CONSECTUVIE RETRIEVAL INFORMATION SYSTEM

by

Kapali P. Eswaran

Memorandum No. ERL-M384

11 May 1973

CONSECUTIVE RETRIEVAL INFORMATION SYSTEM

by

Kapali P. Eswaran

Memorandum No. ERL-M384

11 May 1973

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

TO MY PARENTS

ACKNOWLEDGMENTS

I wish to thank sincerely and gratefully Professor Lotfi A. Zadeh for supervising the research and for his invaluable guidance, help and advice in my academic and non-academic life. I would like to express my appreciation to Professor Chittoor V. Ramamoorthy and Professor William Cooper for their patience, suggestions and criticisms on the manuscript. Special thanks are extended to Professors Richard M. Karp, Eugene L. Lawler, Sreeniva S. Reddi and Dr. Sakti P. Ghosh for many helpful ideas. Finally I want to thank my friend Lynne O'Dell. She gave me all the encouragements and moral support. The thesis might never have been completed without her aid and love.

The author acknowledges the support of the Naval Electronics Systems Command, Contract N00039-71-C-0255 during this research.

TABLE OF CONTENTS

Consecutive Retrieval Information System

Ph.D. Dissertation Kapali P. Eswaran Department of Electrical Engineering and Computer Sciences

L. A. Zadeh
Chairman of Committee

ABSTRACT

The work reported here is in the nature of a general optimization problem applied to storage organization techniques. We assume that we know in advance the set of queries Q regarding a file F. The file F is to be stored on a "linear storage medium" so that efficiency of retrieval is maximum and the storage space required is minimum. A storage medium S is called "linear" if the storage locations of S can be arranged linearly and the access time between any two locations is an increasing function of the distance between them. We analyse the existing file organizations: sequential, index sequential, inverted and multilist file organizations. We observe that though the efficiency of retrieval in an inverted file organization is high, because of unnecessary generality it is not high enough. Based on the principle of inverted file organization, a query inverted file organization (QIFO) is proposed. Let $Q = \{q_1, q_2, \ldots, q_m\}$ be the family of queries regarding a file F. In a QIFO for Q w.r.t. F, for each query $q_i \in Q$ all the records that are relevant to q_i are stored in consecutive storage locations. If in a QIFO, each record is

stored only once then the QIFO is called a Consecutive Retrieval File Organization (CRFO) [5]. QIFO provides minimum overall retrieval time for all queries and CRFO guarantees minimum storage space in addition to minimum overall retrieval time. The CR property is generalised to a set theoretic one. Let $Q = \{q_1, q_2, \ldots, q_m\}$ be a family of finite, non-empty sets and $S = \bigcup_{q_i \in Q} \{q_i\}$. Suppose there exists a one-to-one function f that maps the elements of S into points in the real line R such that for each set $q_i \in Q$ there is an interval I_i containing images of all elements $\in q_i$ but not images of any elements not in q_i. Then we say that Q is linearly orderable (L.O.). If we take each query $q_i \in Q$ as a set of elements (a record corresponding to an element), a query family Q has CR property iff the corresponding family of sets Q has the L.O. property. We take a graph theoretic approach to investigate the conditions under which Q has the L.O. property. We establish a connection between the L.O. property of Q, the intersection graph of Q [8] and an interval graph [8], [9]. We obtain necessary and sufficient conditions for Q to be linearly orderable. We give an algorithm to find a linear ordering if one exists and discuss the complexity of the algorithm. After developing a CRFO on the assumption that the file and the family of queries are time invariant, the problems of updating a CRFO are taken up. Some nice results are developed. Several file organizations that have similar structure as CRFO are introduced. They are theoretically interesting and have practical importance.

— ∗ —

CHAPTER 1.

INTRODUCTION

Information retrieval is a means by which volumes of information can be organized and handled in an efficient manner. The storage and handling of such vast volumes of information by conventional non-computerized methods has become in many cases a formidable task. Computers are being used today to take over information retrieval. They can perform millions of calculations in a second, handle mountains of data and in many cases perform more efficiently and accurately than humans.

Computerized information retrieval has become common in every field of industry: research, engineering, finance, management, marketing, manufacturing, inventory control and employee and community relations. Hospitals maintain computerized histories of medical records on all patients. Criminal investigation, census and taxation are some of the governmental agencies in which computerized information retrieval is prevalent. Space technology is another field where pertinent facts must be readily available in order to take split-second decisions. It can be safely concluded that a computerized file organization is needed in many governmental, social or industrial institutions that require complete, accurate information immediately.

Along with the increase in speed of computers, there has been an increase in the volume and complexity of information to be handled and there is a growing demand from the users to get the information fast. This is particularly true in time-sharing computer systems. The users in a time-sharing environment expect the system to retrieve and respond with the information pertinent to the questions asked as quickly as possible. The response time and the amount of storage space depend on how the information is organized in the computer.

Then, in order to use a computer properly for information retrieval, it is important to acquire a good understanding of the structural relationships present within the information (also called data) and a good knowledge of how the data is going to be used by the users. Usually the programmers or the system analysts who design the organization for a file are not aware of the needs of the users using the file. They provide too much generality or too little.

A file is accessed by a user through queries. Suppose we know in advance the queries the users are going to ask about a particular file. We also know the answers to these queries. Let the queries be equally probable. Using minimum storage space, is there any file organization that guarantees minimum overall retrieval time for all queries? The answer is in the affirmative. We shall consider in detail such a file organization. First let us introduce some terminology and the existing file structures.

CHAPTER 2.

FILE ORGANIZATIONS

Generally speaking, the term "record" refers to a fact or an event or object of some kind. Consider the following sentence. Sylvania can supply part number 300, which is a printed circuit board, from their location at Palo Alto, California within 10 days at a cost of $20. If this is a fact, then this will be a record. In order that facts may be stored, retrieved and updated effectively in a computer, we need to put some structure to this loosely stated word "record". If we look into our example more closely, we can see that a structure is hidden in it. Sylvania is a company that can supply part number 300 having printed circuit board as its name from the location Palo Alto, California within a period of 10 days at a cost of 20 dollars. The statement assigns "values" to the "attributes": company, part number, name of the part, location of the company, period of delivery and cost of the part. Formally, a record is then a set of attribute-value pairs. Our example thus becomes the record {(Name of the company, Sylvania), (part number, 300), (name of part, printed circuit board), (location of company, Palo Alto-California), (period of delivery, 10 days), (cost of the part, 20 dollars)}. We note that in our definition, we do not assign any particular position to any element of the record. If on the other hand, we require that each attribute-value pair in a record gets assigned a position defined by a format, we have a formatted-record.

Considering our example, let the format be:

Name of company = 15A (i.e., 15 characters, alphanumeric, right justified, blank filled) → (symbol → denotes followed by) Part number = 3I (3 digits, integer) → Name of part = 9A → Location of company = 15A → Period of delivery = 2I → Cost of part = D6 (Decimal. The right most 2 digits constitute the cent part).

Our record then is "bbbbbbbSYLVANIA300P.C.BOARDPALOALTO-CALIF10bb2000". b stands for the blank symbol.

A file is a finite collection of distinct records. Though a good definition of a file should take into account the time-varying aspect of the record collection (due to deletion and addition of records), we shall not consider this in the definition. This introduces unnecessary complications. We shall consider the file to be invariant over a span of time. The updating problems are considered separately in chapter 5. Strictly speaking, we do not require that a file, when composed of formatted-records, have a single fixed format for all the records in the file. We may have a single global format for all the records in the file and for the records which do not conform to the global format, local formats may be used and stored as a part of the records themselves. For simplicity, we shall consider the files as having a single format for all its records. First, we shall give an example which describes a file in its logical level.

Let each record of a file F have attributes A_1, A_2, ..., A_n.

Let V_i be the set of values that A_i can take. Then the file F is a relation which is a subset of the product space $V_1 \times V_2 \times \ldots\ldots \times V_n$ and may be represented as a table [7]. This kind of representation of a file is called a logical organization as opposed to a storage organization which deals with the actual storing of the file internally in the computer storage.

Example 2.1: Consider a file in which each record has the following attributes: Part number, Supplier Name and Priority of the part. Let the attribute part number take integer values from 100 to 999, supplier name be any alphanumeric string up to 7 characters and priority take any one of the three values, namely CRITICAL, ESSENTIAL and ORDINARY. Figure 2.1 represents a file at the logical level. Each row of the table corresponds to a record.

File Organization Techniques:

Any question that is asked by a user about a file which results in accessing the file and retrieving information is called a query. Given any file, we need to arrange the file on one or more storage media so that the records pertinent to user queries may be retrieved effectively. This arrangement is called the file organization. File organization is storage organization. In this section, we shall discuss a number of existing file organizations and compare them. We will assume that the storage media on which the file is to be stored are secondary storages like tape, drum or disk. We shall use the file in example 2.1 to exemplify the various

100	JONES	ESSENTIAL
105	JOHN LO	CRITICAL
100	FRISCH	ESSENTIAL
110	BOSECH	ORDINARY
105	DAD & SON	CRITICAL
115	JONES	CRITICAL
120	FRISCH	CRITICAL

Figure 2.1. Representation of a File Tabular Form (Relational Model).

file organizations. The symbol * shall be used as a delimitter or separator between the records of a file.

Sequential File Organization:

In a sequential file organization, the records of a file are stored sequentially on the storage medium. To retrieve information, it is required to check the values of the different attributes of all records of the file till the desired record is reached. Figure 2.2 gives the sequential file organization of the file in example 2.1. For instance, to retrieve information about part number 100 we need to search the whole file to find the record with part number 100 and then retrieve the supplier name and priority. If there is more than one such record, we may need to retrieve all of them. In our example there are 2 records with part number 100. Jones and Frisch are the suppliers and ESSENTIAL is the priority of the part. If we are allowed to assume that the user queries will be based on a specific attribute of the records in the file, then in order to improve the efficiency of retrieval we may (i) use a hash function and hash the value of the attribute (called the key) and get the address where the record is to be stored or (ii) order the records of the file on the value of that attribute and instead of using an exhaustive linear search, resort to efficient search techniques like the binary search [2].

