NORTH-HOLLAND

Informatics and
Computer Science

Some Properties of Optimal Cartesian Product Files for Orthogonal Range Queries

ANNIE Y. H. CHOU
WEI-PANG YANG

Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 300, Republic of China

and

CHIN-CHEN CHANG

Department of Computer Science and Information Engineering, National Chung Cheng University, Hsinchu, Taiwan 621, Republic of China

ABSTRACT

Cartesian product file (CPF) has been proposed as a good multi-attribute file structure. Although designing an optimal CPF for partial match queries (PMQs) has been proven to be NP-hard, some useful properties have been studied for PMQs to help the work. However, a good CPF for PMQs may not be beneficial for orthogonal range queries (ORQs). Therefore, in this paper, we intend to study properties that help the design of a good CPF for ORQs. We found that the problem of designing the optimal CPF for ORQs is related to the problem of finding a minimal-*f* *N*-tuple. We will also show some theories of minimal-*f* *N*-tuples and develop a method for generating a minimal-*f* *N*-tuple. Finally, we will present some properties of the optimal CPF for ORQs from the theories of minimal-*f* *N*-tuples.

1. INTRODUCTION

The need for efficient retrieval methods in large databases has increased recently. Files that store large amounts of data are often used in many applications. In such applications, the size of data is usually too large to be wholly stored in primary memory. In practice, files are usually divided into buckets and then stored in a disk. Therefore, the distribution of a file over multiple buckets directly influences the performance of retrieval.

0020-0255/96/$15.00
SSDI 0020-0255(95)00284-7

A multi-attribute file (MAF) is a collection of records characterized by more than one attribute. An orthogonal range query (ORQ) is a query of the following form: retrieve all records satisfying $(x_1, x_2, \ldots, x_N)$, where x_i is either a range $[l_i, u_i]$ contained in the domain of the ith attribute, or is unspecified (denoted by $*$). For instance, in a two-attribute file system $\{a,b\} \times \{1,2,3,4\}$, $(*,[1,3])$ denotes an ORQ which retrieves all records with the first attribute being either "a" or "b" and the second attribute being in the range [1,3]. If $l_i = u_i$ for all specified x_i, then the ORQ reduces to a partial match query (PMQ).

Since each time the disk is accessed, an entire bucket is brought into primary memory, we can simply measure the retrieval performance of a file structure by counting the average number of buckets (ANB) to be examined over all possible queries. Therefore, the optimal MAF system design problem can be stated as follows: given a set of multi-attribute records, and a fixed number of buckets, arrange the records into buckets such that the ANB to be examined over all possible queries will be as small as possible [3].

In prior research work, great progress has been achieved on designing optimal MAF systems for PMQs. Tang, Buehrer, and Lee [14] showed that this problem is NP-hard, and many heuristic methods have been proposed to solve it, for example, the string homomorphism hashing (SHH) method by Rivest [12], the multi-key hashing (MKH) method by Rothnie and Lozano [13], the multi-dimensional directory (MDD) method by Liou and Yao [11], the optimal Cartesian product file (CPF) design method by Chang, Du, and Lee [4], the Greedy File design method proposed by Chou, Yang, and Chang [9], and so on. Recently, the optimal disk allocation for PMQs was also proposed by Abdel-Ghaffar and El Abbadi [1]. However, the study of that for ORQs is far less progressive [6–8].

In our work, we paid particular attention to the design of Cartesian product files (CPFs) for ORQs. The CPF is a hashed file in essence, and the choice of hash functions (i.e., partition forms) determines the performance of retrieval. Chou et al. have derived the performance formula of CPFs for ORQs in [8]. However, as we should see, the performance formula is too complicated to apply. In this paper, we intend to find some properties to determine which partition form is better, instead of directly calculating their performance.

Chang et al. [3] presented some properties of CPFs for PMQs and showed that the problem of designing the optimal CPF for PMQs is related to the problem of finding a minimal N-tuple, where the N-tuple exactly corresponds to the *partition size form* of a CPF. In this paper, we intend to study whether the problem of designing the optimal CPF for ORQs is also related to the problem of finding a minimal N-tuple.

Fortunately, we are able to show that the answer is yes, although the N-tuple now corresponds to the *partition form* of a CPF. Besides, there exist exceptions in which the minimal N-tuple is not always the optimal partition form of a CPF for ORQs. Therefore, we define a new term, *minimal-f N-tuple*, to cover minimal N-tuples that describe the properties of optimal CPFs for ORQs. Then, we will also show some theories of minimal-f N-tuples and develop a method for generating a minimal-f N-tuple. Finally, we will present some properties of the optimal CPF from the theories of minimal-f N-tuples.

In the next section, we will review the CPF concept and its performance formula for ORQs. In Section 3, we will define a minimal-f N-tuple, then derive some theories of minimal-f N-tuples and propose an algorithm for finding a minimal-f N-tuple. In Section 4, the problem of designing the optimal CPF for ORQs is shown to be related to the problem of finding a minimal N-tuple with *reverse order* (defined later) or finding a minimal-f N-tuple. Finally, conclusions are given in Section 5.

2. A REVIEW OF CPFS FOR ORQS

The CPF concept was originally proposed by Lin, Lee, and Du [10]. They also pointed out that file systems designed using the SHH, MKH, and MDD methods are all CPFs. The CPFs are defined as follows.

DEFINITION 2.1 [10]. An N-attribute CPF is a file in which each domain D_i is divided into m_i equal-sized subdomains $D_{i1}, D_{i2}, \ldots, D_{im_i}$, and all records in each bucket are of the form $D_{1s_1} \times D_{2s_2} \times \cdots \times D_{Ns_N}$, where $1 \leqslant s_i \leqslant m_i$ and $1 \leqslant i \leqslant N$. The *partition form* of this CPF is denoted as $\langle m_1, m_2, \ldots, m_N \rangle$, where m_i is the number of partitions in the ith domain. The *partition size form* of this CPF is denoted as $(z_1, z_2, \ldots, z_N)$, where $z_i = |D_i|/m_i$ is ith subdomain size and is an integer.

