AD-A042 436 OHIO STATE UNIV RESEARCH FOUNDATION COLUMBUS F/G 9/4
LOCALLY STRUCTURED INFORMATION SYSTEMS,(U)
DEC 73 D K RAY-CHAUDHURI, J FRIED N00014-67-A-0232-0016
UNCLASSIFIED OSURF-3430-A1 NL

| OF |
AD
A042436

END
DATE
FILMED
8-77
DDC

(1)

Locally Structured Information Systems

D. K. Ray-Chaudhuri*
The Ohio State University
and
John Fried
Battelle Memorial Institute

(11) Dec 1973 (12) 28 p.

1. Information storage and retrieval. The problem of information storage and retrieval is a companion of any civilization. Any civilized society likes to preserve its knowledge and information for its future generations. In fact to ensure that such information can be used by the posterity, it has to be classified and stored properly. Today's knowledge will be available tomorrow only if we can store it in a manner that will permit ready retrieval. As the pool of human knowledge grows, retrieval of pertinent information becomes more and more difficult. If information is not stored efficiently, it may turn out to be easier to rediscover information on a certain item than to retrieve it. However this need not be the case. Mathematical techniques can be used to devise efficient information storage and retrieval systems that allow for quick retrieval of information pertinent to a given query. The problem of storing information in a computer has many combinatorial aspects. In this paper we try to develop some efficient information storage and retrieval systems by using methods of Combinatorial Mathematics. Combinatorial configurations have been used for constructing filing systems.

* This research was supported in part by O.N.R. Contract No. (15) N00014-67-A-0232-0016 (14) OSURF-0232-0016. (O.S.U.R.F. Project No. 3430-A1)

OSURF-3430

267360

by Abraham, Ghose and Ray-Chaudhuri [1], Ray-Chaudhuri [3] and Bose [2].

2. Description of a file. Any information system (also called a filing system) is concerned with a large collection of units (also called items, individuals, documents) and information about these units on a number of variables. These variables might be called the information variables. The totality of the information variables, the collection of units and information about them constitutes a file. To give an example of a file, there may be a file for all the pilots of the Air Force of a country. The units in such a file are the pilots and some of the information variables may be (1) the age of the pilot, (2) the number of combat missions flown by the pilot, (3) whether or not the pilot is a veteran of a past war and (4) whether or not the pilot is married. In a file for the employees of a company, units are the employees. Some of the information variables might be (a) whether or not the employee worked for the research division of the company, (b) whether or not the employee worked for more than 5 years, (c) whether or not the employee published more than 5 scientific papers and (d) whether or not the employee is married. In a file for research publications on copper units are the research papers. Some of the information variables could be (a) whether or not the paper is relevant for aluminium alloys, (b) whether or not the paper is pertinent to cryogenics, (c) whether or not the paper is written in English and (d) whether or not the paper mentions steel, etc. In a file for publications on nuclear engineering, some of the variables could be (1) proton accelerators (2) bevatrons (3) cyclotrons (4) linear accelerator and so on. In many situations, the information

variables are binary and they are variously called as attributes, terms, descriptors, clue words, or locators. If there is a nonbinary information variable, it is possible to replace it by artificially created binary valued information variables. One of the most important retrieval problems is the following: We are given a subset of the set of all attributes and we like to know the set of individuals who satisfy the given subset of attributes. For instance in our example of the file for the employees of a company, we might like to find out the subcollection of employees who are married and published more than 5 scientific papers. In the file for research publications on copper, we might like to find out all research publications on copper which deal with aluminium alloys as well as cryogenic methods. In a file usually each individual is given an identification which may be a name or a serial number. The identification of the individual and the values of the information variables for the individual constitutes his record.

Large files nowadays are stored in computers. In most computerized filing systems the records are stored in some comparatively slow permanent memory, for instance, in a tape. The address of the permanent memory for a record is called the accession number of the record. Obviously, the accession number is usually much smaller in size than the complete record. A set of addresses of the comparatively faster memory is reserved for storing the accession numbers.

3. Formal definitions of a file, storage rule and retrieval rule and inverted filing system. For any set X, $P(X)$ will denote the set

of subsets of X and $|X|$ will denote the cardinality of X. A file F is a triple (I, A, f) where I is the set of individuals (also called documents or items), A is a set of attributes (also called descriptors or locators) and f is a mapping from I to $P(A)$. For $i \in I$, $f(i)$ is the set of attributes possessed by the individual i. If A is a set of v attributes, $f(i)$ can be represented as a binary v-tuple. A computerized file is a pair (F, a) where F is a file and a is an injective mapping from I to integers. For $i \in I$, $a(i)$ is called the accession number of i. The accession number is usually the address of the permanent computer memory location where the identification of the individual i and the corresponding record $f(i)$ is stored. For the sake of brevity, let $I^* = \{a(i): i \in I\}$. I^* is the set of accession numbers of the items of the file. A storage rule for the file F is a tuple (I, M, s) where I is the set of individuals of the file F, M is a set of fast computer memory locations and s is a mapping from I to $P(M)$ such that for $i, j \in I$, $i \neq j$, $s(i)$ and $s(j)$ have no element in common. The accession number of the individual i is stored at each of the memory locations belonging to $s(i)$. If for each $i \in I$, $s(i)$ contains only one element, we say that the storage rule admits no redundancy. For $A' \subseteq A$, let $I(A') = \{i: i \in I, f(i) \supseteq A'\}$. A retrieval rule is a triple $(\mathcal{A}, I^*, r)$ where $\mathcal{A}$ is a subset of $P(A)$ and r is a mapping from $\mathcal{A}$ to $P(I^*)$ such that

$$\forall A' \in \mathcal{A},\ r(A') = \{a(i): i \in I(A')\} \qquad \ldots\ldots (1)$$