The hash coding scheme provides an easily computable function f which maps a set of keys onto a set of address spaces. The function is, in general, found by trial and error and may not be one-to-one. When the function is many-to-one, more than one key

```
100bbbJONESESSENTIAL*105JOHNLObCRITICAL*100b
FRISCHESSENTIAL*110bBOSECHbORDINARY*105DAD&S
ONbCRITICAL*115bJONESbCRITICAL*120bFRISCHbCR
ITICAL*$
```

STORAGE

MEDIUM

<u>Figure 2.2</u>. <u>Sequential File Organization</u>

maps to the same address and this is referred to in literature as collision. The collision may be resolved in a number of ways as described in [2]. Apart from collision, the hash coding technique suffers from the disadvantage that the available space for storage must be known in advance. Thus, it does not allow for dynamic growth of data.

When we know the attribute of interest, we can use some ordering techniques to linearly order the records in the file and use efficient searching schemes to retrieve information. For example, our file will be as shown in figure 2.3 when it is ordered on the attribute part number.

In the case of an ordered sequential file, the retrieval program fails to operate correctly if we need to retrieve the reply for a query based on an attribute other than the one on which the records are sorted. To take care of this situation, we may have two kinds of retrieval programs: one for the queries based on the attribute on which the file is ordered and another for queries based on other attributes. The latter program will be a linear search. When we have two programs, naturally the complexity of the retrieval system is increased.

The most important disadvantage in the case of ordered sequential file is maintaining the order while records are inserted into the file. Even when a sequential file is not ordered, each updating event generally may require relocation of many stored records. For example, deletion of a record gives rise to the problem of compacting the new file since the storage locations

Part #	Supplier Name	Priority of Part
100	JONES	ESSENTIAL
100	FRISCH	ESSENTIAL
105	JOHN LO	CRITICAL
105	DAD & SON	CRITICAL
110	BOSECH	ORDINARY
115	JONES	CRITICAL
120	FRISCH	CRITICAL

Figure 2.3. File Ordered on Part Number

of the deleted records will be wasted otherwise. In order that the updating of the file will be flexible, we have the index sequential file organization.

Index Sequential File Organization:

An index is assigned to each record of the file. We then build a directory of index-address pairs. A pair (I, A_I) means that the record with index I is stored starting at address A_I. There need not be any relation between indices and the addresses. The records are stored randomly on the storage medium. The directory may be stored sequentially, ordered or unordered. The directory is shown as a two dimensional array in figure 2.4. Accessing a record is now a two-step process. First, search the directory and get the addresses of the records of interest and then get the records themselves. Insertions and deletions are made easier. When inserting records, there is no movement of records involved. At worst, we may have to reorder the directory. Storage space allocated to the directory is of course an overhead.

Inverted File Organization

One of the disadvantages of the sequential file organization is that when a query is based on some attribute other than the one used for ordering or hash coding, the file needs to be searched exhaustively. In the case of an index sequential file, the same thing holds if a query is based on an attribute not related to the index. Hence, we need to make the directory of the index sequential file organization more general. An organization

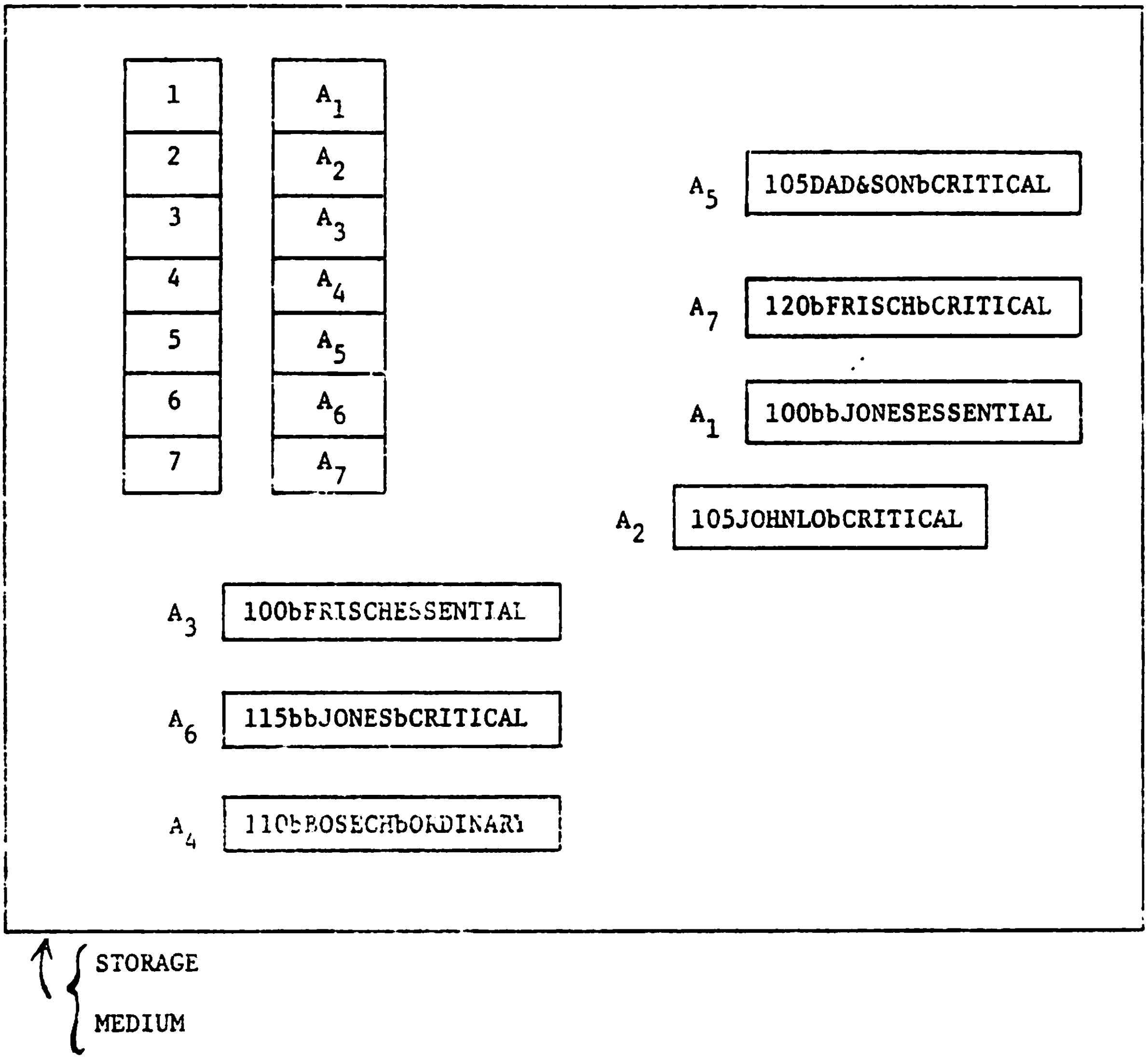

Figure 2.4. Index Sequential File Organization

Each record is assigned an unique index (an integer between 1 and 7 in this case). There is no apparent relation between locations A_1, A_2, ..., A_7.

of such a kind is called an <u>inverted file organization</u>. An entry in the directory is of the form "attribute, value of the attribute, set of addresses of the records in which the attribute has the value indicated". Records are stored randomly in the storage medium. If there is no overlap of the set of values that different attributes can take, we need not store the attributes as a part of an entry in the directory. The inverted file organization for our example is shown in <u>figure 2.5</u>. The directory is shown as an one dimensional array. To answer a query like "give the part numbers and the priorities of the parts which are supplied by JONES", we search the directory for supplier JONES and get addresses A_1 and A_6. We then do fetches at addresses A_1 and A_6 to get the necessary information.

If the query is based on Boolean combination of many attributes then the directory is searched once for each attribute. Set operations like union, intersection corresponding to Boolean operations "or" and "and" are performed on these attributes and the records fetched. For example to answer the Boolean query "give the part numbers of the parts that are ESSENTIAL <u>and</u> supplied by JONES", we first search the directory for PRIORITY, ESSENTIAL and obtain the set of addresses $\{A_1,A_3\}$. We next search the directory for SUPPLIER, JONES and obtain $\{A_1,A_6\}$. $\{A_1,A_3\} \cap \{A_1,A_6\} = \{A_1\}$ is the set of addresses corresponding to the records that are relevant to the query.

Clearly, the efficiency of retrieval is improved in an inverted file organization. The directory consumes a large amount of space. Further, the space requirement for each entry in the directory

Part No., 100, A_1, A_3
Part No., 105, A_2, A_5
Part No., 110, A_4
Part No., 115, A_6
Part No., 120, A_7
Supplier, JONES, A_1, A_6
Supplier, FRISCH, A_3, A_7
⋮
Supplier, BOSECH, A_4
Priority, CRITICAL, A_2, A_5, A_6, A_7
Priority, ORDINARY, A_4
Priority, ESSENTIAL, A_1, A_3

A_4	110bBOSECHbORDINARY
A_5	105DAD&DONbCRITICAL
A_3	100bFRISCHESSENTIAL
A_6	115bbJONESbCRITICAL
A_1	100bbJONESESSENTIAL
A_7	120bFRISCHbCRITICAL
A_2	105bJOHNLObCRITICAL

Figure 2.5. Inverted File Organization. There is no apparent relation between locations A_1, A_2, ..., A_7.

is not predictable - it shrinks and grows as the records are inserted and deleted from the file. The multilist file organization provides a solution to this problem.

Multilist File Organization

Instead of storing addresses of all records corresponding to an attribute-value pair in the directory, we chain these records as a list and store the address of the start of the list in the directory. The successive records are obtained by means of pointers. There is a list per each attribute-value pair of interest and lists can intersect by having one or more records in common. The terminology "multilist" stems from the fact that a record may be (and in general is) a member of many lists. In order to accomodate the pointers, the format of the records is slightly changed. In a formatted file, each member or element of the record is a "value-pointer" pair (see figure 2.6). In an unformatted file, each element of the record is a three tuple, "attribute, value and pointer to the next record in which the attribute has this value".

In the example in figure 2.7, ϕ indicates the null pointer or end of list. To get the part numbers that are CRITICAL, we first search the directory and get the address A_2. The list of parts that are CRITICAL starts with the address A_2. By treading through the list, we find that the rest of the records in the list have the addresses A_5, A_6, and A_7. We thus obtain the part numbers 100, 105, 115, and 120.

