

# Broadcasting Dependent Data for Ordered Queries without Replication in a Multi-Channel Mobile Environment

Jiun-Long Huang and Ming-Syan Chen and Wen-Chih Peng

Department of Electrical Engineering

National Taiwan University

Taipei, Taiwan, ROC

E-mail:mschen@cc.ee.ntu.edu.tw, {jluhuang, wcpeng}@arbor.ee.ntu.edu.tw

## Abstract

*In several mobile applications, the data items broadcast are dependent upon one another. However, most prior studies on broadcasting dependent data mainly consider single broadcast channel environments. In view of this, we explore in this paper the problem of broadcasting dependent data in multiple broadcast channels. By analyzing the model of dependent data broadcasting, we derive several theoretical properties for the average access time in a multiple channel environment. In light of the theoretical results, we develop a genetic algorithm to generate broadcast programs.*

## 1 Introduction

Most prior works about broadcast program generation were under the premise that each user requests only one data item at a time and the requests for all data items are independent. However, in many real applications, there exists dependency among data items. Consider a Web page containing some images as an example. Once the user requests this Web page by a browser, the browser will request these images automatically after receiving this Web page. In this example, data allocation algorithms assuming independent data are not able to effectively optimize the performance of the broadcast programs. This phenomenon attracts a series of studies [1][3] on solving the problem of broadcasting dependent data.

In this paper we first model the problem of broadcast program generation for ordered queries in multiple channels. By analyzing the model of dependent data broadcasting, we derive several theoretical properties for the average access time in a multiple channel environment. In light of the theoretical results, we then formulate the fitness functions and design a genetic algorithm (abbreviated as GA) [2] to generate broadcast programs for ordered queries.

## 2 Preliminaries

It is assumed that the database  $D$  contains  $|D|$  data items,  $D_1, D_2, \dots, D_{|D|}$  and each data item is read-only and of equal size. From the users' perspective, a query is an indivisible request of single or multiple data items as defined below.

**Definition 1:** An *ordered* query  $Q_i = \{D_{q^i(1)}, D_{q^i(2)}, \dots, D_{q^i(|Q_i|)}\}$  is an *ordered*, non-empty subset of all data items where  $|Q_i|$  represents the number of required data items in  $Q_i$ . Note that  $1 \leq q^i(j) \leq |D|$  for all  $j$  where  $1 \leq j \leq |Q_i|$ , and  $q^i(j) = k$  represents that the  $j$ -th accessed data item in  $Q_i$  is  $D_k$ .

The query profile is the aggregation of the access behavior of all users. Formally, we have the following definition.

**Definition 2:** A query profile  $Q$  consists of a set of  $\langle Q_i, Pr(Q_i) \rangle$  pairs where  $|Q|$  indicates the number of queries in  $Q$ .  $Pr(Q_i)$  represents the probability that a query issued by users is  $Q_i$ , and  $\sum_{i=1}^{|Q|} Pr(Q_i) = 1$ .

Let  $n$  be the number of broadcast channels and  $L$  be the length of the broadcast program. The broadcast program can be defined as follows.

**Definition 3:** A broadcast program without replication<sup>1</sup>  $P$  is a *placement* of all data items in  $D$  into an  $n$  by  $L$  array where  $L = \left\lceil \frac{|D|}{n} \right\rceil$ .

We assume that  $|D| = L \times n$  without loss of generality. To facilitate the further discussion, we define a function  $f_1 : \{1, 2, \dots, |D|\} \rightarrow \{1, 2, \dots, L\}$  for each broadcast program to be an onto function which maps each data item into its placement in the broadcast program. We also define the function  $DIST(i, j)$  to represent the distance from  $D_i$

<sup>1</sup>A broadcast program without replication indicates that each data item is broadcast with equal frequency.

D <sub>1</sub>	D <sub>4</sub>	D <sub>2</sub>
D <sub>6</sub>	D <sub>3</sub>	D <sub>5</sub>

**Figure 1. An example of broadcast program**

to  $D_j$  in a broadcast program.  $DIST(i, j)$  can be determined as follows.

$$DIST(i, j) = \begin{cases} 0, & \text{if } i = j; \\ f_1(j) - f_1(i) - 1, & \text{if } i \neq j \text{ and } f_1(j) > f_1(i); \\ L - f_1(i) + f_1(j) - 1, & \text{if } i \neq j \text{ and } f_1(j) \leq f_1(i). \end{cases}$$

We take the access time as the measurement for the quality of broadcast programs. Consequently, given the number of broadcast channels, the database  $D$  and a query profile  $Q$ , the problem of broadcast program generation for ordered queries is to determine a broadcast program  $P$  which minimizes the average access time of the query profile  $Q$ .