The file F, together with the accession numbers, storage rule and

retrieval rule constitutes an information system, (also called a filing system). The elements of $\mathcal{A}$ are called queries. The equation (1) requires that for any query A' and $i \in I(A')$, $r(A')$ contains the accession number of each item i which satisfies all the attributes of A'. The most prevalent computerized filing system is the inverted filing system (also called coordinate indexing). Storage rules and retrieval rules of inverted filing system can be described as follows. Let $A = \{a_1, a_2, \ldots, a_v\}$ be a set of v attributes. The set of storage addresses M is partitioned into v disjoint subsets M_{a_1}, M_{a_2} ..., M_{a_v} called buckets. An individual's accession number will be stored in the bucket M_{a_j} if and only if the individual satisfies the attribute a_j, $1 \leq j \leq v$. In other words s has the property that $s(i) \cap M_{a_j}$ contains one element if and only if $a_j \in f(i)$. For $a \in A$, define $M_a^* = \{a(i): s(i) \cap M_a \neq \phi, i \in I\}$. In other words M_a^* is the set of accession numbers stored in the memory locations M_a. The retrieval rule is also easily described for a query A'. We take $r(A') = \bigcap_{a \in A'} M_a^*$. Clearly for any item i which possesses all the attributes of A', $r(A')$ will contain the accession number of i. If the query consists of a single attribute a, then $r(\{a\}) = M_a^*$. For queries involving only one attribute, retrieval can be done very efficiently in the inverted filing system. However if the query involves more than one attribute, then it is necessary to intersect several of the buckets M_a^* and retrieval can be very slow for a large file. To overcome this difficulty of inverted filing system, we introduce the principle of local structuring.

4. Locally structured information systems. In most large files most of the items satisfy only a small fraction of the descriptors. This important fact is exploited by the principle of local structuring. Local structuring partitions a large file into several small files such that to retrieve the items for a given query, we need to search only a few of these small files. Let $F = (I, A, f)$ be a file. Let $\mathcal{A} \subseteq P(A)$ be the set of queries of interest. Let $\Pi = (I_1, I_2, \ldots, I_b)$ be a b-tuple of subsets of I such that $I_j \cap I_k = \phi$ (the empty set), for all $j \neq k$, $j, k = 1, \ldots b$ and $\bigcup_{j=1}^{b} I_j = I$. In other words Π is an ordered partition of b. Let $\delta = (A_1, A_2, \ldots, A_b)$ be a b-tuple of subsets of A. For $I' \subseteq I$, $A' \subseteq A$, $I'(A')$ denotes the set of items i which satisfy $f(i) \supseteq A'$. Let $J = \{1, 2, \ldots, b\}$ and for $A' \subseteq A$, $J_{A'} = \{j: j \in J_b, A' \subseteq A_j\}$. The pair (Π, δ) is said to be a local structuring for $(F, \mathcal{A})$ if and only if

$$\forall A' \in \mathcal{A},\ I(A') = \bigcup_{j \in J_{A'}} I_j(A')$$
$$\text{and } \forall i \in I_j,\ f(i) \subseteq A_j,\ 1 \leq j \leq b. \qquad \ldots\ldots (2)$$

The local structuring ℓ is said to have the equicardinality property if and only if $|I_1| = |I_2| = \ldots = |I_b|$. Furthermore, ℓ is said to be uniform if and only if $|A_1| = |A_2| = \ldots = |A_b|$. Let f_j be the restriction of f to I_j, i.e. $f_j(i) = f(i)$, for all $i \in I_j$. The files $F_j = (I_j, A_j, f_j)$, $1 \leq j \leq b$ are called the local files. Let M_j, $1 \leq j \leq b$ be b disjoint sets of computer memory locations, s_j be a storage rule for the local file F_j and r_j be a retrieval rule for the local file F_j, $1 \leq j \leq b$. We define storage rule s and retrieval

rule r for F by

$$s(i) = s_j(r) \quad \text{if} \quad i \in I_j \,, \; 1 \leq j \leq b$$

$$r(A) = \bigcup_{j \in J_{A'}} r_j(A') \quad , \quad \forall\, A' \in \mathcal{A} \qquad \ldots\ldots (3)$$

Since $I(A') = \bigcup_{j \in J_A} I_j(A')$, r is easily seen to be a retrieval rule for the file F with respect to the set of queries $\mathcal{A}$.

Let $\pi_o = (I)$ and $\delta_o = (A)$ with $b = 1$. Clearly the pair $\ell_o = (\pi_o, \delta_o)$ satisfies the requirements of local structuring and will be called the trivial local structuring. The principle of local structuring opens up a new dimension for methods of file organization.

The central problem to a builder of a filing system will be that of determining the optimum local structuring for a given file. The optimality criterion itself will vary from situation to situation. In most large files, the majority of the item satisfies only a small fraction of the descriptors of the file. Suppose v is the total number of descriptors. Without much loss in generality one can postulate the existence of a positive integer t much smaller than v such that no item satisfies more than t descriptors. A few items which satisfy more than t attributes can be grouped together into a small file. In the next section we show that under these assumptions one can use combinatorial configurations to construct a local structuring ℓ which is better than the trivial local structuring. In this paper we only want to emphasize the principle of local structuring. We want to point out this alternative to the planners and builders of information systems. The problem of determining the

optimum local structuring for a given file is a complicated and difficult problem. Local structuring will give maximum advantage if all the local files are relatively small and the number of local files is not too large. Finally for every query A', the cardinality of the set $J_{A'}$ should be small. One could safely make the claim that in almost all practical situations there will exist a local structuring which is better than the trivial local structuring.

5. Local structuring based on combinatorial configurations. Let v, k, t and b be positive integers such that $t \leq k \leq v$. Let A be a set of v elements and $\mathcal{A}$ be a class of subsets of A. Members of $\mathcal{A}$ are called queries. A combinatorial configuration with parameters $(A, k, \mathcal{A}, b)$ is a pair $(A, \mathcal{B})$ with $\mathcal{B} = \{A_1, A_2, \ldots, A_b\} \subseteq P(A)$ such that

(1) $|A_j| \leq k$, $1 \leq j \leq b$

and (2) $\forall A' \in \mathcal{A}$, there exists at least one block A_j such that $A_j \supseteq A'$.

The subsets $A_1, A_2, \ldots, A_b$ are called blocks. If $\mathcal{A}$ is the class of subsets of cardinality not greater than t, then the configuration is called a (v, k, t, b) - configuration. Let $b(v, k, t)$ denote the smallest integer b such that a (v, k, t, b) - configuration exists. A (v, k, t, b) - configuration is said to be optimum if and only if $b = b(v, k, t)$. We can use (v, k, t, b) - configurations to construct a locally structured filing system for a file $F = (I, A, f)$ provided no item satisfies more than t descriptors. Let $A_1, A_2, \ldots, A_b$ be the blocks of the (v, k, t, b) - configuration. Let $\delta = (A_1, A_2, \ldots, A_b)$.