The storage space required for multilist file organization is

- - - - - - - -	Value of Attribute A_i	Pointer to the Next Record in which A_i Has the Same Value	Value of Attribute A_{i+1}	- - - - - - - -

<u>Figure 2.6</u>. <u>Format of a Multilist File Organization</u>. Each element of a formatted record is a value-pointer pair.

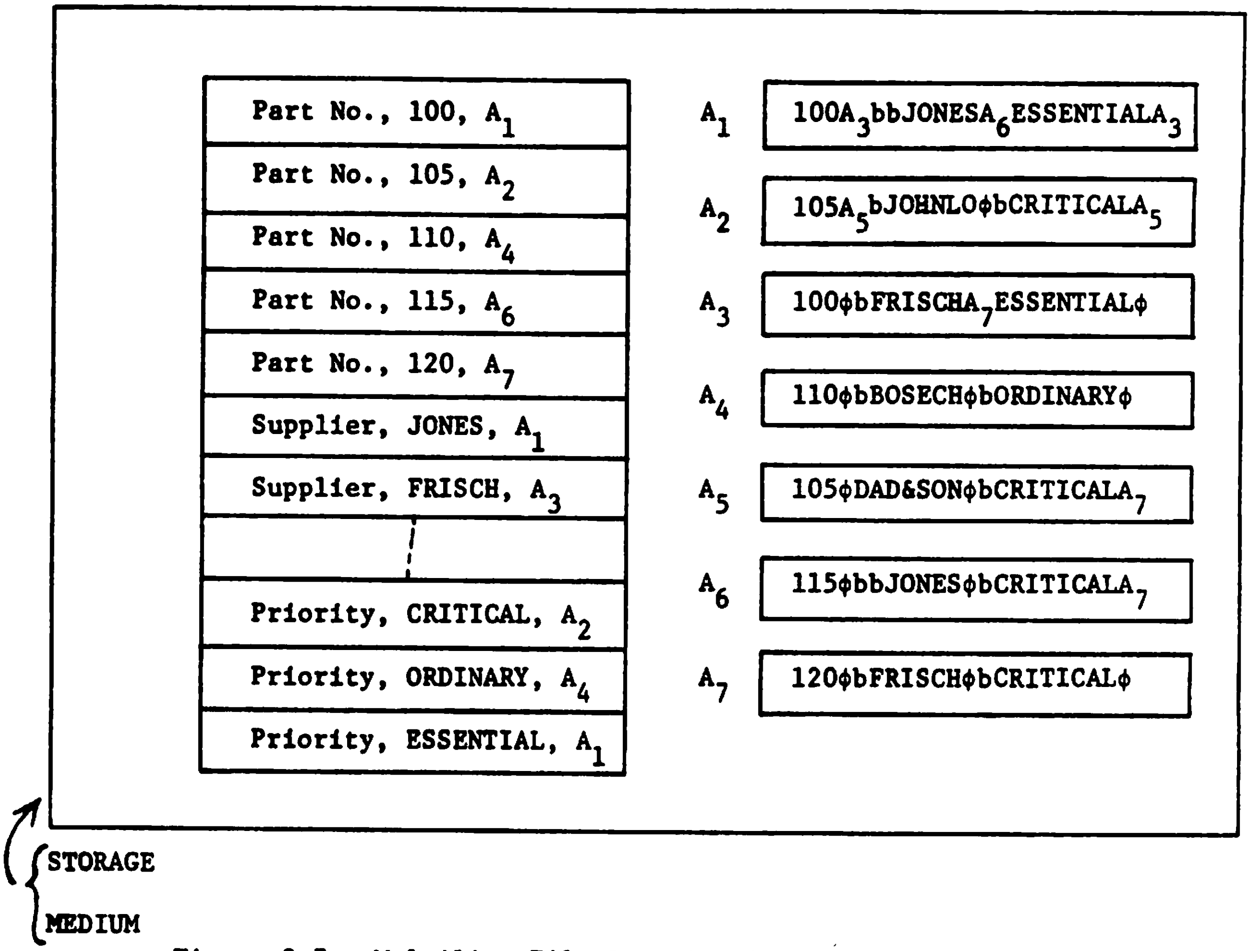

Figure 2.7. Multilist File Organization.

about the same as inverted file organization. It is unfortunate that these definitions of file organizations are not standard in the literature. Our definitions conform more or less to those found in [1], [2], [3] and [4].

CHAPTER 3.

QUERY INVERTED AND CONSECUTIVE RETRIEVAL FILE ORGANIZATIONS

In chapter 2 we discussed the existing file organizations. In this chapter we shall introduce a special kind of inverted file organization called the query inverted file organization and define consecutive retrieval file organization.

The attribute-value pair is commonly known as a field. Normally, a field (or a combination of fields) of a given file uniquely identifies each record of the file - i.e., there exists a one-to-one onto function that maps the field onto a record of the file. Such a field is called a primary field (or a discriminatory field) and the attribute of a primary field, a primary key. Fields other than the primary field(s) are called data fields.

We haven't thus far spoken much about the storage medium on which the file is stored. A storage medium, S, is called linear if the storage locations of S can be arranged linearly and the access time between any two storage locations is an increasing function of the distance between them. Usually the distance is directional. Another common name for such media is sequential-access devices. Tapes, tracks of a disk, books in a library shelf, shops in a street are examples of such devices. Further we shall require that the storage device be one dimensional, i.e. the shelves have only one deck, the shops have just one floor, etc. In other words we are limiting "locality or proximity" of storage locations to one dimension.

Suppose we are given, apriori, the set of queries Q that will be asked by the users regarding the file. Then, how well can we organize the file so that the overall time for retrieving the relevant records for all queries is minimum? We also assume that the queries belonging to Q are equally likely. One way to achieve this is as follows: For each query $q_i \in Q$, we store the records pertinent to q_i in consecutive storage locations. An entry in the directory will consist of "the query, the starting and the end addresses of the block of reply records for this query". The reply records for each query is stored consecutively. This kind of file organization is called the <u>Query-Inverted File Organization</u>. We note the similarity between query inverted file organization and inverted file organization. In the inverted file organization, an entry in the directory consists of the attribute-value pair of interest (called the <u>secondary indices</u>), and addresses of records in which the attribute has this particular value. In the query inverted file organization, the secondary index corresponds to a query. Further, in order to achieve minimum retrieval time, we require that the records pertinent to a query be stored in consecutive locations. If a record is pertinent to more than one query, it may be stored more than once. If a record is stored more than once, it is called <u>redundant</u>.

When the queries belonging to Q are based on a single value of a primary key, each query has only one record as its reply.

Hence, in this case a sequential file organization is a query-inverted file organization. If the queries are related to only one value of a data field, say f_i, then all the records in which the attribute of f_i takes this particular value can be stored in consecutive storage locations. Such an inverted file organization is a query inverted file organization and does not have any redundant storage of records. However, if the queries relate to more than one field (primary or data),or more than one value of a single field, then records may have to be stored redundantly to get a query inverted file organization.

Let us consider an example. We will consider the same file as in example 2 of Chapter 2 (figure 3.1). Let the family of queries Q be $\{q_1,q_2,q_3\}$ and be as follows:

(q_1) Give the list of part numbers and suppliers of the parts that are ESSENTIAL.

(q_2) Give the list of part numbers and priorities of the parts supplied by JONES.

(q_3) Give the part numbers of the parts supplied by JONES that are CRITICAL and supplied by FRISCH that are ESSENTIAL.

Reply to (q_1) is the record set $\{r_1,r_3\}$, to (q_2) is the set $\{r_1,r_6\}$ and to (q_3), $\{r_3,r_6\}$. $r_1r_3r_6r_1$ is a query inverted file organization for this set of queries and replies. This is shown in figure 3.2. An entry in the directory consists of a query, beginning address of the reply block and end address of the reply block. Note that the record r_1 is stored twice. There is no query inverted file organization for this set of queries and replies with no redundancy. Locations A_1,A_2, A_3 and A_4 are consecutive.

r_1	100	JONES	ESSENTIAL
r_2	105	JOHN LO	CRITICAL
r_3	100	FRISCH	ESSENTIAL
r_4	110	BOSECH	ORDINARY
r_5	105	DAD & SON	CRITICAL
r_6	115	JONES	CRITICAL
r_7	120	FRISCH	CRITICAL

Figure 3.1.

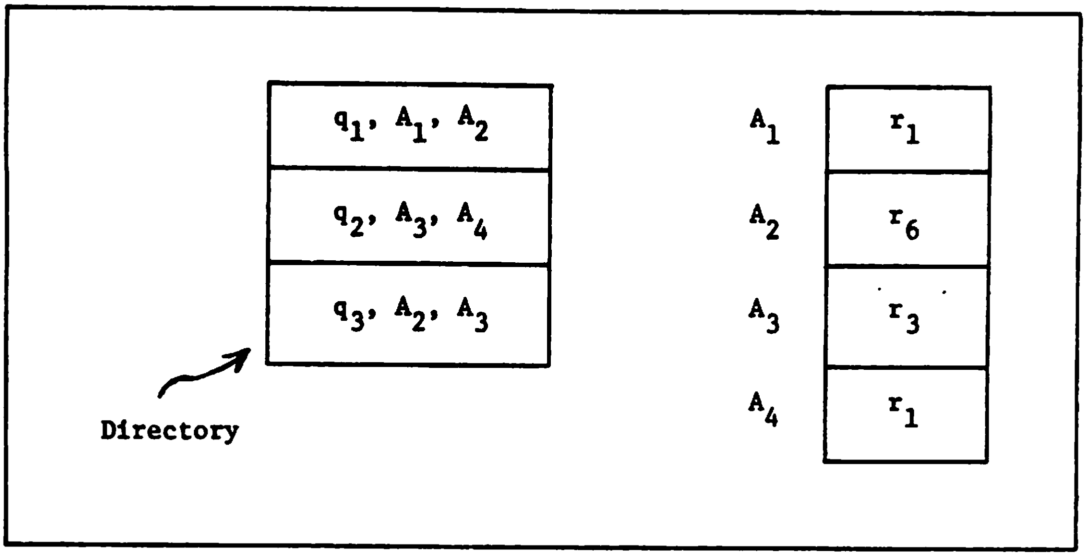

Figure 3.2. Query-Inverted File Organization. Note that locations A_1, A_2, A_3 and A_4 are consecutive. r_1 is a redundant record.