Denote the average access time of a query  $Q_i$  as  $T_{Access}(Q_i)$  and the average access time of a query profile  $Q$  as  $T_{Access}(Q)$ . The average access time of the query profile  $Q$  can be formulated as the following equation,

$$T_{Access}(Q) = \sum_{i=1}^{|Q|} [T_{Access}(Q_i) \times Pr(Q_i)]. \quad (1)$$

### 3 Analytical Models

The access time of an arbitrary query  $Q_i$  can be decomposed into three parts: *startup time*, *wait time* and *retrieval time*. When a mobile user submits a query  $Q_i$ , the mobile device should wait until the system starts to broadcast the first required data item of  $Q_i$  (i.e.,  $D_{q^i(1)}$ ). This time interval is called the startup time. The wait time is defined as the summation of the time intervals between the moment that the mobile device completes the retrieval of the data item  $D_{q^i(j)}$  and the moment that mobile device starts to retrieve the next data item  $D_{q^i(j+1)}$ . The retrieval time is the aggregated time while the mobile device indeed reads data items from broadcast channels. It is noted that the retrieval time of a query is proportion to the number of data items required by the query.

Without loss of generality, we assume that the user submits a query on the  $m$ -th broadcast cycle. We also assume that the offset between the start time of the  $m$ -th broadcast cycle and the moment of the mobile user submits a query is a uniform distribution over  $(0, L)$ . Then, we have the following lemma.

**Lemma 1:** Denote the average startup time of a query  $Q_i$  in a broadcast program as  $T_{Startup}(Q_i)$ . We have

$$T_{Startup}(Q_i) = \frac{L}{2} \times \frac{s}{B}. \quad (2)$$

Similarly, we have the average wait time of the query  $Q_i$  in a broadcast program  $T_{Wait}(Q_i)$  below. According to the definition of wait time, we have

$$T_{Wait}(Q_i) = \left[ \sum_{k=1}^{|Q_i|-1} DIST(q^i(k), q^i(k+1)) \right] \times \frac{s}{B}. \quad (3)$$

Since the retrieval time is the aggregation of the time when the mobile device indeed reads data items from broadcast channels, the corresponding average retrieval time of the query  $Q_i$ ,  $T_{Retr.}(Q_i)$ , is

$$T_{Retr.}(Q_i) = |Q_i| \times \frac{s}{B}. \quad (4)$$

By the definition of the average access time, the average access time of the query  $Q_i$  (denoted as  $T_{Access}(Q_i)$ ) can be formulated as

$$T_{Access}(Q_i) = T_{Startup}(Q_i) + T_{Wait}(Q_i) + T_{Retr.}(Q_i).$$

Thus, Eq. (1) is rewritten as

$$T_{Access}(Q) = T_{Startup}(Q) + T_{Wait}(Q) + T_{Retr.}(Q), \quad (5)$$

where

$$T_{Startup}(Q) = \sum_{i=1}^{|Q|} T_{Startup}(Q_i) \times Pr(Q_i), \quad (6)$$

$$T_{Wait}(Q) = \sum_{i=1}^{|Q|} T_{Wait}(Q_i) \times Pr(Q_i), \quad (7)$$

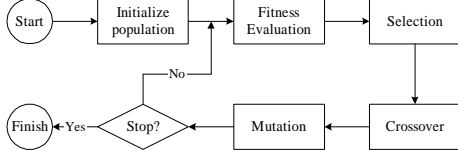
$$T_{Retr.}(Q) = \sum_{i=1}^{|Q|} T_{Retr.}(Q_i) \times Pr(Q_i). \quad (8)$$

**Lemma 2:** With a query profile  $Q$ , according to Eq. (2), Eq. (6) can be formulated as

$$T_{Startup}(Q) = \frac{L}{2} \times \frac{s}{B}. \quad (9)$$

Similarly, according to Eq. (3), Eq. (7) can be also formulated as

$$T_{Wait}(Q) = \sum_{i=1}^{|Q|} \sum_{j=1}^{|Q_i|-1} [DIST(q^i(j), q^i(j+1)) \times Pr(Q_i)] \times \frac{s}{B}.$$



**Figure 2. The flowchart of a genetic algorithm**

Consider a pair of data items  $(D_x, D_y)$  and a query  $Q_i$ .  $(D_x, D_y)$  is said to *occur* in  $Q_i$  if there exists at least one value of  $j$  so that  $q^i(j) = x$  and  $q^i(j+1) = y$ . The number of possible values of  $j$  is said to be the number of *occurrences* of  $(D_x, D_y)$  in  $Q_i$ . In order to facilitate the calculation of  $T_{Wait}(Q)$ , an auxiliary table  $AT$  is defined as follows.