Let $I_j' = \{i: f(i) \subseteq A_j\}$, $1 \leq j \leq b$. Let π be an ordered partition such that

$$I_j \subseteq I_j' \text{ , for all } j \text{ , } 1 \leq j \leq b \qquad \ldots\ldots (4).$$

Since $\forall\, i \in I$, $|f(i)| \leq t$, there will exist at least one integer j such that $1 \leq j \leq b$ and $f(i) \subseteq A_j$. Hence partitions of I satisfying (4) will exist. In particular we can take

$$I_1 = I_1'$$
$$I_j' = I_j' - I_1 \cup I_2 \ldots \cup I_{j-1} \text{ , } 2 \leq j \leq b \qquad \ldots\ldots (5)$$

However, it will be more desirable to have a partition π which satisfies (4) and also satisfies the equicardinality property

$$I_1 = I_2 = \ldots = I_b \qquad \ldots\ldots (6)$$

We now show that if π is an ordered partition satisfying (4), then the pair (π, δ) is a local structuring for the pair $(F, P(A))$. Let $A' \subseteq A$. Clearly $I(A') \supseteq \bigcup_{j \in J_{A'}} I_j(A')$. Conversely, let $i \in I(A')$. Let j_o be such that $i \in I_{j_o}$. Then $i \in I_{j_o}(A')$. Also $A' \subseteq f(i) \subseteq A_{j_o}$. Therefore $j_o \in J_{A'}$ and $i \in \bigcup_{j \in J_{A'}} I_j(A')$. Hence $I(A') \subseteq \bigcup_{j \in J_{A'}} I_j(A')$ and the defining property (2) holds for (π, δ) .

Let F_j denote the restriction of f to I_j , $1 \leq j \leq b$. Let $F_j = (I_j, A_j, F_j)$, $1 \leq j \leq b$ be the local files. Let M_j be a set of computer memory locations reserved for the file F_j , $1 \leq j \leq b$. Let s_j and r_j respectively denote storage rules and retrieval rules for the file F_j , $1 \leq j \leq b$. Then the storage rule s and the retrieval rule r

for the file F is defined by

$$s(i) = s_j(i) \quad \text{if} \quad i \in I_j \,, \; 1 \leq j \leq b \,.$$

$$r(A') = \bigcup_{j \in J_{A'}} r_j(A') \,, \; \forall \, A' \in P(A)$$

We note that if the local storage rules s_j do not admit any redundancy, the storage rule s also has no redundancy. The following is a schematic representation of the local structuring constructed above.

Blocks	Sets of items of the local files	Computer memory locations for the local files	Storage rule of the local file	Retrieval rule of the local files
A_1	I_1	M_1	s_1	r_1
A_2	I_2	M_2	s_2	r_2
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
A_b	I_b	M_b	s_b	r_b

Note that any particular item is stored in exactly one local file. For an item i, we need to determine an integer j_o for which $A_{j_o} \supseteq f(i)$. Then the item i will be stored in the local file F_{j_o} according to the rule s_{j_o}. To retrieve for a query A', the computer program will first scan the list of blocks and determine the set $J_{A'}$ of integers j for which the block A_j contains A'. Once the set $J_{A'}$ is determined, the files F_j with $j \notin J_{A'}$ can be ignored for the purpose of retrieving for the query A'. The program will find out the items satisfying A' in the file F_j for each $j \in J_{A'}$ and take the union of these sets of items.

6. Retrieval efficiency of locally structured information systems. In this section we develop some expressions for retrieval times required for various queries in a locally structured information system. These expressions are based on some simplifying assumptions which can only be an approximation to the very complex situation for a real file. The discussion in this section should be viewed only as an approach for computing the retrieval efficiency of a locally structured information system. However it is hoped that the methods developed in this section will provide a guide for comparing the retrieval efficiencies of various locally structured information systems. Let N denote the set of natural numbers. Let $F = (I, A, f)$ be a file with $\mathcal{A} \subseteq P(A)$ as the set of queries of interest. Let $\Pi = (I_1, I_2, \ldots, I_b)$ and $\delta = (A_1, A_2, \ldots, A_b)$ be b-tuples of subsets of I and A respectively such that $\ell = (\Pi, \delta)$ is a local structuring for $(F, \mathcal{A})$. Let $T_\ell(A')$ denote the amount of time (i.e. the number of units of computational time) required to retrieve all items for the query A'. $T_\ell(A')$ consists of two components $T_1(A')$ and $T_2(A')$. $T_1(A')$ is the amount of time required to determine the set $J_{A'}$. $T_2(A')$ is the amount of time required to search within the local files F_j for $j \in J_{A'}$. The amount of computing time required to check whether the relation $A' \subseteq A_j$ holds or not will depend on the cardinalities $|A'|$ and $|A_j|$. We assume that there exists a function $\alpha: N \times N \to N$ such that $\alpha(|A'|, |A_j|)$ is the amount of time required to check the relation $A' \subseteq A_j$ for all $j, 1 \leq j \leq b$. Under these assumptions

$$T_1(A') = \sum_{j=1}^{b} \alpha\left(|A'|, |A_j|\right) \qquad \ldots\ldots (7).$$

We assume that the sum $\sum_{j=1}^{b} \alpha(|A'|, |A_j|)$ is a function of the cardinalities $|A'|$ and $\sum_{j=1}^{b} |A_j|$. Let $\lambda : N \times N \to N$ be a function such that

$$T_1(A') = \lambda\left(|A'|, \sum_{j=1} |A_j|\right) \qquad \ldots\ldots (8).$$

For the trivial local structuring ℓ_o, the component $T_1(A')$ is equal to o. The function λ is assumed to have the property that $\lambda(r, v) = o$, for all positive integers r. Clearly the time required to retrieve for a query A' in a file $F = (I, A, f)$ will depend on $|A'|$, $|A|$ and the number of items in the file. Let $\phi : N \times N \times N \to N$ be a function such that $\phi(|A'|,|I|,|A|)$ is the amount of time required to retrieve for a query A'. Under these assumptions we get

$$T_2(A') = \sum_{j \in J_{A'}} \phi\left(|A'|, |I_j|, |A|\right), \forall\, A' \in \mathcal{A} \qquad \ldots\ldots (9).$$