Suppose that a query family Q is such that there exists a one-to-one function f which maps the records belonging to the file F into storage locations of a linear storage medium satisfying (i) for each query $q_i \in Q$, there exists a sequence s_i of consecutive storage locations containing all records pertinent to q_i and (ii) s_i does not contain any record not pertinent to q_i. We then say that the family of queries Q has the consecutive retrieval property (CR property). A file organization having this property is called a CR organization. It is a query inverted file organization with no redundancy. The CR organization, then, provides minimum overall retrieval time for all queries with no redundancy. Let us consider the same file in the previous example (figure 3.1). Let a family of queries Q be $\{q_1,q_2,q_3,q_4\}$ and be as follows:

(q_1) Give the list of part numbers and suppliers of the parts that are CRITICAL.

(q_2) Give the list of part numbers and priorities of the parts supplied by FRISCH.

(q_3) Give the supplier names and priority of the part number 105.

(q_4) What are the part numbers and priorities of the parts that BOSECH and/or JONES supply.

Records r_2,r_5,r_6,r_7 are replies to (q_1), r_3, r_7 to (q_2), r_2, r_5 to (q_3) and r_1, r_4, r_6 to (q_4). We need to find a CR file organization for this set of queries and replies. $r_3\ r_7\ r_2\ r_5\ r_6\ r_1\ r_4$ is such an organization. This is shown in figure 3.3. Locations A_1 through A_7 are consecutive.

Query inverted file organization is mentioned in reference

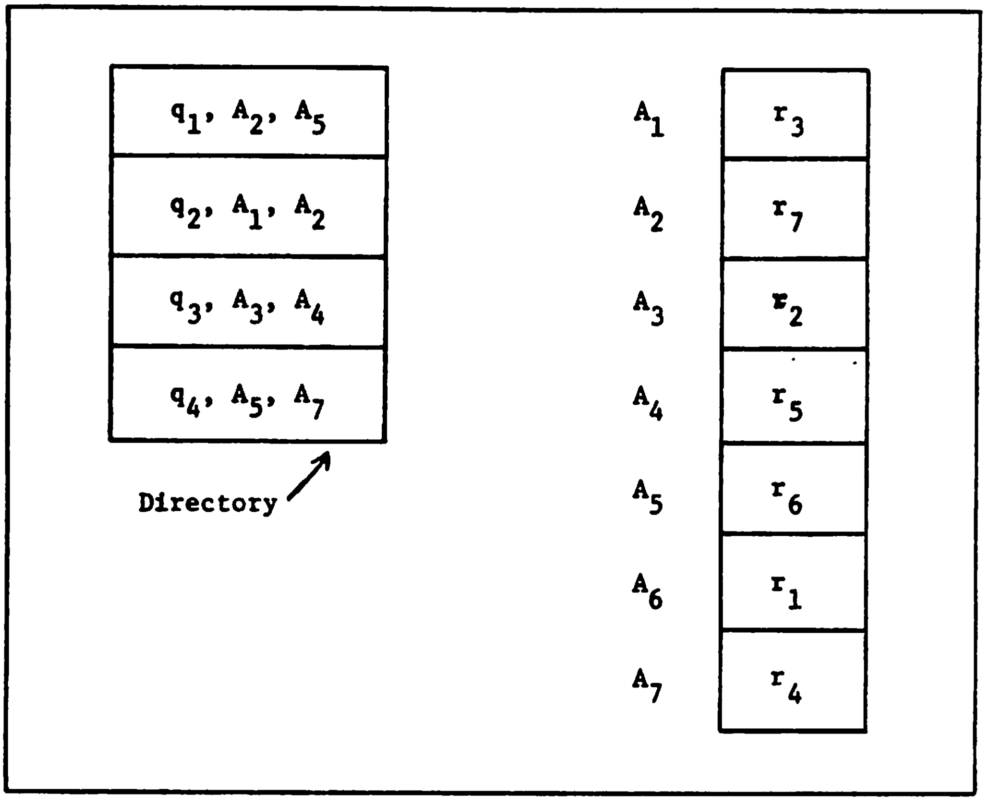

Figure 3.3. **Consecutive Retrieval File Organization.** Locations A_1 through A_7 are consecutive. There is no redundant record.

[5] and consecutive retrieval file organization in references [5] and [6].

CHAPTER 4.

CONSECUTIVE RETRIEVAL FILE ORGANIZATION AND LINEARLY ORDERABLE SETS

In this chapter we derive a number of results regarding consecutive retrieval file organization and give necessary and sufficient conditions for a family of queries to be consecutively retrievable with respect to the given set of reply records. We shall first assume that the file is time invariant and develop the results. Problems of updating a file with CR property is considered in detail in Chapter 5.

Let $Q = \{q_1, q_2, \ldots, q_m\}$ be a family of queries regarding the file under discussion. We can consider each query $q_i \in Q$ as a set of elements where each element corresponds to a record that is pertinent q_i. Queries q_i and q_j belonging to Q have some pertinent records in common means that $q_i \cap q_j \neq \phi$ when we consider the queries as sets.

Hereafter Q will denote the family of sets $\{q_1,q_2,\ldots,q_m\}$ and S the set $\bigcup_{q_i \in Q} \{q_i\} = \{a_1, a_2, \ldots, a_n\}$. q_i is finite and nonempty. Elements belonging to the set $(S-q_i)$ are called <u>foreign</u> with respect to (w.r.t.) q_i. Suppose there exists a 1 - 1 function f that maps the elements of S into points in the real line $\mathbf{R}$ such that for each $q_i \in Q$, there exists an interval I_i containing the images of all elements $\in q_i$ but not images of any foreign elements w.r.t. q_i. Then we say that the family Q possesses the property of linear ordering or Q is linearly orderable. The intervals $I_1, I_2, \ldots, I_m$ may either be open or closed. In the sequel, we shall take consistently the intervals to be closed.

We say that $(f; I_1, I_2, \ldots, I_m)$ implies the linear ordering property of the family Q to mean that the function f and the intervals $I_1, I_2, \ldots, I_m$ are such that $f: \bigcup_{q_i \in Q} \{q_i\} \xrightarrow{1:1} \mathbb{R}$ and $f(a_j) \in I_i$ for $\forall a_j \in q_i$ and $f(a_j) \notin I_i$ for $\forall a_j \notin q_i$. The following assertion is obvious:

Assertion: A family of queries Q is consecutively retrievable iff the family of sets obtained when we consider each query in Q as a set of elements of pertinent records is linearly orderable.

Section 1: Interval Graphs and Linearly Orderable Sets.

In this section we shall show the relation between interval graphs [8] and the intersection graph [8] of a family of sets that is linearly orderable.

Let $Q = \{q_1, q_2, \ldots, q_m\}$ be a family of distinct, non-empty, finite sets. The intersection graph of Q is denoted by $\Omega(Q)$ and is defined as follows: for each set $q_i \in Q$, there exists a corresponding node $q_i \in \Omega(Q)$ and vice versa and for $i \neq j$, q_i is connected with q_j iff $q_i \cap q_j \neq \phi$.

Example 4.1:

Let $Q = \{q_1, q_2, q_3, q_4, q_5\}$

where $q_1 = \{a_1, a_4, a_6, a_7\}$

$q_2 = \{a_1, a_2, a_5\}$

$q_3 = \{a_1, a_6, a_7\}$

$q_4 = \{a_1, a_2, a_3, a_4, a_5\}$

and $q_5 = \{a_2, a_3, a_5\}$

The intersection graph $\Omega(Q)$ of Q is given in <u>figure 4.1</u>. ****

Let G be any undirected graph. If it is possible to assign to each node a_i of G, a distinct interval I_i in the real line such that I_i overlaps with I_j iff nodes a_i and a_j are connected, then G is called an <u>interval graph</u>. The intervals may be open or closed.

<u>Example 4.2</u>

The graph $\Omega(Q)$ in example 4.1 is an interval graph.

Let $I_1 = [0,4]$

$I_2 = [2,7]$

$I_3 = [-2,3]$

$I_4 = [1,8]$

$I_5 = [5,7]$ where I_i, $1 \leq i \leq 5$ corresponds to node q_i. ****

From now onwards let Q denote a family of finite, non-empty, distinct sets $\{q_1, q_2, \ldots, q_m\}$ and S the set $\bigcup_{q_i \in Q} \{q_i\} = \{a_1, a_2, \ldots, a_n\}$.

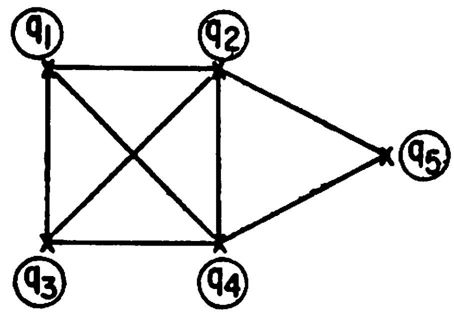

Figure 4.1

Theorem 1:

If Q is linearly orderable, then $\Omega(Q)$ is an interval graph.

Proof: Since Q is linearly orderable, there exists a 1 - 1 function f and intervals $I_1, I_2, \ldots, I_m$ where $I_i = [\min_{a_p \in q_i} (f(a_p)), \max_{a_p \in q_i} (f(a_p)) + \delta_i]$ and for δ_i small enough $(f; I_1, I_2, \ldots, I_m)$ implies the linear ordering property of Q. Note that the increment δ_i is added to I_i to take care of the situation that q_i may be a singleton. Then, I_i overlaps with I_j, $j \neq i$, iff $q_i \cap q_j \neq \phi$. But $q_i \cap q_j \neq \phi$ iff q_i and q_j are connected in $\Omega(Q)$. QED ****

Theorem 2:

If G is an interval graph, then there exists a family of sets Q such that $\Omega(Q) = G$ and Q is linearly orderable. In other words every interval graph is an intersection graph is some family of linearly orderable sets.

Proof: Let $\{q_1, q_2, \ldots, q_m\}$ be the set of nodes of the interval graph G and $I = \{I_1, I_2, \ldots, I_m\}$ be a set of distinct intervals such that interval I_i corresponds to node q_i. We shall assume that the intervals are finite. If I_i is an infinite interval, we can always choose a finite subinterval I_i' of I_i such that I_i' overlaps with I_j iff I_i overlaps with I_j. I_i may be replaced by I_i' in I.

Define i_{min} = Minimum of interval I_i.

i_{max} = Maximum of interval I_i.