To measure the performance of CPFs for ORQs, we have derived the following formula to directly evaluate the ANB of a CPF over all possible ORQs in [8]. Let F be an N-attribute CPF with partition form $\langle m_1, m_2, \ldots, m_N \rangle$ and d_i be the domain size of the ith attribute. If the probabilities of all occurring queries are equal, then the ANB of this CPF over all possible ORQs is as follows.

$$ANB(\mathrm{ORQ}) = \frac{NB}{NOQ}\left(\frac{1}{6}\right)^N \prod_{i=1}^{N}\left(-\left(\frac{d_i}{m_i}\right)^2 + \frac{3d_i}{m_i} + \frac{3d_i^2}{m_i} + d_i^2 + 6\right),$$

where NB is the number of buckets $= \prod_{i=1}^{N} m_i$, and NOQ is the total number of ORQs $= (\frac{1}{2})^N \prod_{i=1}^{N} (d_i^2 + d_i + 2)$.

EXAMPLE 2.1. Consider a two-attribute file with $D_1 = \{a, b\}$ and $D_2 = \{1,2,3,4,5,6\}$. Suppose we have six buckets, and each can hold two records. If $D_{11} = \{a, b\}$, $D_{21} = \{1\}$, $D_{22} = \{2\}$, $D_{23} = \{3\}$, $D_{24} = \{4\}$, $D_{25} = \{5\}$, and $D_{26} = \{6\}$. Then the following file F is a CPF.

$$
\begin{aligned}
&\text{Bucket 1:} \quad D_{11} \times D_{21} = \{(a,1),(b,1)\},\\
&\text{Bucket 2:} \quad D_{11} \times D_{22} = \{(a,2),(b,2)\},\\
&\text{Bucket 3:} \quad D_{11} \times D_{23} = \{(a,3),(b,3)\},\\
&\text{Bucket 4:} \quad D_{11} \times D_{24} = \{(a,4),(b,4)\},\\
&\text{Bucket 5:} \quad D_{11} \times D_{25} = \{(a,5),(b,5)\},\\
&\text{Bucket 6:} \quad D_{11} \times D_{26} = \{(a,6),(b,6)\}.
\end{aligned}
$$

According to our formula, the corresponding ANB(ORQ) of this CPF is 2.82 and its partition form $\langle m_1, m_2 \rangle$ is $\langle 1,6 \rangle$. However, if $D_{11} = \{a\}, D_{12} = \{b\}$, $D_{21} = \{1,2\}$, $D_{22} = \{3,4\}$, and $D_{23} = \{5,6\}$. Then the following file F' is also a CPF.

$$
\begin{aligned}
&\text{Bucket 1:} \quad D_{11} \times D_{21} = \{(a,1),(a,2)\},\\
&\text{Bucket 2:} \quad D_{11} \times D_{22} = \{(a,3),(a,4)\},\\
&\text{Bucket 3:} \quad D_{11} \times D_{23} = \{(a,5),(a,6)\},\\
&\text{Bucket 4:} \quad D_{12} \times D_{21} = \{(b,1),(b,2)\},\\
&\text{Bucket 5:} \quad D_{12} \times D_{22} = \{(b,3),(b,4)\},\\
&\text{Bucket 6:} \quad D_{12} \times D_{23} = \{(b,5),(b,6)\}.
\end{aligned}
$$

F' has partition form $\langle 2,3 \rangle$ and the corresponding ANB(ORQ) is 2.73. Since there are only two CPF partition forms $\langle 1,6 \rangle$ and $\langle 2,3 \rangle$, we know F' is the optimal CPF with partition form $\langle 2,3 \rangle$.

In summary, the problem of designing an optimal CPF for ORQs can be formally stated as follows: given a set of integers $d_1, d_2, \ldots, d_N$ and NB, our work is to find N integers $m_1, m_2, \ldots, m_N$ that satisfy the following conditions:

(1) $\prod_{i=1}^{N} m_i = NB$,
(2) d_i/m_i is an integer for $i = 1, 2, \ldots, N$, and
(3) $ANB(\mathrm{ORQ})$ is minimal, or $\prod_{i=1}^{N}(-(d_i/m_i)^2 + 3d_i/m_i + 3d_i^2/m_i + d_i^2 + 6)$ is minimal.

Since an integer can be factored into a finite number of different N-tuples, there are a finite number of feasible solutions, and we can conduct an exhaustive search. That is, given $m_1, m_2, \ldots, m_N$, we can calculate

$$\prod_{i=1}^{N}\left(-\left(\frac{d_i}{m_i}\right)^2 + \frac{3d_i}{m_i} + \frac{3d_i^2}{m_i} + d_i^2 + 6\right).$$

We then choose the m_i's that minimize the above formula. However, we shall show that an exhaustive search through all possible solutions can be avoided. Consider Example 2.1 again. The first solution of the problem is $\langle 1,6\rangle$. $\langle 1,6\rangle$ can be transformed into $\langle 2,3\rangle$ without affecting the feasibility of the solution. However, this transformation decreases the value of $\prod_{i=1}^{2}(-(d_i/m_i)^2 + 3d_i/m_i + 3d_i^2/m_i + d_i^2 + 6)$. Let us now consider the following problem: given two N-tuples $(m_1, m_2, \ldots, m_N)$ and $(d_1, d_2, \ldots, d_N)$, where $m_i, d_i \in \mathcal{N}$ and $m_i | d_i$, for $1 \leqslant i \leqslant N$, can we transform $(m_1, m_2, \ldots, m_N)$ into another N-tuple $m'_1, m'_2, \ldots, m'_N)$ such that $m'_i \in \mathcal{N}$ and $m'_i | d_i$, for $1 \leqslant i \leqslant N$, and $\prod_{i=1}^{N} m'_i = \prod_{i=1}^{N} m_i$, but the value of $\prod_{i=1}^{N}(-(d_i/m'_i)^2 + 3d_i/m'_i + 3d_i^2/m'_i + d_i^2 + 6)$ is smaller than the value of $\prod_{i=1}^{N}(-(d_i/m_i)^2 + 3d_i/m_i + 3d_i^2/m_i + d_i^2 + 6)$? In the next section, we will develop some theories to answer this problem.