Let $n = |I|$, $n_j = |I_j|$, $k_j = |A_j|$, $\bar{n} = \frac{1}{b} \sum_{j=1}^{b} n_j$ and $\bar{k} = \frac{1}{b} \sum_{j=1}^{b} k_j$, $1 \le j \le b$. Let $r \le t$ be a positive integer and $P_r(A)$ denote the set of subsets of A with cardinality equal to r. For an integer r, $T_\ell(r)$ will denote the average retrieval time for a query A' with $|A'| = r$ in the local structuring ℓ. Combining the expressions (8) and (9) we get

$$T_\ell(A') = \lambda\left(|A'|, b\bar{k}\right) + \sum_{j \in J_{A'}} \phi\left(|A'|, n_j, k_j\right)$$

$$T_\ell(r) = \frac{1}{\binom{v}{r}} \sum_{A' \in P_r(A)} T_\ell(A') = \lambda(r, b\bar{k}) + \sum_{j=1}^{b} \sum_{\substack{A' \in P_r(A) \\ A' \subseteq A_j}} \phi(r, n_j, k_j)$$

$$= \lambda(r, b\bar{k}) + \frac{1}{\binom{v}{r}} \sum_{j=1}^{b} \phi(r, n_j, k_j) \binom{k_j}{r} \qquad \ldots\ldots (10)$$

If the local structuring ℓ satisfies the equicardinality property and the uniformity property i.e. if $n_1 = n_2 = \ldots = n_b = \bar{n}$ and $k_1 = k_2 = \ldots = k_b$, then we get

$$T_\ell(r) = \lambda(r,bk) + \phi(r,\bar{n},k)\,\frac{b\binom{k}{r}}{\binom{v}{r}} \qquad \ldots\ldots(11).$$

If the (v,k,t,b) - configuration is uniform and the local structuring ℓ does not necessarily satisfy the equicardinality property, then the formula (10) reduces to

$$T_\ell(r) = \lambda(r,bk) + \frac{\binom{k}{r}}{\binom{v}{r}} \sum_{j=1}^{b} \phi(r,n_j,k) \qquad \ldots\ldots(12).$$

It can be expected that $\sum_{j=1}^{b} \phi(r,n_j,k)$ will take the smallest possible value when all the n_j's are equal, $1 \le j \le b$. Therefore it will be desirable to make sure that the local structuring ℓ satisfies the equicardinality property as closely as possible. The formula (11) suggests that for fixed v, k and t, the retrieval time $T_\ell(r)$ takes the smallest possible value when the number of blocks b is smallest possible. Therefore for fixed v, k and t, one should look for a (v,k,t,b) - configuration which minimizes the number of blocks. For the trivial local structuring ℓ_o, (11) reduces to

$$T_{\ell_o}(r) = \phi(r,n,v) \qquad \ldots\ldots(13).$$

Let $\omega(A')$ be a nonnegative real number satisfying

$$0 \le \omega(A') \le 1$$

and

$$\sum_{A' \in \mathcal{A}} \omega(A') = 1$$

where $\mathcal{A}$ is the set of queries of interest. The number $\omega(A')$ is the weight attached to the query A' and measures the relative importance

of the query A' . The retrieval parameter $T(\ell)$ for the local structuring ℓ (with respect to the weight system $\omega(A')$, $A' \in \mathcal{A}$) can be defined by

$$T(\ell) = \sum_{A' \in \mathcal{A}} T_{\ell}(A')\ \omega(A')$$

7. Locally structured inverted filing system. Let $F = (I, A, f)$ be a file, $\Pi = (I_1, I_2, \ldots, I_b)$ be an ordered partition of I, $\delta = (A_1, A_2, \ldots, A_b)$ be a b-tuple of subsets of A, $\ell = (\Pi, \delta)$ be a local structuring and $F_j = (I_j, A_j, f_j)$, $1 \leq j \leq b$ be the local files of F. Let s_j and r_j be respectively storage rules and retrieval rules of an inverted filing system for F_j. Then we define

$$s(i) = s_j(i), \text{ if } i \in I_j$$

and

$$r(A') = \bigcup_{j=1}^{b} r_j(A'), \forall\ A' \in \mathcal{A}$$

where $\mathcal{A}$ is the set of queries of interest. The locally structuring ℓ together with the storage rule s and retrieval rule r will be called a locally structured inverted filing system. We now derive expressions for the retrieval parameter $T_{\ell}(r)$ for the locally structured inverted filing system. First we consider the trivial local structuring ℓ_o. For a query $A' = \{a_1, \ldots, a_r\}$ the retrieval time $T_2(A')$ can be split up into two components $T_3(A')$ and $T_4(A')$. $T_3(A')$ is the amount of time required to identify the buckets corresponding to the attributes belonging to A'. Let d be the amount of time required by the computer to check the equality of two attributes. We assume that there exists a function $\varphi : N \times N \to N$ such that $T_3(A') = d\varphi(|A'|, |A|)$.

$T_4(A')$ is the amount of time required to form the intersection $\bigcap_{a \in A'} M_a^*$ where M_a is the bucket corresponding to the attribute a. Let c be the amount of time required to test the equality of two accession numbers. We postulate that there exists a function $\mu : N \times N \times N \to N$ such that

$$T_4(A') = c\,\theta(r,n,v) \qquad .....(14).$$

Under these assumptions, we get

$$T_{\ell_o}(r) = d\varphi(r,v) + c\,\theta(r,n,v) \qquad .....(15).$$

Similarly for a locally structured inverted filing system ℓ with the equicardinality property and the uniformity property we get

$$T_\ell(r) = \lambda(r,bk) + \frac{b\binom{k}{r}}{\binom{v}{r}} \left(d\varphi(r,k) + c\,\theta(r,\frac{n}{b},k)\right) \qquad .....(16).$$

If we further make the simplifying assumption that $\lambda(r,bk) = d\varphi(r,b)$, we get

$$T_\ell(r) = d\left(\varphi(r,b) + \frac{b\binom{k}{r}}{\binom{v}{r}}\varphi(r,k)\right) + c\,\frac{b\binom{k}{r}}{\binom{v}{r}}\theta(r,\frac{n}{b},k) \qquad .....(17).$$