We, now, define sets $q_1, q_2, \ldots q_m$.

For $1 \leq i \leq m$, $q_i = \{i_{min}, i_{max}\} \cup \{j_{min} | i_{min} \leq j_{min} \leq i_{max}\}$

$$\cup \{j_{max} | i_{min} \leq j_{max} \leq i_{max}\}$$

Since $\{i_{min}, i_{max}\} \neq \{j_{min}, j_{max}\}$, it follows that $q_i \neq q_j$ for $i \neq j$. If q_i and q_j are connected in G, then intervals I_i and I_j overlap. Then we have $\{i_{min}, i_{max}, j_{min}\} \subseteq q_i$ or $\{i_{min}, i_{max}, j_{max}\} \subseteq q_i$. Since $\{j_{min}, j_{max}\} \subseteq q_j$, $q_i \cap q_j \neq \phi$.

Let $Q = \{q_1, q_2, \ldots q_m\}$. We first observe that G is $\Omega(Q)$. Let $S = \bigcup_{q_i \in Q} \{q_i\}$. Note that the elements of S are either minimum or maximum of some interval $\in I$.

We need to prove that Q has linear orderable property. Define $S_{min} = \underset{a_i \in S}{\text{Min}}(a_i)$ and $S_{max} = \underset{a_i \in S}{\text{Max}}(a_i)$. Let f be the identity function that maps S into points in $[S_{min}, S_{max}]$. We claim that the function f and the set of intervals I imply that Q is linearly orderable. To see this, assume to the contrary that f and I do not imply the linear orderable property of Q. Then, there exists an interval, say I_i, such that I_i contains the image of at least one foreign element, say k, w.r.t. q_i, the set corresponding to I_i. I_i is $[i_{min}, i_{max}]$. Then k is such that $i_{min} \leq k \leq i_{max}$. Since $k \in S$, k is either j_{min} or j_{max} for some interval I_j, $j \neq i$. This implies that I_j overlaps I_i and $k \in q_i$. Then from the definition of q_i, k is not foreign to q_i. Contradiction. <u>QED</u> ****

Example 4.3:

Let $q_1 = \{b, c, g, h, a\}$

$q_2 = \{a, e, d\}$

$q_3 = \{h, a, e, d\}$

$q_4 = \{c, g, h, e, a\}$

and $q_5 = \{e, d\}$

Let $Q = \{q_1, q_2, q_3, q_4, q_5\}$

The intersection graph $\Omega(Q)$ is shown in figure 4.2. $\Omega(Q)$ is an interval graph. Let the intervals corresponding to the nodes be:

$I_1 = [1, 5]$

$I_2 = [5, 7]$

$I_3 = [4, 7]$

$I_4 = [2, 6]$

$I_5 = [6, 7]$. Interval I_i corresponds to node q_i for $1 \leq i \leq 5$. The intervals are represented pictorially in figure 4.3. The node that an interval represents is given in parenthesis in the figure.

Define a function f:

$f(b) = 1$

$f(c) = 2$

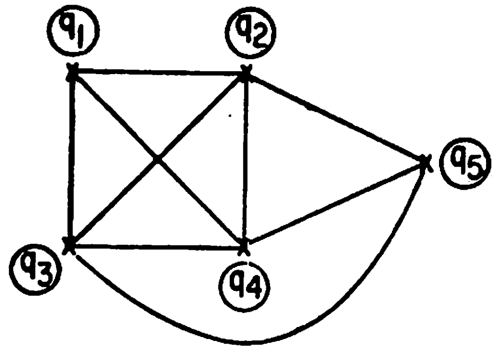

Figure 4.2

$f(g) = 3$

$f(h) = 4$

$f(a) = 5$

$f(e) = 6$ and $f(d) = 7$

The function f and the intervals I_1, I_2, I_3, I_4 and I_5 imply that Q is linearly orderable. The pre-image of i for i = 1, 2, ..., 7 is given in parenthesis next to i in figure 4.3. ****

If G is any graph, then the complement of G, denoted by G^c, is a graph that has the same nodes as G with an edge connecting a pair of distinct nodes in G^c iff that edge does not occur in G [8]. Let G be an undirected graph defined by [V, R], where V is a finite non-empty set of nodes of G and R is an irreflexive relation on V such that for $\forall\ a_i$, $a_j \in V$, $i \neq j$, $a_i\ R\ a_j$ iff a_i is connected with a_j in G. An undirected graph [V, R] is transitive orientable iff there exists a directed graph $\tilde{G} = [V, \tilde{R}]$ such that for $\forall\ a_i, a_j, a_k \in V$, if $a_i\ R\ a_j$ then either $a_i\ \tilde{R}\ a_j$ or $a_j\ \tilde{R}\ a_i$ and $a_i\ \tilde{R}\ a_j$, $a_j\ \tilde{R}\ a_k \Rightarrow a_i\ \tilde{R}\ a_k$. $a_i\ \tilde{R}\ a_j$ iff there is an edge from a_i to a_j in $\tilde{G}$. In other words, it is possible to assign directions to the edges of G such that G is transitive [10]. A transitive orientable graph is sometimes called a comparable graph [9]. The following theorem is due to Gilmore and Hoffman and is given here for the sake of completeness. The proof may be found in [9].

Theorem 3:

A graph G is an interval graph iff every quadrilateral in G

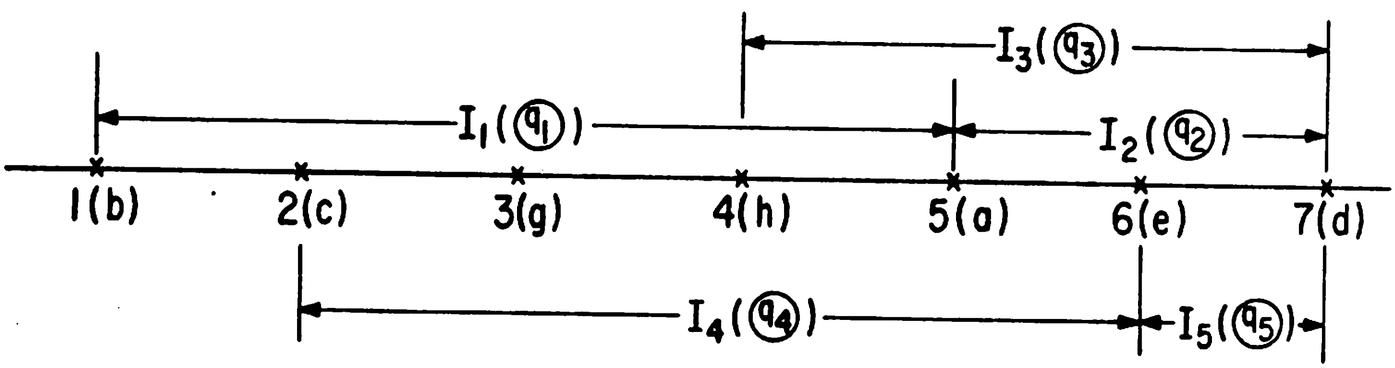

Figure 4.3

has a diagonal and G^c is transitive orientable.

Pnueli et al. [10] give an algorithm to check if a graph is transitive orientable.

Example 4.4:

Consider the graph in figure 4.2. We saw that it was an interval graph. The quadrilaterals (q_1, q_2, q_3, q_4), (q_1, q_2, q_3, q_5), (q_1, q_2, q_4, q_5), (q_1, q_3, q_4, q_5), (q_2, q_3, q_4, q_5) have at least one diagonal and the complement of the graph is transitive orientable. See figure 4.4. ****

Section 2: Singleton Sets in a Family of Sets.

In this section we show that the singleton sets in a family of sets do not influence the linear ordering (L.O.) property of the family. The proof of the following two lemmas are obvious.

Lemma 1:

If Q is linearly orderable, then $Q' \subseteq Q$ is linearly orderable.

Proof:

Since Q is linearly orderable, there exists a function f and intervals $I_1, I_2, \ldots I_m$ implying the L.O. property of Q.

Let $S' = \bigcup_{q_j \in Q'} \{q_j\}$ where $Q' \subseteq Q$.

= set of elements belonging to the sets in the subfamily Q'.

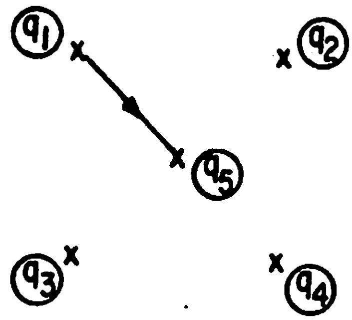

Figure 4.4

Now, define f': $f'(a_i) = f(a_i)$ $\forall a_i \in S'$

and $I_i' = I_i$ for $\forall q_i \in Q'$

f' and $\{I_i' | q_i \in Q'\}$ imply the L.O. property of Q'. QED ****

Lemma 2:

Let $Q = \{q_1, q_2, \ldots q_m\}$, $S = \bigcup_{q_j \in Q} \{q_j\} = \{a_1, a_2, \ldots a_n\}$

and $\bar{q}_j = \{a_j\}$ for $1 \leq j \leq n$. Then Q is linearly orderable iff $\bar{Q}$ is linearly orderable where $\bar{Q} = Q \cup \{\bar{q}_i\}$, $i \in \{1, 2, \ldots n\}$.

Proof: If Q is linearly orderable then there exists a function f and intervals $I_1, I_2, \ldots I_m$ such that they satisfy the linearly orderable property of Q. Let $\bar{I}_i = [f(a_i), f(a_i) + \delta_i]$ for $\forall \bar{q}_i \in \bar{Q}$. For δ_i small enough, $\bar{I}_i$ does not contain images of any elements other than a_i. Then f and $\{I_i\} \cup \{\bar{I}_i | \bar{q}_i \in \bar{Q}\}$ imply that $\bar{Q}$ is linearly orderable.

$\Leftarrow$. If $\bar{Q}$ is linearly orderable, then by Lemma 1, $Q \subseteq \bar{Q}$ is linearly orderable. QED ****

We can, therefore, assume that as far as linear ordering is concerned, no set in Q is a singleton.

Section 3: Directed Semantic Graphs and Linearly Orderable Sets.

In this section, we derive a number of results regarding the L.O. property of Q when $\Omega(Q)$ is a complete graph. We establish necessary and sufficient conditions for Q to be linearly orderable when $\Omega(Q)$ is complete.