3. SOME THEORIES OF MINIMAL-f N-TUPLES

In the rest of this paper, whenever we mention an N-tuple $(a_1, a_2, \ldots, a_N)$, we shall assume that a_i is an integer.

DEFINITION 3.1 [3]. An N-tuple $(a_1, a_2, \ldots, a_N)$ is called an *N-tuple of C* if $\prod_{i=1}^{N} a_i = C$.

DEFINITION 3.2. An N-tuple $(a_1, a_2, \ldots, a_N)$ is called a *factor* of $(d_1, d_2, \ldots, d_N)$ if $a_i | d_i$, for $1 \leqslant i \leqslant N$.

DEFINITION 3.3. A 2-tuple (a_1, a_2) is called a *minimal* 2-tuple with respect to (d_1, d_2) if (a_1, a_2) is a factor of (d_1, d_2) and for all other factors (a'_1, a'_2) with $a'_1 a'_2 = a_1 a_2$, $a_1 + a_2 < a'_1 + a'_2$.

DEFINITION 3.4. An N-tuple $(a_1, a_2, \ldots, a_N)$ is called a minimal N-tuple of C with respect to $(d_1, d_2, \ldots, d_N)$ if $\prod_{i=1}^{N} a_i = C$ and for $1 \leqslant i,\ j \leqslant N$, (a_i, a_j) is a minimal 2-tuple with respect to (d_i, d_j).

EXAMPLE 3.1. The 3-tuple (2, 12, 3) is not a minimal 3-tuple of 72 with respect to (12, 24, 6), because (2, 12) and (12, 3) are not minimal 2-tuples with respect to the corresponding 2-tuples (12, 24) and (24, 6), respectively. The 3-tuple (3, 4, 6) is a minimal 3-tuple with respect to (12, 24, 6), because each pair of this 3-tuple is a minimal 2-tuple.

DEFINITION 3.5. Let $S = (a_1, a_2, \ldots, a_N)$ be a factor of $R = (d_1, d_2, \ldots, d_N)$; we define

$$f_R(S) = \prod_{i=1}^{N} \left(-\left(\frac{d_i}{a_i}\right)^2 + \frac{3d_i}{a_i} + \frac{3d_i^2}{a_i} + d_i^2 + 6 \right).$$

When S is not a factor of R, we define $f_R(S) = \infty$.

DEFINITION 3.6. A 2-tuple $S = (a_1, a_2)$ is called a *minimal-f* 2-tuple with respect to R if S is a factor of R and for all other factors $T = (a'_1, a'_2)$ with $a'_1 a'_2 = a_1 a_2$, $f_R(S) < f_R(T)$.

DEFINITION 3.7. An N-tuple $(a_1, a_2, \ldots, a_N)$ is called a minimal-f N-tuple of C with respect to $(d_1, d_2, \ldots, d_N)$ if $\prod_{i=1}^{N} a_i = C$ and for $1 \leqslant i$, $j \leqslant N$, (a_i, a_j) is a minimal-f 2-tuple with respect to (d_i, d_j).

3.1 MINIMAL-f 2-TUPLES

Before discussing the general theories, we shall first discuss the theories for minimal-f 2-tuples. For example, is a minimal 2-tuple (a_1, a_2) with respect to (d_1, d_2) also a minimal-f 2-tuple? In the following, we will show the answer is yes except when $a_1 a_2 = 4$ or $\min\{d_1, d_2\} \leqslant 8$.

THEOREM 3.1. *For a 2-tuple R, and its two factors $S = (a_1, a_2)$ and $T = (a'_1, a'_2)$, if $a_1 a_2 = a'_1 a'_2 \neq 4$ and $a_1 + a_2 < a'_1 + a'_2$, then $f_R(S) < f_R(T)$.*

Proof. We defer the proof until Appendix A.

COROLLARY 3.1. *If $S = (a_1, a_2)$, $a_1 a_2 \neq 4$, is a minimal 2-tuple with respect to R, then $f_R(S)$ is the smallest among all $f_R(T)$, where $T = (a'_1, a'_2)$ is a 2-tuple of $a_1 a_2$ and $a_1 + a_2 < a'_1 + a'_2$.*

DEFINITION 3.8. We say $(a_1, a_2, \ldots, a_N)$ is in *reverse order* with respect to $(d_1, d_2, \ldots, d_N)$ if $(a_i - a_j)(d_i - d_j) < 0$ for all $1 \leqslant i,\ j \leqslant N$.

DEFINITION 3.9. Let $S=(a_1,a_2)$. We define $S^{-1}=(a_2,a_1)$ as the *inverse* of S.

THEOREM 3.2. *If a 2-tuple S is a factor of $R=(d_1,d_2)$ and in reverse order with respect to R, and* $\min\{d_1,d_2\}\geqslant 9$, *then* $f_R(S)<f_R(S^{-1})$.

Proof. Let $S=(a_1,a_2)$. If $S^{-1}=(a_2,a_1)$ is not a factor of R, then the theorem obviously holds. Let us consider the case in which S^{-1} is a factor of R. Since S is in reverse order with respect to R, we can set $a_1>a_2$, $9\leqslant d_1<d_2$, $d_1=px_1$, and $d_2=px_2$, where $p=lcm(a_1,a_2)$ and $x_1<x_2$. Let integers $l_1=p/a_1$ and $l_2=p/a_2$. By $a_1>a_2$, we have $l_1<l_2$. Then

$$f_R(S)=\left(-l_1^2x_1^2+3l_1x_1+3l_1px_1^2+p^2x_1^2+6\right)$$

$$\times\left(-l_2^2x_2^2+3l_2x_2+3l_2px_2^2+p^2x_2^2+6\right),$$

$$f_R(S^{-1})=\left(-l_2^2x_1^2+3l_2x_1+3l_2px_1^2+p^2x_1^2+6\right)$$

$$\times\left(-l_1^2x_2^2+3l_1x_2+3l_1px_2^2+p^2x_2^2+6\right),$$

and

$$f_R(S)-f_R(S^{-1})=3(l_1-l_2)(x_2-x_1)E,$$

where

$$E=l_1l_2x_1x_2+2(l_1+l_2)(x_1+x_2)+p^2x_1x_2-6(1+p(x_1+x_2)).$$

Since $l_1<l_2$ and $x_2>x_1$, $E>0$ holds if and only if $f_R(S)<f_R(S^{-1})$. We now prove $E>0$ as follows.