8. Lexicographic storage and retrieval rules in an inverted filing system. Consider an inverted filing system for a file $F = (I, A, f)$ with $A = \{a_1, a_2, \ldots, a_v\}$. Let M_i be the bucket corresponding to the attribute a_i, $1 \le i \le v$. Let $a(m)$ denote the accession number stored in the memory location m. Let $M' = \{m_1, m_2, \ldots, m_b\}$ be a bucket where $m_1, m_2, \ldots, m_b$ is a natural ordering of the memory locations in M'. The lexicographic storage rule requires that $a(m_1) < a(m_2) < \ldots < a(m_b)$. Consider two buckets $M_1 = \{m_{11}, m_{12}, \ldots, m_{1p}\}$ and $M_2 = \{m_{21}, m_{22}, \ldots, m_{2q}\}$ where $a(m_{11}) < a(m_{12}) < \ldots < a(m_{1p})$,

and $a(m_{21}) < a(m_{22}) < \ldots < a(m_{2q})$. To form the intersection $M_1^* \cap M_2^*$, the program starts with $a(m_{11})$, compares it with $a(m_{21})$, $a(m_{22})$ etc. and finds the smallest integer j such that $1 \leq j \leq q$ and $a(m_{11}) = a(m_{2j})$. If no such integer exists, then clearly $M_1^* \cap M_2^* = \phi$. Suppose such an integer j exists. Let $N_1^* = M_1^* - \{a(m_{11})\}$ and $N_2^* = M_2^* - \{a(m_{21}), \ldots, a(m_{2j})\}$. Then it is easily seen that

$$M_1^* \cap M_2^* = \{a(m_{11})\} \cup (N_1^* \cap N_2^*)$$

We now see that the number of comparisons of accession numbers to compute $M_1^* \cap M_2^*$ will be $\text{Min}(|M_1|, |M_2|)$. If $|M_1| = |M_2|$, then the number is equal to the common cardinality. Similarly to compute the set $M_1^* \cap M_2^* \cap M_3^*$, the required number of comparisons of accession numbers will be $\text{Min}(|M_1^*|, |M_2^*|) + \text{Min}(|M_1^* \cap M_2^*|, |M_3^*|)$. If we assume that all attributes are equally likely i.e.

$$\left| \bigcap_{a \in A'} M_a^* \right| = \frac{n}{\binom{v}{r}} \text{ , } \forall\, A' \in P_r(A) \text{ ,}$$

$$n = |I| \text{ ,}$$

then the number of comparisons of accession numbers required to compute $\bigcap_{a \in A'} M_a^*$ will be $\frac{n}{\binom{v}{1}} + \frac{n}{\binom{v}{2}} + \ldots + \frac{n}{\binom{v}{r-1}}$.

Therefore, for $A' \in P_r(A)$, $r > 1$, $T_4(A')$, the time required to compute the intersection $\bigcap_{a \in A'} M_a^*$ will be given by $\sum_{i=1}^{r-1} \frac{cn}{\binom{v}{i}}$ where c is the amount of time required to compare two accession numbers. In other words as a first approximation we may assume that the function $\theta : N \times N \times N \to N$

is given by

$$\theta(x,y,z) = \sum_{i=1}^{x-1} \frac{y}{\binom{z}{i}} \text{ , if } x > 1 \qquad \text{.....(18)}$$

and $\theta(x,y,z) = 0$, if $x = 1$

Next we analyze the amount of time required to check if a relation $A' \subseteq A''$ holds where $A' , A'' \subseteq A$. Suppose the attributes are assigned numbers. The attributes belonging to a subset A' of A will be stored in ascending order. If the rules of lexicographic search are applied, then the number of comparisons of attribute numbers required to test $A' \subseteq A''$ will be $\min(|A'|, |A''|)$. Therefore the time required to test $A' \subseteq A''$ will be $d \min(|A'|, |A''|)$ where d is the amount of time required to compare two attribute numbers. In other words the function $\varphi : N \times N \to N$ introduced in the last section can be assumed to be

$$\varphi(x,y) = xy \qquad \text{.....(19).}$$

Under these simplifying assumptions for $r > 1$ the formulas (15) and (17) reduce to

$$T_{\ell_o}(r) = drv + \sum_{i=1}^{r-1} \frac{cn}{\binom{v}{i}}$$

$$T_{\ell}(r) = d\left(rb + rkb \frac{\binom{k}{r}}{\binom{v}{r}}\right) + \sum_{i=1}^{r-1} \frac{c\binom{k}{r}}{\binom{v}{r}} \frac{n}{\binom{k}{i}} \qquad \text{.....(20)}$$

9. Boolean local structuring. Let $v \geq k \geq t > 0$ be integers. Let A be a set of v attributes and $P_k(A) = \{A_1 , A_2 , \ldots , A_b\}$, $b = \binom{v}{k}$, be the set of k-element subsets of A . Clearly the blocks $A_1 , A_2 , \ldots , A_b$ define a (v,k,t,b)-configuration. A local structuring ℓ based on this

trivial (v,k,t,b) - configuration will be called a Boolean (v,k,t) - local structuring. We assume that the local structuring ℓ has the equicardinality property and that an inverted filing system with lexicographic storage and retrieval rule is used for each local file. The retrieval parameters $T_\ell(r)$ and $T_{\ell_o}(r)$ are given by

$$T_{\ell_o}(1) = dv$$

$$T_\ell(1) = d\left(\binom{v}{k} + k\binom{v-1}{k-1}\right)$$

$$T_{\ell_o}(r) = drv + \sum_{i=1}^{r-1} \frac{cn}{\binom{v}{i}}, \quad r > 1$$

$$T_\ell(r) = d\left(r\binom{v}{k} + rk\binom{v-r}{k-r}\right) + \sum_{i=1}^{r-1} \frac{cn\binom{k}{r}}{\binom{v}{r}\binom{k}{i}}, \quad r > 1$$

Clearly $T_{\ell_o}(1) \leq T_\ell(1)$ and the trivial local structuring is better than any other local structuring for retrieval of queries of size 1.

If $r > 1$ and $1 \leq i \leq r-1$, $\dfrac{\binom{k}{r}}{\binom{v}{r}\binom{k}{i}} = \dfrac{1}{\binom{v}{i}} \dfrac{\binom{k-i}{r-i}}{\binom{v-i}{r-i}} < \dfrac{1}{\binom{v}{i}}$.