A graph G is <u>complete</u> iff every pair of distinct nodes of G is joined by an edge in G; i.e. no more edge can be added to G [8].

<u>Lemma 3:</u> If $\Omega(Q)$ is complete and Q is linearly orderable, then

$$I = \bigcap_{q_i \in Q} q_i \neq \phi.$$

<u>Proof:</u> Let $(f;I_1,I_2,\ldots,I_m)$ imply the linear ordering property of Q. Since $\Omega(Q)$ is complete, $q_i \cap q_j \neq \phi$ for $1 \leq i, j \leq m$. This implies that $I_i \cap I_j \neq \phi$ and there exists an $a_{ij} \in S$ such that $f(a_{ij}) \in (I_i \cap I_j)$ for $i, j = 1, 2, \ldots, m$.

We shall assume that all the intervals are finite (see proof of theorem 2). Let $I_i = [i_{min}, i_{max}]$ for $1 \leq i \leq m$. Let I_p be such that $p_{max} = \underset{1 \leq j \leq m}{\mathrm{Min}} [j_{max}]$ and I_k be such that

$k_{min} = \underset{1 \leq j \leq m}{\mathrm{Max}} [j_{min}]$. We first observe that $k_{min} \leq p_{max}$.

For, if $k_{min} > p_{max}$, intervals I_p and I_k would not overlap which would be a contradiction.

Since $i_{min} \leq k_{min}$ and $i_{max} \geq p_{max}$ for $\forall\, i = 1, 2, \ldots m$, all intervals contain the subinterval $[k_{min}, p_{max}]$ which is $I_p \cap I_k$ (see <u>figure 4.5</u>). But, we know that the intersection of every pair of intervals contains the image of at least one element $\in S$. Then there exists an $a_{pk} \in S$ such that

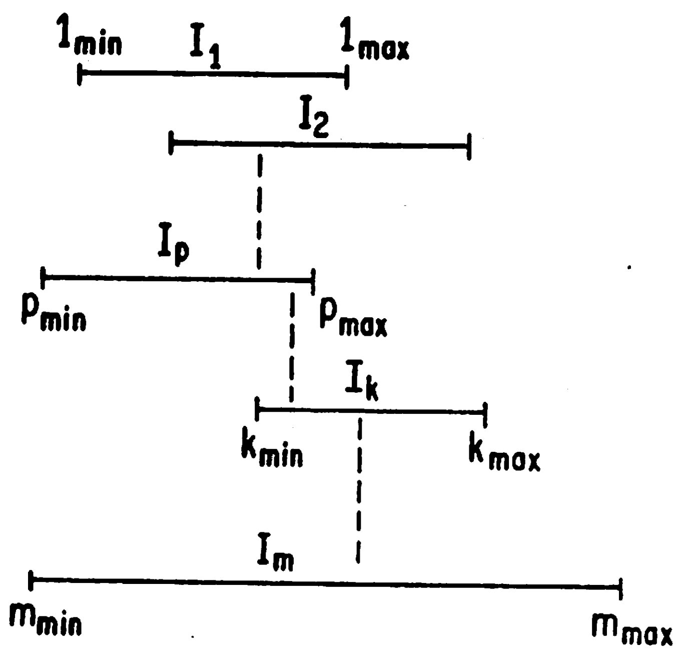

Figure 4.5

$f(a_{pk}) \in I_p \cap I_k \Rightarrow f(a_{pk})$ belongs to all intervals $I_1, I_2, \ldots, I_m$. Since $(f; I_1, I_2, \ldots, I_m)$ imply the linearly orderable property of Q, a_{pk} is not foreign to any set $\in$ Q, i.e. $\bigcap_{q_i \in Q} q_i \neq \phi$. QED ****

Define a directed semantic graph $\bar{G} = [V, R, I]$. V is a finite non-empty set of nodes. R is an irreflexive relation on V such that $\forall\, a_i, a_j \in V$, $i \neq j$, $a_i \, R \, a_j \Leftrightarrow$ there is an edge from a_i to a_j in $\bar{G}$. R is the connectivity relation of $\bar{G}$. I is a subset of V. Nodes in I are called direction-changer nodes and are denoted by an * mark in $\bar{G}$. Nodes in (V-I) are non-direction-changer nodes. $\{a_i, a_j\}$ denotes the edge between a_i and a_j, ignoring the direction on the edge. (a_i, a_j) denotes the directed edge from node a_i to node a_j. A path in a directed semantic graph (DSG) $\bar{G}$ is a sequence of distinct nodes $a_0, a_1, \ldots, a_i, a_{i+1}, \ldots a_k$ of $\bar{G}$ such that for $0 \leq i < k$, (a_i, a_{i+1}) is an edge of $\bar{G}$ when in direct mode and (a_{i+1}, a_i) is an edge of $\bar{G}$ when in reverse mode, where the modes are defined as follows: If a path starts with a non-direction-changer node, then the mode is direct. If it starts with a direction-changer-node, the mode is reverse. Whenever a direction-changer node is reached from a non-direction-changer node, the mode is switched. (If a direction-changer node is reached from a direction-changer node, no change of mode occurs.) We shall enclose the sequence of nodes defining a path in angle brackets, $\langle$ and $\rangle$. If $P = \langle a_0, a_1, \ldots, a_k \rangle$ is a path of $\bar{G}$, then a_0 is called the starting node of P, a_k the end node of P and $a_1, a_2, \ldots a_{k-1}$ the intermediate nodes of P. Note that if $I = \phi$, then our definition of a path is the same as the usual definition of a directed-path in a directed graph. Because of the presence of direction changer nodes in $\overline{G}$, there is some semantics in the definitions regarding $\overline{G}$. Hence the name directed semantic graph.

Example 4.5:

Consider the DSG, $\overline{G}$, shown in figure 4.6.

We have $V = \{a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$

$I = \{a_2, a_5\}$

and R is the connectivity relation. $\langle a_0, a_1, a_2 \rangle$, $\langle a_1, a_2, a_3 \rangle$, $\langle a_4, a_3, a_2 \rangle$, $\langle a_2, a_3, a_4, a_5, a_6 \rangle$, $\langle a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7 \rangle$, $\langle a_1, a_2, a_3, a_4, a_5, a_0 \rangle$ are some of the paths in $\overline{G}$. ****

A Hamiltonian path in a directed semantic graph $\overline{G}$ is a path that passes through all the nodes of $\overline{G}$.

Example 4.6:

Consider the graph $\overline{G}$ in figure 4.6. $\overline{G}$ has only one Hamiltonian path which is $\langle a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7 \rangle$. ****

We now define the DSG of a family of sets Q. Let $I = \bigcap_{q \; q_i \in Q} q_i$.

Let $\overline{R}$ be an irreflective relation defined on S as follows: $a_i \overline{R} a_j$ iff $i \neq j$ and for $\forall q_k \in Q$, $a_i \in q_k \Rightarrow a_j \in q_k$. Note that $\overline{R}$ is transitive. The DSG of Q is denoted by $\overline{G}(Q)$ and is $[S', \overline{R}, I']$. S' is the set of nodes of $\overline{G}(Q)$ and is $\{a_1, a_2, \ldots, a_i, \ldots, a_n\}$ where node a_i corresponds to element $a_i \in S$ and vice versa. $a_i \overline{R} a_j$ iff $a_i \overline{R} a_j$. We use the same symbol $\overline{R}$ for a relation between two elements of S and for the connectivity relation of $\overline{G}(Q)$ since there is no confusion. I' is the set of direction-

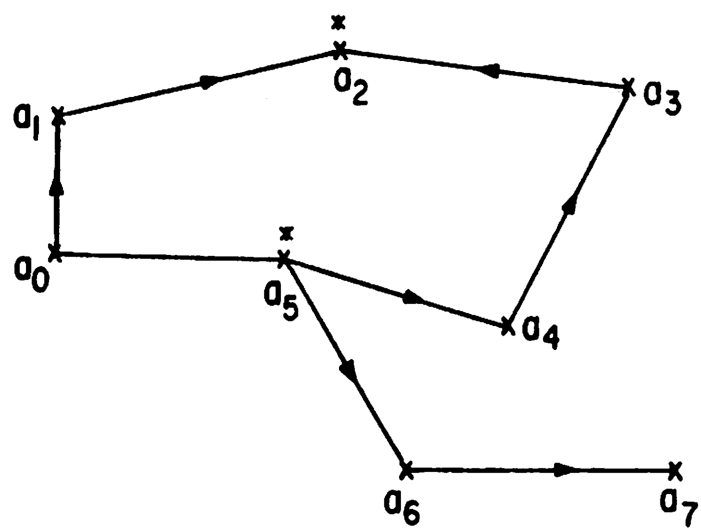

Figure 4.6

changer nodes of $\bar{G}(Q)$ with $a_p \in I'$ iff $a_p \in I$.

Example 4.7

Let $q_1 = \{a_2, a_3\}$

$q_2 = \{a_1, a_2, a_3\}$

$q_3 = \{a_2, a_3, a_4\}$

and $q_4 = \{a_3, a_4, a_5\}$.

Let $Q = \{q_1, q_2, q_3, q_4\}$. Then, $S = \{a_1, a_2, a_3, a_4, a_5\}$

$$I = \bigcap_{q_i \in Q} q_i = \{a_3\}$$

$\bar{R}$: $a_1 \bar{R} a_2$, $a_1 \bar{R} a_3$ i.e. $q_k \supseteq \{a_1\} \Rightarrow q_k \supseteq \{a_2\}$ for $\forall q_k \in Q$.

$a_2 \bar{R} a_3$

$a_4 \bar{R} a_3$

$a_5 \bar{R} a_4$, $a_5 \bar{R} a_3$

$\bar{G}(Q) = [S', \bar{R}, I']$ where

$S' = \{a_1, a_2, a_3, a_4, a_5\}$ and $I' = \{a_3\}$.

