficult to obtain. PIX is not an implementable policy, because it's based on the access probabilities of frames in a mobile client. Thus, LIX is modified from LRU by taking into account of the broadcast frequencies to approximate PIX. In [20], the same authors also discussed the prefetch strategies of caching for broadcast disks. However, the results are not conclusive.

Three different cache invalidation strategies, *broadcasting timestamps, amnesic terminals,* and *signatures,* are proposed in [6]. The timestamp method periodically broadcasts the latest changes on the information frames. The PCSs listen to the report and update their caches. The amnesic terminal method periodically broadcasts the identifier of items which are changed since the last broadcast of the invalidation report. Finally, in the signature method, a set of combined signatures is generated from the information frames and broadcasted periodically. The PCS caches, along with the individual frames of interest, all of the combined signatures of subsets that include items of interest to the user. By comparing the new set of combined signatures with the cached ones, changes on the items in the cache may be identified.

The authors studied the impact of disconnection time on these strategies. As a result, the signature approach is effective for long sleepers, while the timestamp method is the best for the cases when queries are much more frequent than updates. The amnesic terminal method is the best for clients which hardly disconnect.

5 Conclusion

Since most queries on broadcast information select only a small number of information frames, indexing is very effective in reducing the tune-in time. With a reasonable false drop probability and small signature overhead, the signature schemes is excellent for information filtering in information broadcast services. Compared to traditional indexing, the signature method is particularly suitable for PCSs because it can perform realtime filtering with little processing and memory requirement.

This paper consists of two parts. The first part discusses the application of signature techniques to information filtering and the second part describes policies for caching signatures with mobile clients. We propose three signature schemes and four caching policies. We also analyse the impact of the ratio between integrated signatures and simple signatures on optimizing tune-in time for the multi-level signature scheme.

Unlike the performance consideration of traditional disk accesses, tune-in time, corresponding to battery consumption in the operation, and access time are used as performance criteria in evaluating our signature designs. The cost models for the tune-in time and access time of each of the signature schemes and caching policies have been developed and compared based on various factors.

The result shows that, with fixed signature size, the multi-level scheme has the best tune-in time performance but has the longest access time; the integrated scheme has the best average access time, but its tune-in time depends on the similarity among the information frames; the simple scheme has a fair access time and tune-in time. Compared to a broadcast channel without any indexing, all of the three schemes improve tune-in time performance dramatically with a reasonable access time overhead.

With reasonable access time delay, the multi-level signature scheme with various caching policies outperforms the same scheme without caching. The policies using version numbers are in general better than those using bit tags. As the percentage of updated frames increased, the policies using passive refreshing strategy is better than that using active strategy.

For future research, we will investigate caching policies for caches of fixed size and optimization issues involved with data and channel allocation when PCSs are capable of transmitting queries to the central server.

References

[1] B. Barriginer et al., "Infopad: A System Design for Wireless Multimedia Access," *Wireless94, Calgary*, July 1994.

[2] S. Acharya, R. Alonso, M. Franklin & S. Zdonik, "Broadcast Disks: Data Management for Asymmetric Communication Environments," Department of Computer Science, Brown University, Technical Report No. CS-94-43, December 1994.

[3] R. Alonso & S. Ganguly, "Energy Efficient Query Optimization," Matsushita Information Technology Laboratory, MITL-TR-33-92, November 1992.

[4] A. Asthana, M. Cravatts & P. Krzyzanowski, "An Indoor Wireless System for Personalized Shopping Assistance," *MOBIDATA: An Interactive Journal of Mobile Computing*, Vol. 1, No. 1, Novenber 1994.

[5] N. Ballard, "Nigel Ballard's MINI PDA Comparison Chart #47," Nigel Ballard Bournemouth, U.K., http://www.eit.com/mailinglists/zoomer/documents/pda.html, Febuary 1994.

[6] D. Barbara & T. Imielinski, "Sleepers and Workaholics: Cashing Strategies in Mobile Environments," *Proceedings of the International Conference on SIGMOD*, 1994, 1–12.

[7] T.D. Burd & R.W. Brodersen, "Energy efficient CMOS microprocessor design," *Proceedings of the Twenty-Eighth Hawaii International Conference on System Sciences*, 1995, 288–297.

[8] W.W. Chang & H.J. Schek, "A signature access method for the Starburst database system," *Proceedings of the Fifteenth International Conference on Very Large Data Bases*, Amsterdam, The Netherlands, Aug. 1989, 145–153.

[9] C. Faloutsos & S. Christodoulakis, "Signature files: an access method for documents and its analytical performance evaluation," *ACM Transactions on Office Information Systems*, Vol. 2, No. 4, Oct. 1984, 267–288.

[10] S. Ganguly & R. Alonso, "Query Optimization in Mobile Environments," Department of Computer Science, Rutgers University, Technical Report, 1993.

[11] Wireless Data Group, "Envoy: Personal Wireless Communicator," Motorola, Inc, Brochure, July 1994.

[12] T. Imielinski & B. R. Badrinath, "Mobile Wireless Computing: Solutions and Challenges in Data Management," Department of Computer Science, Rutgers University, DCS-TR-296, 1993.

[13] T. Imielinski, S. Viswanathan & B. R. Badrinath, "Power Efficiency Filtering of Data on Air," *Proceedings of the International Conference on Extending Database Technology*, 1994, 245–258.

[14] T. Imielinski, S. Viswanathan & B. R. Badrinath, "Energy Efficiency Indexing on Air," *Proceedings of the International Conference on SIGMOD*, 1994, 25–36.

[15] D.L. Lee, Y.M. Kim & G. Patel, "Efficient signature file methods for text retrieval.," *To appear in IEEE Transactions on Data and Knowledge Engineering*, 1994.

[16] S.-Y. Lee, M.-C. Yang & J.-W. Chen, "Signature file as a spatial filter for iconic image database," *Journal of Visual Languages and Computing*, Vol. 3, No. 4, December 1992, 373–397.

[17] F. Rabitti & P. Zezula, "A dynamic signature technique for multimedia databases," *Proceedings of the 13th International Conference on Research and Development in Information Retrieval*, September 1990, iv+493.

[18] S. Stiassny, "Mathematical analysis of various superimposed coding methods," *American Documentation*, Vol. 11, No. 2, February 1960, 155–169.

[19] P. Tiberio & P. Zezula, "Selecting signature files for specific applications," *Information Processing and Management* , Vol. 29, No. 4, 1993, 487–498.

[20] S. Zdonik, M. Franklin, R. Alonso & S. Acharya, "Are "Disk in the Air" Just Pie in the Sky?," *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz*, December 1994.

Contents