(1) Let us consider the case in which $x_1=1$.

$$E=l_1l_2x_2+2(l_1+l_2)(1+x_2)+p^2x_2-6(1+p(1+x_2))$$

$$=\underbrace{l_1l_2x_2+2(l_1+l_2)(1+x_2)-6}_{>0}+\underbrace{p((p-6)x_2-6)}_{\geqslant 0}$$

$$>0,\quad \text{by } 1\leqslant l_1<l_2,\quad x_2\geqslant 2\quad \text{and}\quad p=d_1/x_1\geqslant 9.$$

(2) Let us consider the case in which $x_1 \geqslant 2$.

$$E = l_1 l_2 x_1 x_2 + 2(l_1 + l_2)(x_1 + x_2) + p^2 x_1 x_2 - 6(1 + p(x_1 + x_2))$$

$$= \underbrace{l_1 l_2 x_1 x_2 + 2(l_1 + l_2)(x_1 + x_2) - 42}_{\geqslant 0} + \underbrace{(d_1 - 6)(d_2 - 6)}_{> 0}$$

$$> 0, \quad \text{by } 1 \leqslant l_1 < l_2, \quad 2 \leqslant x_1 < x_2 \quad \text{and} \quad 9 \leqslant d_1 < d_2.$$

By (1) and (2), we complete the proof. □

COROLLARY 3.2. *If $S = (a_1, a_2)$, $a_1 a_2 \neq 4$, is a minimal 2-tuple and in reverse order with respect to $R = (d_1, d_2)$, and* $\min\{d_1, d_2\} \geqslant 9$, *then $f_R(S)$ is the smallest among all $f_R(T)$, where T is a 2-tuple of $a_1 a_2$. Furthermore, S is a minimal-f 2-tuple with respect to R.*

Proof. This follows directly from Corollary 3.1 and Theorem 3.2. □

We can find a minimal-f 2-tuple by employing a minimal 2-tuple. Although Corollary 3.1 holds only when $a_1 a_2 \neq 4$, the unsatisfied cases simply include (1,4), (4,1) and (2,2). Hence, when $a_1 a_2 = 4$, we can get a minimal-f 2-tuple by directly calculating the value of $f_R(S)$. As for the deficiency of Theorem 3.2, we can also directly calculate the value of $f_R(S)$ and $f_R(S^{-1})$ as $\min\{d_1, d_2\} \leqslant 8$. Therefore, we have the following algorithm to generate a minimal-f 2-tuple.

ALGORITHM 1. An algorithm to find a minimal-f 2-tuple of C with respect to (d_1, d_2).

Input: C and a 2-tuple $R = (d_1, d_2)$.

Output: A minimal-f 2-tuple of C with respect to (d_1, d_2).

STEP 1. If $C = 4$, then make comparisons to obtain the 2-tuple S^* with the smallest $f_R(S')$, where all possible 2-tuples S' include (2,2), (1,4), and (4,1), and go to Step 6.

STEP 2. Find (p, q), where $pq = C$ and (p, q) is a minimal 2-tuple with respect to (d_1, d_2).

STEP 3. If $d_1 = d_2$ or (q, p) is not a factor of (d_1, d_2), then set $S^* = (p, q)$ and go to Step 6.

STEP 4. If (p, q) is in reverse order with respect to (d_1, d_2) and $\min\{d_1, d_2\} \geqslant 9$, then set $S^* = (p, q)$ and go to Step 6.

STEP 5. Set $S=(p,q)$. Compare S with S^{-1} by evaluating $f_R(S)$ and $f_R(S^{-1})$, and set S^* to be the 2-tuple with the smallest $f_R(S')$, where $S' \in \{S, S^{-1}\}$.

STEP 6. S^* is a minimal-f 2-tuple with respect to R.

3.2 *MINIMAL-f N*-TUPLES

For an N-tuple $(d_1, d_2, \ldots, d_N)$, we can transform an N-tuple $(a_1, a_2, \ldots, a_N)$ with $\prod_{i=1}^N a_i | \prod_{i=1}^N d_i$ into a minimal-f N-tuple with respect to $(d_1, d_2, \ldots, d_N)$ by employing Algorithm 1. The detailed process is described in the following algorithm.

ALGORITHM 2. An algorithm that transforms an N-tuple $(a_1, a_2, \ldots, a_N)$ into a minimal-f N-tuple with respect to $(d_1, d_2, \ldots, d_N)$.

Input: Two N-tuples $(a_1, a_2, \ldots, a_N)$ and $(d_1, d_2, \ldots, d_N)$, where $\prod_{i=1}^N a_i | \prod_{i=1}^N d_i$.

Output: A minimal-f N-tuple of $\prod_{i=1}^N a_i$ with respect to $(d_1, d_2, \ldots, d_N)$.

STEP 1. If $(a_1, a_2, \ldots, a_N)$ is a factor of $(d_1, d_2, \ldots, d_N)$, then set $a'_j = a_j$, for $j = 1, 2, \ldots, N$, and go to Step 3.

STEP 2. Set $a'_1 = \gcd(d_1, \prod_{i=1}^N a_i)$ and $a'_j = \gcd(d_j, \prod_{i=1}^N a_i / \prod_{i=1}^{j-1} a'_i)$, for $j = 2, 3, \ldots, N$, to obtain a factor $(a'_1, a'_2, \ldots, a'_N)$ of $(d_1, d_2, \ldots, d_N)$.

STEP 3. $\pi_0 \leftarrow \begin{pmatrix} 1 & 2 & \cdots & N \\ 1 & 2 & \cdots & N \end{pmatrix}$, $i \leftarrow N$, and $j \leftarrow N-1$.

STEP 4. $C = a'_j a'_i$. Find (p,q) by Algorithm 1, where $pq = C$ and (p,q) is a minimal-f 2-tuple with respect to $(d_{\pi_0(j)}, d_{\pi_0(i)})$.

STEP 5. If $(p,q) = (a'_j, a'_i)$, then go to Step 8.