$\bar{R}$ is the connectivity relation. $\bar{G}(Q)$ is given in figure 4.7. $\langle a_1, a_2, a_3, a_4, a_5 \rangle$, $\langle a_5, a_4, a_3, a_2, a_1 \rangle$ are Hamiltonian paths in $\bar{G}(Q)$. ****

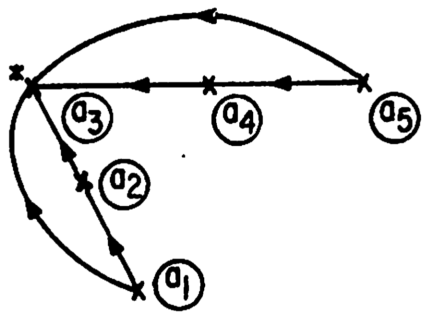

Figure 4.7

Lemma 4: Let $\bar{G}(Q)$ be the DSG of Q and h be any Hamiltonian path in $\bar{G}(Q)$. Then, there does not exist a subpath h' of h such that the starting and end nodes of h' are direction-changer nodes and the intermediate nodes are non-direction-changer nodes.

Proof: Let $I = \bigcap_{q_i \in Q} q_i$. Assume to the contrary that there exists a subpath h' of h = $\langle a_i, a_{i+1}, \ldots, a_{j-1}, a_j \rangle$ where a_i and a_j are direction-changer- nodes and $a_{i+1}, a_{i+2}, \ldots, a_{j-1}$ are not. Since $a_i, a_j \in I$ and $a_{i+1}, a_{j-1} \in (S-I)$, we have that (a_{i+1}, a_i) and (a_{j-1}, a_j) are edges of $\bar{G}(Q)$ and (a_i, a_{i+1}) and (a_j, a_{j-1}) are not edges of $\bar{G}(Q)$ (see figure 4.8).

In the subpath h', since there are no direction-changer nodes between a_i and a_j we should have either (i) edges (a_i, a_{i+1}) and (a_{j-1}, a_j) or (ii) edges (a_{i+1}, a_i) and (a_j, a_{j-1}). In either case, we have a situation that contradicts the earlier statement that (a_i, a_{i+1}) and (a_j, a_{j-1}) are not edges of $\bar{G}(Q)$. QED ****

Corollary 1: Any Hamiltonian path in the DSG of Q should have a subpath of the form $\langle a_i, a_{i+1}, \ldots, a_{j-1}, a_j \rangle$ where $\{a_i, a_{i+1}, \ldots, a_{j+1}, a_j\} = I = \bigcap_{q_i \in Q} q_i$.

Lemma 5: If there exists a Hamiltonian path in $\bar{G}(Q)$, then $\bigcap_{q_i \in Q} q_i \neq \phi$, i.e. there exists at least one direction changer node in $\bar{G}(Q)$.

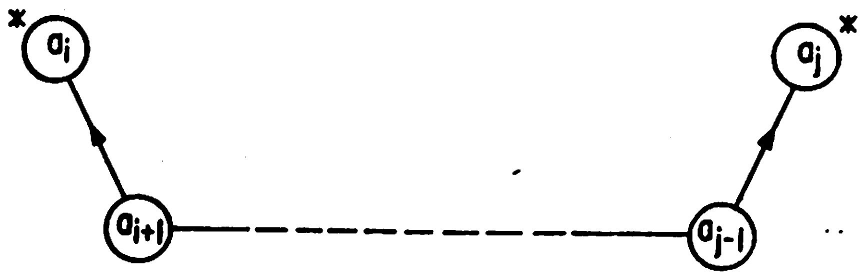

Figure 4.8

Proof: Suppose to the contrary that the set of direction-changer nodes in $\bar{G}(Q) = \phi$. Let $h = (a_0, a_1, \ldots, a_n)$ be a Hamiltonian path of $\bar{G}(Q)$. Then $(a_0, a_1), (a_1, a_2), \ldots (a_{n-1}, a_n)$ are among the edges of $\bar{G}(Q)$. Relation $\bar{R}$ is transitive. Then for $0 \leq p \leq n-1$, (a_p, a_n) is an edge of $\bar{G}(Q)$. This means that $a_n \in q_j$ for $\forall q_j \in Q \Rightarrow \bigcap_{q_j \in Q} q_j \neq \phi$. Contradiction. QED ****

The following theorem gives another necessary condition for the existence of a Hamiltonian path in $\bar{G}(Q)$.

Theorem 4: Let there exist a Hamiltonian path in $\bar{G}(Q)$. If $S_1 \subseteq S$ is a set of incomparable elements (w.r.t. $\bar{R}$), then $|S_1|^{\dagger} \leq 2$.

Proof: By contradiction. Suppose there exists a set $S_1 = \{a_i, a_j, a_k\}$ of incomparable elements and $S_1 \subseteq S$. $S_1' = \{a_i, a_j, a_k\}$ is the set of nodes of $\bar{G}(Q)$ that correspond to S_1. Let h be a Hamiltonian path of $\bar{G}(Q)$.

Let $I = \bigcap_{q_i \in Q} q_i$. Since all the elements of S are $\bar{R}$ - related with the elements $\in I$, we have $S_1 \cap I = \phi$, i.e. none of the nodes $\in S_1'$ is a direction-changer node. Without loss of generality, we shall assume that in the Hamiltonian path h of $\bar{G}(Q)$, a_i precedes a_j and a_j precedes a_k.

By Lemma 4, in any Hamiltonian path non-direction-changer nodes are not present between any two direction-changer nodes. We then have only the following cases for h:

Case (i) h passes through all the direction-changer nodes after leaving

---

$^{\dagger}|S_1|$ denotes the cardinality of the set S_1.

a_j. Then (a_i, a_{i+1}), (a_{i+1}, a_{i+2}), ..., (a_{j-1}, a_j) are all edges of $\bar{G}(Q)$. $\bar{R}$ is transitive. Thus we have that (a_i, a_j) is an edge of $\bar{G}(Q)$. But this is not possible since a_i and a_j are not comparable w.r.t. $\bar{R}$.

__Case (ii)__ h passes through all the direction-changer nodes before reaching a_i. After an argument similar to that of case (i), we get (a_j, a_i) which again contradicts the incomparability of a_i and a_j.

__Case (iii)__ h visits all the direction-changer nodes between a_i and a_j. After visiting a_j, h needs to visit a_k and this is the same as case (ii) which leads to a contradiction.

By cases (i), (ii) and (iii), we see that h can not visit all a_i, a_j and a_k. Then h is not a Hamiltonian path which is a contradiction. __QED.__ ****

__Corollary 2__: For all a_i, $a_j \in S$, $i \neq j$, a_i and a_j are incomparable (w.r.t. $\bar{R}$) only if in any Hamiltonian path of $\bar{G}(Q)$, a_i and a_j exist on the opposite sides of the subpath which passes through all the direction-changer nodes. ****

The following lemma leads us to the connection between the linear ordering property of a family Q and the existence of a Hamiltonian path in $\bar{G}(Q)$, when $\Omega(Q)$ is complete.

__Lemma 6__: Let h = (a_1, ..., a_i, a_{i+1}, ..., a_{j-1}, a_j, ..., a_n) be a Hamiltonian path of $\bar{G}(Q)$. If $\{a_i, a_j\} \subseteq q_p \in Q$ then $\{a_i, a_{i+1}, a_{i+2}, \ldots a_{j-1}, a_j\} \subseteq q_p$.

Proof: Let $I = \bigcap_{q_i \in Q} q_i$. We have three situations.

(1) Both a_i and a_j are direction changer nodes. By Lemma 4, all the nodes between a_i and a_j are also direction-changer nodes. Then the elements corresponding to a_{i+1}, a_{i+2}, ..., a_{j-1} belong to I. Hence the lemma.

(2) Both a_i and a_j are not direction-changer nodes. For this situation, we have the following possible cases similar to the ones we had in the proof of theorem 4.

Case (i): h visits all the direction-changer nodes after leaving a_j. Then (a_i, a_{i+1}), (a_{i+1}, a_{i+2}), ..., (a_{j-1}, a_j) are edges of $\overline{G}(Q)$. Since $(a_\ell, a_m) \Rightarrow (a_\ell \in q_k \Rightarrow a_m \in q_k$, for $\forall q_k$ in Q), we have $a_i, a_{i+1}, \ldots, a_{j-1}, a_j \in q_p$.

Case (ii): Let h visit all the direction-changer nodes before reaching a_i. A similar argument as in case (i) leads us to the conclusion that $a_j, a_{j-1}, \ldots, a_{i-1}, a_i \in q_p$ when $a_i, a_j \in q_p$.

Case (iii): The direction-changer nodes are between a_i and a_j in h. Let a_ℓ and $a_{\ell+k}$ be the starting and end nodes of the subpath of h that consists only of the direction-changer nodes (by Lemma 4). Then the path h is $\langle a_1, \ldots, a_i, \ldots, \underbrace{a_\ell, \ldots, a_{\ell+k}}_{\text{direction-changers}}, \ldots, a_j \rangle$

By case (i), all the elements that correspond to nodes between a_i and $a_{\ell-1}$ in h belong to q_p and by case (ii), all the elements corresponding to nodes between $a_{\ell+k+1}$ and a_j in h belong to q_p. The direction-changer nodes correspond to the elements of I which is a subset of all sets in Q. Hence we have the lemma.

(3) Either a_i or a_j is a direction-changer node. Suppose a_i is. Let a_{i+k} be the end node of the subpath of h that consists only of the direction-changer nodes. Then the path h is $\langle a_1, \ldots, a_i, \ldots, a_{i+k}, a_{i+k+1}, \ldots, a_{j-1}, a_j \rangle$. By case (ii) of (2), $\{a_j, a_{j-1}, \ldots, a_{i+k+1}\} \subseteq q_p$. We know that $\{a_i, a_{i+1}, \ldots, a_{i+k}\} \subseteq I \subseteq q_p$. Hence, $\{a_i, a_{i+1}, \ldots, a_{j-1}, a_j\} \subseteq q_p$.

If a_j is a direction-changer node and a_i is not, a similar argument as above can be applied and the lemma proved. QED.

Let us now state and prove the counterpart of Lemma 4.

Lemma 7: Let $\Omega(Q)$ be complete and Q linearly orderable. Let $I = \bigcap_{q_i \in Q} q_i$ and $(f; I_1, I_2, \ldots, I_m)$ imply the linearly orderable property of Q. Then there does not exist a_b, a_c, a_d such that a_b, $a_d \in I$ and $a_c \in (S-I)$ and $f(a_c)$ is between $f(a_b)$ and $f(a_d)$.