Therefore $\displaystyle\sum_{i=1}^{r-1} \frac{\binom{k}{r}}{\binom{v}{r}\binom{k}{i}} < \sum_{i=1}^{r-1} \frac{1}{\binom{v}{i}}$ and for fixed v, k, r, c and d and sufficiently large n (i.e. the number of items in the file) $T_\ell(r) < T_{\ell_o}(r)$. In fact as the size of a file grows, the size of the accession numbers grow and the constant c will tend to become larger. As an illustration consider the case $k = t = v - 1$. Then we get the

expressions

$$T_{\ell_o}(1) = dv$$

$$T_{\ell}(1) = d(v + (v-1)^2)$$

$$T_{\ell_o}(r) = drv + \sum_{i=1}^{r-1} \frac{cn}{\binom{v}{i}} \quad , \quad r > 1$$

$$T_{\ell}(r) = d(rv + r(v-1)^2) + \sum_{i=1}^{r-1} \frac{cn}{\binom{v}{i}} \frac{(v-r)}{(v-i)} \, , \; r > 1 \, .$$

$$T_{\ell_o}(2) = drv + \frac{cn}{v}$$

$$T_{\ell}(2) = d(2v + 2(v-1)^2) + \frac{cn(v-2)}{v(v-1)}$$

If $n \geq \frac{2dv(v-1)^3}{c}$, then $T_{\ell_o}(2) \geq T_{\ell}(2)$.

10. Projective local structuring. Let m be a positive integer and q be a prime power. Let $GF(q)$ denote the Galois field of order q. Let $PG(m-1, q)$ be the $(m-1)$-dimensional projective space over $GF(q)$. The number of $(l-1)$-flats of $PG(m-1,q)$ is given by the function $\phi(m,l,q)$ where $\phi(m,l,q) = \frac{(q^m-1)(q^{m-1}-1)..(q^{m-l+1}-1)}{(q-1)(q^2-1)\ldots(q^l-1)}$; $1 \leq l \leq m$; and $\phi(m,o,q) = 1$. Let l be a fixed integer satisfying $1 \leq l \leq m$. Let $v = \phi(m,1,q)$, $k = \phi(l,1,q)$, $b = \phi(m,l,q)$ and t be a positive integer satisfying $t \leq l$. Let A be the set of points of $PG(m-1,q)$. Corresponding to the b $(l-1)$ flats of $PG(m-1,q)$, we take blocks A_1 , A_2 , ..., A_b . The ith block A_i contains the points belonging to the ith $(l-1)$ flat, $1 \leq i \leq b$. Then for $1 \leq i \leq b$, $|A_i| = k$. A set of t-points will determine a $(c-1)$-flat where $c \leq t$. Let A' be a

set of t - points lying in a c - flat. Then the number of blocks (or (l-1)-flats) containing A' will be given by $\phi(m-c,l-c,q)$. Since $m-c \geq l-c \geq o$, $\phi(m-c,l-c,q) > o$. Therefore the blocks A_1 , A_2 , ..., A_b define a (v,k,t,b) - configuration and the corresponding local structuring is called a projective local structuring. For computational purpose, we realize the flats of PG(m-1,q) as subspaces of an m-dimensional vector space. Let V be the vectorspace of m-tuples $\underline{x} = (x_1 , x_2 , \ldots, x_m)$ of elements of GF(q) . Let $\alpha_o = 0 , \alpha_1 = 1 , \alpha_2 , \ldots, \alpha_{q-1}$ be the q elements of K = GF(q) . An (l-1) flat of PG(m-1,q) will be represented by an l-dimensional subspace of V . In particular a o-flat or a point of PG(m-1,q) will be represented by a 1-dimensional subspace of V . A 1-dimensional subspace V_1 of V will be represented by a nonnull vector belonging to V_1 . Let U_i be a set of vectors of V defined by

$$U_i = \left\{(0,0,..,0,1, \beta_1 , \beta_2 , \ldots, \beta_{m-i}) \mid \beta_j \in K , 1 \leq j \leq m-i\right\} .$$

Let $U = U_1 \cup U_2 \cup \ldots \cup U_m$. It is easily seen that any two vectors of U are linearly independent. The number of vectors of U is $q^{m-1} + q^{m-2} + \ldots + q^o = \phi(m,o,q)$. The $\phi(m,o,q)$ projective points (or 1-dimensional subspaces of V) are conveniently represented by vectors belonging to U . In other words in the projective local structuring every attribute will be identified with a vector belonging to U. The blocks or the (l-1) - dimensional subspaces for $l > o$ are more conveniently represented by matrices. Let B be an (m-l x m) matrix with entries from F and rank equal to m-l . Let $\underline{o}$ be the (m x 1) matrix with all entries equal to o . The set of vectors $\underline{x}$ which

satisfy the matrix equation

$$B\,\underline{x}^{T} = \underline{o}$$

defines an 1 - dimensional subspace. Every block and the corresponding local file will be identified by a matrix B . Let $A' = \{\underline{x}^{(1)}, \underline{x}^{(2)}, \ldots, \underline{x}^{(r)}\}$ be a set of r attributes. Let A_i be a block given by the equation $B\underline{x} = \underline{o}$. To test if $A' \subseteq A_i$, we need to check if $B\underline{x}^{(j)}$ is equal to $\underline{o}$ or not for $j = 1, 2, \ldots, r$. Thus for projective local structuring, the amount of time required to identify the local files relevant for a given query will probably be less than the same for Boolean local structuring.

11. Euclidean local structuring. Let m be a positive integer, q be a prime power and $K = GF(q)$ be the Galois field of order q . Let Σ_{m-1} be an (m-1) dimensional projective space over GF(q) . Let Σ_{m-2} be an (m-2) flat. The Euclidean space EG(m-1,q) is the space of consisting of the 1 - flats of Σ_{m-1} which are not contained in Σ_{m-2}, $1 = o, 1, \ldots, m-1$. Let $\phi(m,1,q)$ be the function introduced in the last section. The number of (1-1) - flats of EG(m-1,q) is given by $\phi(m,1,q) - \phi(m-1,1,q)$. Let A be the set of o-flats of EG(m-1,q) . Let $v = \phi(m,1,q) - \phi(m-1,1,q)$, $k = \phi(1,1,q) - \phi(1-1,1,q)$, $b = \phi(m,1,q) - \phi(m-1,1,q)$ and t be a positive integer satisfying $t \leq 1$. Corresponding to the b (1-1) - flats we take b blocks $A_1, A_2, \ldots, A_b$. The ith block A_i contains the elements of A corresponding to the o-flats contained in the ith (1-1) - flat. It is easily seen that the blocks $A_1, A_2, \ldots, A_b$ satisfy the properties of a (v,k,t,b) - configuration. The corresponding local structuring is called an Euclidean local structuring. Let V be

the $(m-1)$ - dimensional vector space of $(m-1)$ - tuples $(x_1, x_2, \ldots, x_{m-1})$ of elements of $GF(q)$. For computational purposes, the o-flats of $EG(m-1,q)$ can be taken to be the vectors of V . Let B and $\underline{a}$ be respectively $\left((m-1) \times (m-1)\right)$ and $\left((m-1) \times 1\right)$ matrices with elements from K such that the rank of B is $(m-1)$. Then the set of vectors $\underline{x} = (x_1, \ldots, x_{m-1})$ which satisfy the matrix equation

$$B \underline{x}^T = \underline{a}$$

will be an $(1-1)$ - flat of $EG(m-1,q)$. Therefore each local file of the Euclidean local structuring will be identified by a pair of matrices $(B,\underline{a})$.