STEP 6. Reorder $(a'_1, a'_2, \ldots, p, q, \ldots, a'_N)$ to obtain a new N-tuple $(a'_1, a'_2, \ldots, a'_N)$ such that $a'_{k-1} \leqslant a'_k$ and $a'_k = a'_{\pi_1^{-1}(k)}$, where π_1 is a permutation of $\{1, 2, \ldots, N\}$.

STEP 7. $\pi_0 \leftarrow \pi_1 \cdot \pi_0$, and return to Step 4.

STEP 8. If $j \neq 1$, then $j \leftarrow j-1$, and return to Step 4.

STEP 9. If $i \neq 2$, then $i \leftarrow i-1$, $j \leftarrow N$, and return to Step 4.

STEP 10. $(a'_1, a'_2, \ldots, a'_N)$ is a minimal-f N-tuple with respect to $(d_{\pi_0(1)}, d_{\pi_0(2)}, \ldots, d_{\pi_0(N)})$. Let $a^*_i = a'_{\pi_0^{-1}(i)}$. Thus, $(a^*_1, a^*_2, \ldots, a^*_N)$ is a minimal-f N-tuple of $\prod_{i=1}^N a_i$ with respect to $(d_1, d_2, \ldots, d_N)$.

EXAMPLE 3.2. Consider $(a_1, a_2, a_3) = (2, 12, 3)$ and $(d_1, d_2, d_3) = (8, 12, 6)$. Then, the process of Algorithm 2 is as follows.

1. Since $(2, 12, 3)$ is a factor of $(8, 12, 6)$, set $(a'_1, a'_2, a'_3) = (2, 12, 3)$.
2. Go to Step 3. $\pi_0 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$, $i = 3$, and $j = 2$.
3. $C = a'_2 a'_3 = 12 \times 3 = 36$. We find $(6, 6)$ as the minimal-f 2-tuple with respect to $(d_{\pi_0(2)}, d_{\pi_0(3)}) = (d_2, d_3)$ by Algorithm 1.
4. Reorder. We obtain $(2, 6, 6)$ and $\pi_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$.
5. $\pi_0 = \pi_1 \cdot \pi_0 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$, and return to Step 4.
6. $C = a'_2 a'_3 = 6 \times 6 = 36$. We find $(6, 6)$ as the minimal-f 2-tuple with respect to $(d_{\pi_0(2)}, d_{\pi_0(3)}) + (d_2, d_3)$ by Algorithm 1.
7. Go to Step 8. Since $j = 2 \neq 1$, $j \leftarrow 1$ and return to Step 4.
8. $C = a'_1 a'_3 = 2 \times 6 = 12$. We find $(4, 3)$ as the minimal-f 2-tuple with respect to $(d_{\pi_0(1)}, d_{\pi_0(3)}) = (d_1, d_3)$ by Algorithm 1.
9. Reorder. We obtain $(3, 4, 6)$ and $\pi_1 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$.
10. $\pi_0 = \pi_1 \cdot \pi_0 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$.
11. Since every 2-tuple in $(3, 4, 6)$ is a minimal-f 2-tuple, $(3, 4, 6)$ is a minimal-f 3-tuple with respect to $(d_{\pi_0(1)}, d_{\pi_0(2)}, d_{\pi_0(3)}) = (d_3, d_1, d_2)$. That is, $(4, 6, 3)$ is a minimal-f 3-tuple with respect to (d_1, d_2, d_3).

DEFINITION 3.10 [3]. Let $S = (a_1, a_2, \ldots, a_N)$, $T = (a'_1, a'_2, \ldots, a'_N)$, $R = (d_1, d_2, \ldots, d_N)$, $\prod_{i=1}^N a_i = \prod_{i=1}^N a'_i$, and both S and T be factors of R. If there exists an i such that $a_i = pq$ and $a'_i = q$, and a j such that $a'_j = pa_j$, and for all k, $k \neq i, j$, $a'_k = a_k$, then T is a *pq-transformation* of S for R.

DEFINITION 3.11. Let T be a pq-transformation of S for R. If $f_R(T) < f_R(S)$, then T is an *effective pq-transformation* of S for R.

Recall that in Example 3.2, the 3-tuple $(2, 12, 3)$ is transformed as follows.

$$(2, 12, 3) \to (2, 6, 6) \to (4, 6, 3).$$

Their corresponding $f_R(S)$'s are 2436480, 2249856, and 2150400, respectively. That is, each transformation gradually reduces the corresponding $f_R(S)$. We show that such transformation is an effective pq-transformation as follows.

LEMMA 3.1. *Let $S = (a_1, a_2, \ldots, a_N)$, $T = (a'_1, a'_2, \ldots, a'_N)$, and $R = (d_1, d_2, \ldots, d_N)$. Let T be a pq-transformation of S for R. If in this pq-trans-*

formation,

$$\prod_{k=i,j}\left(-\left(\frac{d_k}{a'_k}\right)^2+\frac{3d_k}{a'_k}+\frac{3d_k^2}{a'_k}+d_k^2+6\right)$$

$$<\prod_{k=i,j}\left(-\left(\frac{d_k}{a_k}\right)^2+\frac{3d_k}{a_k}+\frac{3d_k^2}{a_k}+d_k^2+6\right),$$

then T is an effective pq-transformation of S for R.

Proof. Since T is a pq-transformation of S, there exists an i such that $a_i=pq$ and $a'_i=q$, and a j such that $a'_j=pa_j$. Therefore,