**Definition 4:** An auxiliary table (referred to as table  $AT$ ) of a query profile  $Q$  is a  $|D|$  by  $|D|$  array where  $AT(x, y)$  is defined as

$$AT(x, y) = \sum_{i=1}^{|Q|} \left[ \text{the number of occurrences of } (D_x, D_y) \text{ in query } Q_i \times Pr(Q_i) \right].$$

With table  $AT$ , we have the following lemma:

**Lemma 3:** With the table  $AT$  for the query profile  $Q$ ,  $T_{Wait}(Q)$  can be formulated as

$$T_{Wait}(Q) = \sum_{j=1}^{|D|} \sum_{k=1}^{|D|} \left[ AT(j, k) \times DIST(j, k) \right] \times \frac{s}{B}. \quad (10)$$

By Lemma 3,  $T_{Wait}(Q)$  can be calculated by traversing the non-zero cells in table  $AT$  instead of scanning the whole query profile. It will greatly reduce the calculation time of  $T_{Wait}(Q)$  especially when there are many queries which require only a few data items.

In addition, according to Eq. (4), Eq. (8) is formulated as below,

$$T_{Retr.}(Q) = \left\{ \sum_{i=1}^{|Q|} \left[ |Q_i| \times Pr(Q_i) \right] \right\} \times \frac{s}{B}. \quad (11)$$

Finally, according to Eq. (5), the average access time of the query profile on a broadcast program can be obtained from the summation of Eqs. (9), (10) and (11).

## 4 Design of Genetic Algorithm

The flowchart of GAs is presented in Figure 2. By the definitions in Section 2, a broadcast program can be transformed into a list. As described before, fitness is the measurement of the quality of the chromosomes, and the GA is

designed to search the chromosome with the highest fitness (i.e, maximize the fitness). Since the goal of broadcasting dependent data is to minimize the average access time of the given query profile, the fitness function is defined as  $Fitness(P) = \frac{1}{T_{Access}(Q)}$ .

In accordance with Eqs. (9) and (11), the values of  $T_{Wait}(Q)$  and  $T_{Retr.}(Q)$  are independent of the broadcast programs, and therefore, they can be calculated in advance. Similarly, according to Definition 4, table  $AT$  only depends on the query profile and can also be constructed after reading the query profile. Finally,  $T_{Access}(Q)$  can be calculated by the following procedure.

### Procedure CalAccessTime( $Q, P$ )

**Input:** A query profile  $Q$  and a broadcast program  $P$ .

**Output:**  $T_{Access}(Q)$  over the broadcast program  $P$ .

- 1:  $T_{Wait}(Q) \leftarrow 0$ ;
- 2: **for**  $j = 1$  to  $|D|$  **do**
- 3:   **for**  $k = 1$  to  $|D|$  **do**
- 4:      $T_{Wait}(Q) \leftarrow T_{Wait}(Q) + AT(j, k) \times DIST(j, k)$ ;
- 5:   **end for**
- 6: **end for**
- 7:  $T_{Wait}(Q) \leftarrow T_{Wait}(Q) \times \frac{s}{B}$ ;
- 8: **return**  $T_{Startup}(Q) + T_{Wait}(Q) + T_{Retr.}(Q)$ ;

With above procedure, one can effectively optimize broadcast programs without replication for ordered queries. In some prior works of broadcast program generation for independent data, it has been shown that broadcasting frequently accessed data items with high frequencies will be able to further optimize the broadcast programs. As a result, broadcast program generation for ordered queries with replication is a problem worth study.

## Acknowledgement

The authors are supported in part by the Ministry of Education Project No. 89E-FA06-2-4-7 and the National Science Council, Project No. NSC 91-2213-E-002-034 and NSC 91-2213-E-002-045, Taiwan, Republic of China.

## References

- [1] Y. C. Chehadeh, A. R. Hurson, and M. Kavehrad. Object Organization on a Single Broadcast Channel in the Mobile Computing Environment. *Multimedia Tools and Applications*, 9(1):69–94, July 1999.
- [2] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, 1989.
- [3] A. R. Hurson, Y. C. Chehadeh, and J. Hannan. Object Organization on Parallel Broadcast Channels in a Global Information Sharing Environment. In *Proceedings of the 19th IEEE International Performance, Computing, and Communications Conference*, pages 347–353, February 2000.