12. <u>(v,k,t) - designs</u>. Let $v \geq k \geq t > o$, λ and b be positive integers. Let A be a set of v elements. Let $A_1, A_2, \ldots, A_b$ be the blocks of a (v,k,t,b) - configuration. If every t - element subset of A is contained in exactly λ blocks, then the (v,k,t,b) - configuration is called a (v,k,t,λ) - design. $(v,k,2,\lambda)$ - designs are also called balanced incomplete block designs (bibd) . It is easily seen that (v,k,t,λ) - designs are optimum (v,k,t,b) - configurations. The problem of constructing (v,k,t,λ) - designs and more generally optimum (v,k,t,b) - configurations is extremely difficult. If $k = t$, then the class of $\binom{v}{t}$ k - element subsets is a $(v,k,k,1)$ - design. Except for this trivial $(v,k,k,1)$ - design, no $(v,k,t,1)$ - design with $t > 5$ is known. For a file $F = (I, A, f)$ occurring in practice, $\underset{i \in I}{\text{Max}} |f(i)|$ will be greater than 5 . Therefore the known $(v,k,t,1)$ - designs will not be very useful for constructing local structuring. The following table

shows the parameters of the known $(v,k,t,1)$ - designs with $k > t \geq 3$.

v	k	t
$q^m + 1$ q a prime power and m a positive integer	q + 1	3
v = 2 or 4 (mod 6)	4	3
$v = \frac{4^m + 2}{3}$ $m \geq 3$, an integer	5	3
11	5	4
12	6	5
23	7	4
24	8	5

There is a large literature on bibd. Since the known $(v,k,t,1)$-designs will not be of much practical interest for constructing local structuring, we do not discuss the details of construction of these designs. However we like to emphasize the point that the theory of $(v,k,t,1)$-designs for large t will be vitally important for constructing efficient local structuring.

13. An algorithmic approach for constructing (v,k,t,b)-configuration. Let $v \geq k \geq t > o$ be integers. Let A be a v-element set and $\mathcal{B} = \{A_1, A_2, \ldots, A_{b'}\}$ be such that $A_i \subseteq A$ and $|A_i| \leq k$, $1 \leq i \leq b'$. For $A' \subseteq A$, define

$$c(A',\mathcal{B}) = |\{i : |A' \cap A_i| \geq t, 1 \leq i \leq b\}|$$

Let $C = (A,\mathcal{B})$ be a (v,k,t,b)-configuration with $\mathcal{B} = \{A_1, A_2, \ldots, A_b\}$. Let $\mathcal{B}_i = \{A_1, A_2, \ldots, A_i\}$, $1 \leq i \leq b$. The configuration C is said to be locally optimum iff for $\forall\, i, 2 \leq i \leq b$, $A' \in P_k(A)$, $c(A_i,\mathcal{B}_{i-1}) \leq c(A',\mathcal{B}_{i-1})$. Clearly a $(v,k,t,1)$-design is a locally optimum (v,k,t,b)-configuration. After choosing the first $i-1$ blocks, we try to choose the ith block such that the number of new t element subsets covered by the ith block is as large as possible.

14. Multistage local structuring. Let $F = (I, A, f)$ be a file such that $|A| = v$, $|I| = n$ and $\operatorname{Max}_{i \in I} |f(i)| = t$. Let $\ell = (\pi, \delta)$ be a local structuring of F with $\pi = (I_1, I_2, \ldots, I_{b_1})$ and $\delta = (A_1, A_2, \ldots, A_{b_1})$. Let $\operatorname{Max}_{1 \leq i \leq b} |A_i| = k_1$ and $F_j = (I_j, A_j, f_j)$, $1 \leq j \leq b_1$ be the local files. If $k_1 > t$, and each $|I_j|$ is large, we can expect to improve the organization of the local files F_j by applying

the principle of local structuring. A 2-stage local structuring L of the file F is a (b_1+1)-tuple $(\ell, \ell_1, \ldots, \ell_{b_1})$ where ℓ is a local structuring of F with local files $F_1, \ldots, F_{b_1}$ and ℓ_j is a local structuring of the file F_j, $1 \leq j \leq b_1$. Let $\ell_j = (\pi_j, \delta_j)$, $1 \leq j \leq b_1$. Suppose $\pi_j = (I_{j1}, I_{j2}, \ldots, I_{jb_2})$ and $\delta_j = (A_{j1}, A_{j2}, \ldots, A_{jb_2})$, $1 \leq j \leq b_1$. Let $F_{jl} = (I_{jl}, A_{jl}, f_{jl})$ where f_{jl} is the restriction of f_j to I_{jl}, $1 \leq j \leq b_1$, $1 \leq l \leq b_2$. Let s_{jl} and r_{jl} respectively denote storage and retrieval rules for the local files F_{jl}, $1 \leq j \leq b_1$, $1 \leq l \leq b_2$. For $A' \subseteq A$ let

$$J_{A'} = \left\{(j, l): 1 \leq j \leq b_1, 1 \leq l \leq b_2, A_{jl} \supseteq A'\right\}$$

We can define a storage rule s and retrieval rule r for the file F by setting

$$s(i) = s_{jl}(i), \text{ if } i \in I_{jl}$$

and

$$r(A') = \bigcup_{(jl) \in J_{A'}} r_{jl}(A')$$

To get an expression for $T_L(r)$, the average retrieval time required for a query of size r, we assume that $|A_j| = k_1$ and $|A_{jl}| = k_2$ for $1 \leq j \leq b_1$ and $1 \leq l \leq b_2$. The subsets A_j and A_{jl} are respectively called first stage and second stage blocks, $1 \leq j \leq b_1$ $1 \leq l \leq b_2$. We also assume that for each second stage local file F_{jl} inverted filing system with lexicographic storage and retrieval rules are used. As before let d' and c respectively denote the amount of time required to compare a pair of attribute numbers and a pair of accession numbers. Let $\nu_p(A')$ denote

the number of pth stage blocks containing A', $p = 1, 2$. We assume that

$$|I_{j1}| = \frac{n}{b_1 b_2} \quad \text{and} \quad |I_{j1}(A')| = \frac{n}{b_1 b_2 \binom{k_2}{r}} .$$