$$\begin{aligned}
&f_R(T)-f_R(S)\\
&=\left[\left(-\left(\frac{d_i}{a'_i}\right)^2+\frac{3d_i}{a'_i}+\frac{3d_i^2}{a'_i}+d_i^2+6\right)\left(-\left(\frac{d_j}{a'_j}\right)^2+\frac{3d_i}{a'_j}+\frac{3d_j^2}{a'_j}+d_j^2+6\right)\right.\\
&\quad\left.-\left(-\left(\frac{d_i}{a_i}\right)^2+\frac{3d_i}{a_i}+\frac{3d_i^2}{a_i}+d_i^2+6\right)\left(-\left(\frac{d_j}{a_j}\right)^2+\frac{3d_i}{a_j}+\frac{3d_j^2}{a_j}+d_j^2+6\right)\right]\\
&\quad\times\prod_{\substack{k=1\\k\neq i,j}}\left(-\left(\frac{d_k}{a_k}\right)^2+\frac{3d_k}{a_k}+\frac{3d_k^2}{a_k}+d_k^2+6\right)\\
&=\left(\prod_{k=i,j}\left(-\left(\frac{d_k}{a'_k}\right)^2+\frac{3d_k}{a'_k}+\frac{3d_k^2}{a'_k}+d_k^2+6\right)\right.\\
&\quad\left.-\prod_{k=i,j}\left(-\left(\frac{d_k}{a_k}\right)^2+\frac{3d_k}{a_k}+\frac{3d_k^2}{a_k}+d_k^2+6\right)\right)\\
&\quad\times\prod_{\substack{k=1\\k\neq i,j}}^{N}\left(-\left(\frac{d_k}{a_k}\right)^2+\frac{3d_k}{a_k}+\frac{3d_k^2}{a_k}+d_k^2+6\right)\\
&<0.
\end{aligned}$$

Hence T is an effective pq-transformation of S for R. □

THEOREM 3.3. *Let $S=(a_1,a_2,\ldots,a_N)$, $R=(d_1,d_2,\ldots,d_N)$, and $\Pi_{i=1}^{N}a_i|\Pi_{i=1}^{N}d_i$. If S is not a minimal-f N-tuple with respect to R, then S can be converted into S^* such that S^* is a minimal-f N-tuple of $\Pi_{i=1}^{N}a_i$ with respect to R and $f_R(S^*)<f_R(S)$.*

Proof. Note that in Algorithm 2, the algorithm always terminates and produces an N-tuple in which every pair (a_i,a_j) is a minimal-f 2-tuple with respect to (d_i,d_j). According to Lemma 3.1, every transformation executed in the algorithm is an effective pq-transformation. Therefore, we can apply a sequence of effective pq-transformations to S to transform S into S^* such that S^* is a minimal-f N-tuple with respect to R. Assume that Algorithm 2 takes m steps to finish. Let $S_0=S$ and after the execution of the mth step, the N-tuple becomes S_m. We now have $S_0,S_1,\ldots,S_m$, where $S_0=S$ and $S_m=S^*$. According to Lemma 3.1, $f_R(S_k)<f_R(S_{k-1})$. In particular, $f_R(S^*)=f_R(S_m)<f_R(S_0)=f_R(S)$. This completes the proof. □

Chang et al. [3] showed that there exist few exceptions such that minimal N-tuples are not unique. For instance, among all integers from 1 to 1000 for $N=3$, integer 360 is the only case with two minimal 3-tuples, namely (6,6,10) and (5,8,9). Besides, a minimal-f N-tuple $(a_1,a_2,\ldots,a_N)$ with respect to $(d_1,d_2,\ldots,d_N)$ is a minimal N-tuple except when there exists one pair a_i and a_j such that $a_ia_j=4$, or $\min_{1\leqslant i\leqslant N}d_i\leqslant 8$. Therefore, there exists only one minimal-f N-tuple in most cases, and the minimal-f N-tuple is the N-tuple such that $f_R(S)$ is the minimum. Furthermore, we obtain the following corollary.

COROLLARY 3.3. *Let S be an N-tuple of C and a factor of N-tuple R. If there is only one minimal-f N-tuple of C with respect to R, $f_R(S)$ is the smallest among all possible N-tuples of C with respect to R if and only if S is the only minimal-f N-tuple with respect to R.*

4. THE APPLICATION OF N-TUPLE THEORIES TO THE DESIGN OF CPFs

In Section 2, we found that the problem of designing an optimal CPF can be reduced to the problem of dividing each domain D_i into m_i subsets. The values of $m_1,m_2,\ldots,m_N$ should satisfy the following conditions:

1. $m_1m_2\ldots m_N=NB=$ the number of buckets,
2. d_i/m_i is an integer, and
3. $\Pi_{i=1}^{N}(-(d_i/m_i)^2+3d_i/m_i+3d_i^2/m_i+d_i^2+6)$ is minimal.

Applying Corollary 3.3, we obtain the following theorem.

THEOREM 4.1. *Let all records be N-attribute, d_i be the domain size of the ith attribute, and NB be the number of buckets. Then a CPF F is the optimal*

CPF for ORQs if its partition form $\langle m_1, m_2, \ldots, m_N \rangle$ *satisfies the following conditions*:

(1) $\Pi_{i=1}^{N} m_i = NB$, *and*

(2) $(m_1, m_2, \ldots, m_N)$ *is the only minimal-f N-tuple of NB with respect to* $(d_1, d_2, \ldots, d_N)$.

To obtain a set of m_i's that satisfy conditions (1) and (2), we simply apply Algorithm 2 to the N-tuple $(1, 1, \ldots, NB)$. If the resulting N-tuple $(m_1, m_2, \ldots, m_N)$ is the only minimal-f N-tuple with respect to $(d_1, d_2, \ldots, d_N)$, then an optimal CPF has been obtained.

EXAMPLE 4.1. Consider $|D_1| = 8$, $|D_2| = 4$, $|D_3| = 6$, and $NB = 32$. Applying Algorithm 2 to $(1, 1, 32)$, *we obtain* $(4, 2, 4)$ *as a minimal-f* 3*-tuple with respect to* $(|D_1|, |D_2|, |D_3|)$. *Since* $(4, 2, 4)$ *is the only minimal* 3*-tuple of* 32, *it is the only minimal-f* 3*-tuple of* 32. *Therefore,* D_1 *should be equally partitioned into four subsets,* D_2 *into two subsets, and* D_3 *into four subsets to produce an optimal CPF.*

THEOREM 4.2. *Let all records be N-attribute,* d_i *be the domain size of the i-th attribute, and NB be the number of buckets. Then a CPF F is the optimal CPF for ORQs if its partition form* $\langle m_1, m_2, \ldots, m_N \rangle$ *satisfies the following conditions*:

(1) $\Pi_{i=1}^{N} m_i = NB$,

(2) *each pair* (m_i, m_j) *satisfies* $m_i m_j \neq 4$,

(3) $(m_1, m_2, \ldots, m_N)$ *is the only minimal N-tuple of NB with respect to* $(d_1, d_2, \ldots, d_N)$,

(4) $(m_1, m_2, \ldots, m_N)$ *is in reverse order with respect to* $(d_1, d_2, \ldots, d_N)$, *and*

(5) *all* $d_i \geqslant 9$.