To retrieve for a query A' of size r the computer will take $d' b_1 r$ units of time to check the relations, $A' \subseteq A_j$ $1 \leq j \leq b_1$. For each of the first stage block A_j containing A', the computer will take $d' b_2 r$ units of time to determine the second stage blocks which contain A'. Finally for each second stage block A_{j1}, the comparison of accession numbers will require $\frac{n}{b_1 b_2} \sum_{i=1}^{r-1} \frac{1}{\binom{k_2}{i}}$ units of time. Therefore for $r > 1$ we get the expression

$$T_L(A') = d'(rb_1 + \nu_1(A')rb_2) + (\nu_2(A') \frac{c\, n}{b_1 b_2} \sum_{i=1}^{r-1} \frac{1}{\binom{k_2}{i}} \qquad \text{.....(21).}$$

Let $d \geq 1$ be a positive integer. A d-stage local structuring L for the file F is a $(1 + \sum_{p=1}^{d-1} b_1, b_2 \ldots b_p)$-tuple $\left(\ell, \ell_{j_1 j_2 \ldots j_p}, 1 \leq p \leq d-1, 1 \leq j_p \leq b_p\right)$ where $b_1, b_2, \ldots, b_{d-1}, b_d$ are positive integers, $\ell = (\Pi, \delta)$ is a local structuring of F, $\Pi = (I_1, I_2, \ldots, I_{b_1})$, $\delta = (A_1, A_2, \ldots, A_{b_1})$, $\ell_{j_1 \ldots j_p} = (\Pi_{j_1 \ldots j_p}, \delta_{j_1 \ldots j_p})$, $\Pi_{j_1 \ldots j_p} = (I_{j_1 \ldots j_p 1}, \ldots, I_{j_1 \ldots j_p b_{p+1}})$, $\delta_{j_1 \ldots j_p} = (A_{j_1 \ldots j_p 1}, \ldots, A_{j_1 \ldots j_p b_{p+1}})$, $\ell_{j_1 \ldots j_p}$ is a local structuring of the pth stage local file $F_{j_1 \ldots j_p} = (I_{j_1 \ldots j_p}, A_{j_1 \ldots j_p}, f_{j_1 \ldots j_p})$, $F_{j_1 \ldots j_p}$ is a local file of $\ell_{j_1 \ldots j_{p-1}}$, $1 \leq p \leq d-1$, $1 \leq j_p \leq b_p$. The subsets $A_{j_1 \ldots j_{p-1}, j_p}$ are called the pth stage blocks. In our definition, we require that the number of pth stage blocks for each local

structuring $\ell_{j_1 \cdots j_{p-1}}$ is a constant number. This assumption is, of course not necessary. For the sake of simplicity we assume that the cardinality of each pth stage block is k_p, $1 \leq p \leq d$. Suppose inverted filing systems with lexicographic storage and retrieval rules are used for the dth stage local file. Further we assume that $|I| = n$, $|I_{j_1 \cdots j_d}| = \frac{n}{b_1 b_2 \cdots b_d}$, $|I_{j_1 \cdots j_d}(A')| = \frac{n}{b_1 \cdots b_d \binom{k_d}{r}}$, for $A' \in P_r(A)$. Let $\nu_p(A')$ denote the number of pth stage blocks which contain A'. Arguing as in the case of 2-stage local structuring, we get

$$T_L(A') = d'\left(rb_1 + \sum_{p=1}^{d-1} \nu_p(A')\, b_{p+1}\, r\right) + \nu_d(A')\, \frac{c\, n}{b_1 \cdots b_d} \sum_{i=1}^{r-1} \frac{1}{\binom{k_d}{i}} \qquad \ldots\ldots(22).$$

15. Multistage boolean local structuring. Suppose the first stage local structuring ℓ is a $(v, k_1 = v-1, t)$ Boolean local structuring and each second stage local structuring is a $(v-1, k_2 = v-2, t)$ - Boolean local structuring. Suppose in general each pth stage local structuring is a $(v-p+1, k_p = v-p, t)$- Boolean local structuring with $p = 1, 2 \ldots (v-t)$. Then we have $k_1 = v-1$, $k_2 = v-2$, and $k_{v-t} = t$. Also for $A' \in P_r(A)$, $r \leq t$, $\nu_1(A') = v-r$, $\nu_2(A') = (v-r)(v-1-r), \ldots \nu_p(A') = (v-r)(v-1-r)\ldots (v+1-p-r)$, $p = 1, 2, \ldots (v-t)$

and $\quad b_1 = v, b_2 = v-1, \ldots b_{v-t} = t+1$.

The expression (22) simplifies to

$$T_L(r) = d'r\left(v + \sum_{p=1}^{v-t-1} (v-r)(v-1-r)\ldots(v+1-p-r)(v-p)\right) + \qquad \ldots\ldots(23).$$

$$\frac{c\, n (v-r)(v-1-r)\ldots(t+1-r)}{v(v-1)\ldots\ldots(t+1)} \sum_{i=1}^{r-1} \frac{1}{\binom{t}{i}}$$

Comparing (20) and (23), we get

$$T_L(r) - T_{\ell_o}(r) = a(d,r,v,t) + c\,n \sum_{i=1}^{r-1} \left(\frac{(v-r)\,(v-1-r)\ldots(t+1-r)}{(v-i)\,(v-1-i)\ldots(t+1-i)} - 1 \right) \frac{1}{\binom{v}{i}}$$

where $a(d,r,v,t)$ is a function of d,r,v and t. If $1 \leq i \leq r-1$, $\frac{(v-r)\,(v-1-r)\ldots(t+1-r)}{(v-i)\,(v-1-i)\ldots(t+1-i)} < 1$. Therefore, for fixed c,d,r,v,t, $r > 1$ and sufficiently large n, $T_L(r)$ will be smaller than $T_{\ell_o}(r)$. Multistage projective and Euclidean local structurings can be developed in a manner similar to that of multistage Boolean Local structuring.