Proof. According to Corollary 3.2, each pair (m_i, m_j) is a minimal-f 2-tuple with respect to (d_i, d_j). This yields $(m_1, m_2, \ldots, m_N)$ as a minimal-f N-tuple of NB with respect to $(d_1, d_2, \ldots, d_N)$. Besides, $(m_1, m_2, \ldots, m_N)$ is the only minimal N-tuple of NB. Thus $(m_1, m_2, \ldots, m_N)$ is the only minimal-f N-tuple of NB with respect to $(d_1, d_2, \ldots, d_N)$. From Theorem 4.1, we conclude that $\langle m_1, m_2, \ldots, m_N \rangle$ is the optimal CPF partition form and F is the optimal CPF. □

5. CONCLUSIONS

In this paper, we have studied properties that help in designing a good CPF for ORQs. We have shown the problem of designing an optimal CPF for ORQs to be related to the problem of finding a minimal N-tuple, which corresponds to the *partition form* of a CPF, with few exceptions. Because of these exceptions, we defined a new term, *minimal-f N-tuple*, to cover minimal N-tuples that describe the properties of optimal CPFs for ORQs. We have also shown some theories of minimal-f N-tuples. By employing the theories of minimal-f N-tuples, we have developed a method for generating a minimal-f N-tuple and derived some properties of the optimal CPF for ORQs. Intuitively, these properties will provide a guideline to solve the problem of the optimal CPF for ORQs in a distributed system, and design a good search tree structure for ORQs in the near future.

APPENDIX A

THEOREM 3.1. *For a 2-tuple R, and its two factors $S=(a_1,a_2)$ and $T=(a'_1,a'_2)$, if $a_1a_2=a'_1a'_2\neq 4$ and $a_1+a_2<a'_1+a'_2$, then $f_R(S)<f_R(T)$.*

Proof. By $a_1a_2=a'_1a'_2$, we can assume $(a_1,a_2)=(mp,q)$ and $(a'_1,a'_2)=(m,pq)$. Since $a_1+a_2<a'_1+a'_2$, $mp+q<m+pq$ or $(p-1)(q-m)>0$. Hence $p>1$ and $q>m$. Let $R=(d_1,d_2)$. Because R is a common multiple of S and T, we set $d_1=mpx_1$ and $d_2=pqx_2$. Then

$$f_R(S)=\left(-x_1^2+3x_1+3mpx_1^2+m^2p^2x_1^2+6\right)$$
$$\times\left(-p^2x_2^2+3px_2+3p^2qx_2^2+p^2q^2x_2^2+6\right),$$
$$f_R(T)=\left(-p^2x_1^2+3px_1+3mp^2x_1^2+m^2p^2x_1^2+6\right)$$
$$\times\left(-x_2^2+3x_2+3pqx_2^2+p^2q^2x_2^2+6\right),$$

and

$$f_R(S)-f_R(T)=(1-p)(A+B+C),$$

where

$$A=p^2(q-m)x_1^2x_2^2(3mpq-(q+m)(p+1)-3),$$
$$B=3(x_2-x_1)(px_1x_2+2(p+1)(x_2+x_1)-6)-18p\left(qx_2^2-mx_1^2\right),\quad\text{and}$$
$$C=3p^2x_1x_2\left(q^2x_2-m^2x_1\right).$$

Since $1 < p$, $A + B + C > 0$ holds if and only if $f_R(S) < f_R(T)$. Besides, by $a_1 a_2 \neq 4$, we know the case of $m = 1$ and $p = q = 2$ does not exist. Thus, we need to show $A + B + C > 0$ holds except for $m = 1$ and $p = q = 2$. We distinguish this into three cases: (1) $m = 1$, $p \geqslant 2$, and $q \geqslant 3$, (2) $m = 1$, $p \geqslant 3$, and $q = 2$, and (3) $m \geqslant 2$, $p \geqslant 2$, and $q \geqslant m + 1$.

CASE 1. $\{m = 1,\ p \geqslant 2, \text{ and } q \geqslant 3\}$.
Here

$$A = p^2(q-1)x_1^2x_2^2(3pq - (q+1)(p+1) - 3),$$

$$B = 3(x_2 - x_1)(px_1x_2 + 2(p+1)(x_2 + x_1) - 6) - 18p(qx_2^2 - x_1^2), \quad \text{and}$$

$$C = 3p^2x_1x_2(q^2x_2 - x_1).$$

Since

$$A \geqslant 3(q-1)p^2x_1^2x_2^2, \quad \text{by } p \geqslant 2 \quad \text{and} \quad q \geqslant 3, \quad \text{and}$$

$$B > 3(x_2 - x_1)px_1x_2 - 18pqx_2^2, \quad \text{by } p \geqslant 2,$$

hence $A + B + C > 3(q-1)p^2x_1^2x_2^2 + 3(x_2 - x_1)px_1x_2 - 18pqx_2^2 + C = 3px_2D$, where $D = ((q-1)x_1x_2 + q^2x_2 - x_1)x_1 p + (x_2 - x_1)x_1 - 6qx_2$. Similarly, by substituting $p \geqslant 2$, $q \geqslant 3$, $x_1 \geqslant 1$, and $x_2 \geqslant 1$ into equation D to eliminate the corresponding variables, we obtain $D > 0$. This yields $A + B + C > 0$.

CASE 2. $\{m = 1,\ p \geqslant 3, \text{ and } q = 2\}$.
Here

$$A = 3p^2x_1^2x_2^2(p-2) \geqslant 3p^2x_1^2x_2^2, \quad \text{by } p \geqslant 3.$$

Define $a' \equiv 3p^2x_1^2x_2^2$. We further distinguish the following two cases.

(i) $x_1 > x_2$.
Let $x_1 = kx_2$, where $k > 1$. Then, $a' = 3p^2k^2x_2^4$ and $C = 3p^2kx_2^2(4x_2 - kx_2)$.

$$B = -3(k-1)x_2(pkx_2^2 + 2(p+1)(k+1)x_2 - 6) - 18p(2-k^2)x_2^2$$

$$> -3p(k-1)kx_2^3 + 9pk^2x_2^2 - 30px_2^2, \quad \text{by } p \geqslant 3 \quad \text{and} \quad k > 1.$$

Let $b' \equiv -3p(k-1)kx_2^3 + 9pk^2x_2^2 - 30px_2^2$. Hence $A+B+C > a' + b' + C = 3px_2^2D$, where $D = pk^2x_2^2 - (k-1)kx_2 + 3k^2 - 10 + 4pkx_2 - pk^2x_2$. Similarly, by substituting $p \geqslant 3$, $x_2 \geqslant 1$, and $k > 1$ into equation D to eliminate the corresponding variables, we obtain $D > 0$. That yields $A+B+C > 0$.

(ii) $x_1 \leqslant x_2$.

Let $x_2 = kx_1$, where $k \geqslant 1$. Then, $a' = 3p^2k^2x_1^4$ and $C = 3p^2kx_1^2(4kx_1 - x_1)$.

$$B = 3(k-1)x_1(pkx_1^2 + 2(p+1)(k+1)x_1 - 6) - 18p(2k^2-1)x_1^2$$

$$\geqslant -36pk^2x_1^2, \quad \text{by } p \geqslant 3 \quad \text{and} \quad k \geqslant 1.$$

Let $b' \equiv -36pk^2x_1^2$. Hence $A+B+C > a' + b' + C = 3pkx_1^2D$, where $D = 4pkx_1 - 12k + pkx_1^2 - px_1$. Similarly, by substituting $p \geqslant 3$, $x_1 \geqslant 1$, and $k \geqslant 1$ into equation D to eliminate the corresponding variables, we obtain $D \geqslant 0$. That yields $A+B+C > 0$.

CASE 3. $\{m \geqslant 2,\ p \geqslant 2, \text{ and } q \geqslant m+1\}$.

Here

$$A = p^2(q-m)x_1^2x_2^2(3mpq - (q+m)(p+1) - 3),$$

$$B = 3(x_2 - x_1)(px_1x_2 + 2(p+1)(x_2+x_1) - 6) - 18p(qx_2^2 - mx_1^2), \quad \text{and}$$

$$C = 3p^2x_1x_2(q^2x_2 - m^2x_1).$$

Since

$$A \geqslant 3(m-1)p^3q(q-m)x_1^2x_2^2, \quad \text{by } p \geqslant 2 \quad \text{and} \quad q \geqslant m+1, \quad \text{and}$$

$$B > 3(x_2 - x_1)px_1x_2 - 18pqx_2^2, \quad \text{by } p \geqslant 2 \quad \text{and} \quad x_2 \geqslant 1,$$

hence $A+B+C > 3(m-1)p^3q(q-m)x_1^2x_2^2 + 3(x_2-x_1)px_1x_2 - 18pqx_2^2 + C = 3px_2D$, where $D = ((m-1)p^2x_1^2x_2 + px_1x_2)q^2 - (m(m-1)p^2x_1^2x_2 + 6x_2)q - m^2px_1^2 + (x_2-x_1)x_1$. Similarly, by substituting $q \geqslant m+1$, $p \geqslant 2$, $m \geqslant 2$, $x_1 \geqslant 1$, *and* $x_2 \geqslant 1$ *into equation D to eliminate the corresponding variables, we obtain* $D > 0$. *That yields* $A+B+C > 0$.

These cases complete the proof. □

This research was supported by the National Science Council, Taiwan, R.O.C., under contract NSC 83-0408-E009-15.

REFERENCES

1. K. A. S. Abdel-Ghaffar and A. El Abbadi, Optimal disk allocation for partial match queries, *ACM Trans. Database Syst.* 18(1):132–156 (1993).
2. A. V. Aho and J. D. Ullman, Optimal partial match retrieval when fields are independently specified. *ACM Trans. Database Syst.* 4(2):168–179 (1979).
3. C. C. Chang, R. C. T. Lee, and H. C. Du, Some properties of Cartesian product files, *Proc. ACM SIGMOD—Int. Conf. Management of Data*, Santa Monica, 1980, pp. 157–168.
4. C. C. Chang, M. W. Du, and R. C. T. Lee, Performance analyses of Cartesian product files and random files, *IEEE Trans. Software Eng.* SE-10:88–99 (1984).
5. C. C. Chang, Optimal information retrieval when queries are not random, *Inform. Sci.* 34(3):199–223 (1984).
6. C. C. Chang, C. Y. Chen, and S. S. Chang, Optimality properties of binary Cartesian product file systems, *Policy Inf.* 13(1):115–125 (1989).
7. C. Y. Chen, C. C. Chang, and R. C. T. Lee, Optimal MMI file systems for orthogonal range retrieval, *Inf. Syst.* 18(1):37–54 (1993).
8. A. Y. H. Chou, The design of a good multi-attribute file system for queries, M. S. thesis, National Chiao Tung University, Taiwan, ROC, 1988.
9. A. Y. H. Chou, W. P. Yang, and C. C. Chang, Greedy file—A new data organization concept for partial matcha retrieval, *Comput. J.* 35:403–408 (1992).
10. W. C. Lin, R. C. T. Lee, and H. C. Du, Common properties of some multi-attribute file systems, *IEEE Trans. Software Eng.* SE-5:160–174 (1979).
11. J. H. Liou and S. B. Yao, Multi-dimensional clustering for data base organizations, *Inf. Syst.* 2:187–198 (1977).
12. R. L. Rivest, Partial-match retrieval algorithms, *SIAM J. Comput.* 14(1):19–50 (1976).
13. J. B. Rothnie and T. Lozano, Attribute based file organization in paged memory environment, *Commun. ACM* 17(2):63–69 (1974).
14. T. Y. Tang, D. J. Buehrer, and R. C. T. Lee, On the complexity of some multi-attribute file design problem, *Inf. Syst.* 10(1):21–25 (1985).

Received 1 January 1995; revised 30 August 1995