Proof: Assume to the contrary that there exist such a_b, a_c and a_d. Since $a_c \notin I$, there exists a set $q_i \in Q$ such that $a_c \notin q_i$. Since $\{a_b, a_c\} \subseteq I$, the interval I_i corresponding to q_i contains $f(a_b)$ and $f(a_d)$. If $f(a_c)$ is between $f(a_b)$ and $f(a_d)$, then I_i also contains $f(a_c)$. But a_c is foreign to q_i. This means that $(f; I_1, I_2, \ldots, I_m)$ does not imply the linearly orderable property of Q. Contradiction.

QED ****

The theorem that follows gives the necessary and sufficient conditions for a family of sets Q whose intersection graph is complete to be linearly orderable.

Theorem 5: Let $\Omega(Q)$ be complete. Q is linearly orderable iff there exists a Hamiltonian path in $\bar{G}(Q)$

Proof: The sufficiency part of the theorem is easy to prove. We have $Q = \{q_1, q_2, \ldots q_m\}$ and $S = \{a_1, a_2, \ldots a_n\}$. Let h be a Hamiltonian path of $\bar{G}(Q)$. We can consider h as a n-tuple. Define a set of functions, $\{k_1, k_2, \ldots, k_n\}$, where k_i, $1 \leq i \leq n$, maps any n-tuple to the i^{th} member of the tuple, i.e. $k_i\ (\langle x_1, x_2, \ldots, x_i, \ldots, x_n\rangle) = x_i$.

Corresponding to Hamiltonian path h, let f_h be a 1-1 function that maps S into R such that for $\forall a_i \in S$, $f_h(a_i) = j$ where $k_j(h) = a_i$. (Note that f_h maps elements of S onto integers from 1 to n.)

Now, for $\forall q_i \in Q$, define $I_i = [\ \min\limits_{a_p \in q_i} (f_h(a_p)),\ \max\limits_{a_p \in q_i} (f_h(a_p))]$

It can be observed that I_i contains the images of all elements $\in q_i$. Further, for $1 \leq i \leq m$, I_i does not contain images of foreign elements w.r.t. q_i. To see this, suppose to the contrary that there exists an interval I_i containing images of foreign element(s) w.r.t. q_i. Then there exist a_b, a_{c_1}, a_{c_2}, ..., a_{c_k}, ..., a_{c_j}, a_d belonging to S with $a_{c_k} \notin q_i$ and $f_h(a_{c_k})$ between $f_h(a_b)$ and $f_h(a_d)$ and $\{a_b, a_d\} \subseteq q_i$. This by the definition of f_h implies that a_{c_k} is between a_b and a_d which contradicts Lemma 6. Hence $(f_h; I_1, I_2, \ldots, I_m)$ imply the linearly orderable property of Q.

Now the necessity part of Theorem 5. By Lemma 3, we know that

$I = \bigcap_{q_i \in Q} q_i \neq \phi$. Let $I = \{a_p, a_{p+1}, \ldots, a_{p+\ell}\}$. There exist a function f_h and a set of intervals $\{I_1, I_2, \ldots, I_m\}$ such that $(f_h; I_1, I_2, \ldots, I_m)$ imply the linearly orderable property of Q. For all $a_i, a_j \in S$, $i \neq j$, either $f_h(a_i) < f_h(a_j)$ or $f_h(a_j) < f_h(a_i)$. Then we can define a total ordering on the elements of S such that a_i precedes a_j in the total ordering iff $f_h(a_i) < f_h(a_j)$. By Lemma 7, there does not exist a_b, a_c, a_d such that $a_b, a_d \in I$ and $a_c \in (S-I)$ and $f_h(a_c)$ is between $f_h(a_b)$ and $f_h(a_d)$. Without loss of generality we can then assume that f_h is such that

$$f_h(a_1) < f_h(a_2) < \ldots < \underbrace{f_h(a_p) < f_h(a_{p+1}) < \ldots < f_h(a_{p+\ell})}_{\text{images of elements} \in I} < \ldots < f_h(a_n)$$

All the intervals contain the images of elements belonging to I. Hence, whenever an interval I_i contains $f_h(a_1)$ it has to contain $f_h(a_2)$, $f_h(a_3), \ldots, f_h(a_{p+\ell})$. Then, $a_1 \in q_i \Rightarrow \{a_2, a_3, \ldots a_p, \ldots a_{p+\ell}\} \subseteq q_i$. For, if any of the elements $\in \{a_2, a_3, \ldots, a_{p-1}\}$ is foreign to q_i, we will have a contradiction that $(f_h; I_1, I_2, \ldots, I_m)$ does not establish the L. O. property of Q.

Thus we have $a_1 \bar{R} a_2$, $a_1 \bar{R} a_3$, $\ldots$ $a_1 \bar{R} a_{p+\ell}$. In particular, (a_1, a_2) is an edge of $\bar{G}(Q)$. By considering intervals that contain $f_h(a_2), f_h(a_3), \ldots f_h(a_{p-1})$ and repeating the same argument as above, we see that $(a_2, a_3), (a_3, a_4), \ldots, (a_{p-1}, a_p)$ are among the edges of $\bar{G}(Q)$.

A similar argument as above shows that $(a_n, a_{n-1}), (a_{n-1}, a_{n-2}), \ldots, (a_{p+\ell+1}, a_{p+\ell})$ are also edges of $\overline{G}(Q)$. Since the relation $\overline{R}$ is symmetric for I, every pair of nodes belonging to $I' = \{a_p, a_{p+1}, \ldots, a_{p+\ell}\}$ is connected and directed both ways. But the nodes $\in I'$ are precisely the direction-changer nodes of $\overline{G}(Q)$. Hence $\langle a_1, a_2, \ldots, a_{p-1}, a_p, \ldots, a_{p+\ell}, a_{p+\ell+1}, \ldots, a_n\rangle$ is a path of $\overline{G}(Q)$ which is Hamiltonian. QED. ****

Example 4.8:

Let $q_1 = \{a_2, a_3, a_4, a_6\}$

$q_2 = \{a_1, a_2, a_3, a_6, a_5\}$

$q_3 = \{a_1, a_3, a_6\}$

and $q_4 = \{a_1, a_2, a_3, a_4, a_6\}$

Define $Q = \{q_1, q_2, q_3, q_4\}$

We have $S = \{a_1, a_2, a_3, a_4, a_5, a_6\}$

and $I = \bigcap_{q_i \in Q} q_i = \{a_3, a_6\}$

The intersection graph of Q is given in figure 4.9. We note that $\Omega(Q)$ is complete. The DSG of Q is $\overline{G}(Q) = [S', \overline{R}, I']$ where $S' = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ $I' = \{a_3, a_6\}$ and $\overline{R}$ is the connectivity relation. $\overline{G}(Q)$ is given in figure 4.10. $h = \langle a_4, a_2, a_3, a_6, a_1, a_5\rangle$ is a Hamiltonian path of $\overline{G}(Q)$ and is shown in solid lines in figure 4.10.

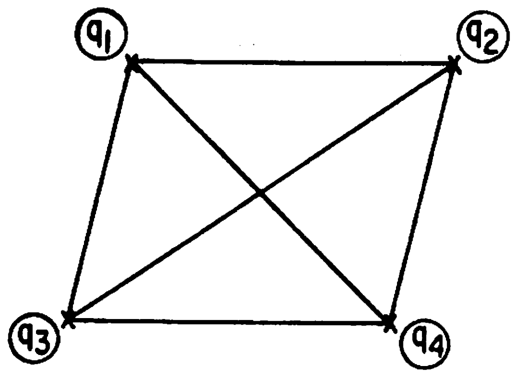

Figure 4.9

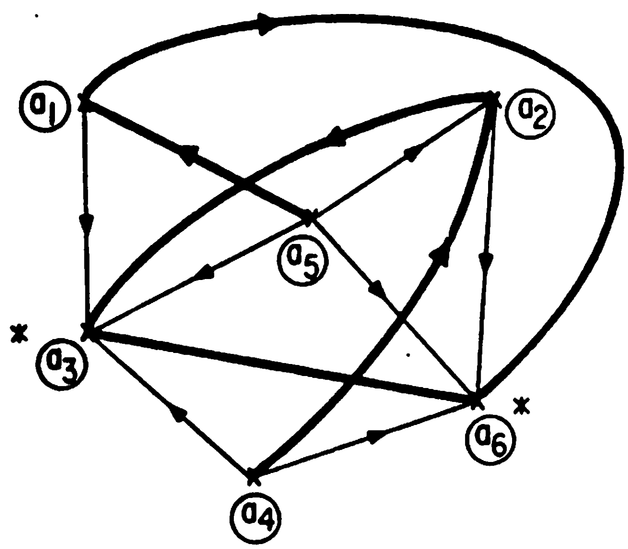

Figure 4.10

Define f_h: $f_h(a_4) = 1$ as $k_1(h) = a_4$

$f_h(a_2) = 2$ as $k_2(h) = a_2$

$f_h(a_3) = 3$ as $k_3(h) = a_3$

$f_h(a_6) = 4$ as $k_4(h) = a_6$

$f_h(a_1) = 5$ as $k_5(h) = a_1$

and $f_h(a_5) = 6$ as $k_6(h) = a_5$

We then define the intervals I_1, I_2, I_3 and I_4.

$$I_1 = [\min_{a_i \in q_1} (f_h(a_i)), \max_{a_i \in q_1} (f_h(a_i))]$$

$$= [f_h(a_4), f_h(a_6)] = [1, 4]$$

Similarly $I_2 = [2, 6]$, $I_3 = [3, 5]$ and $I_4 = [1, 5]$. The intervals are shown pictorially in <u>figure 4.11</u>. ****

Let $P = (a_1, a_2, \ldots, a_i, a_{i+1}, \ldots, a_k)$ be a path in $\overline{G}(Q)$. We say that each a_i in P, for $1 < i < k$, has both <u>left</u> and <u>right</u> <u>neighbors</u>. The left and right neighbor of a_i are a_{i-1} and a_{i+1} respectively. a_1 has only the right neighbor namely a_2 and a_k has only the left neighbor which is a_{k-1}. The left neighbor of a_1 is said to be <u>empty</u> and so is the right neighbor of a_k.

Two paths P_1 and P_2 are <u>equal</u> (or non-distinct) iff the starting and end nodes of P_1 are the starting and end nodes of P_2 and for $\forall\, a_i \in P_1$ such that a_i is not the starting node of P_1, the left neighbor of a_i in

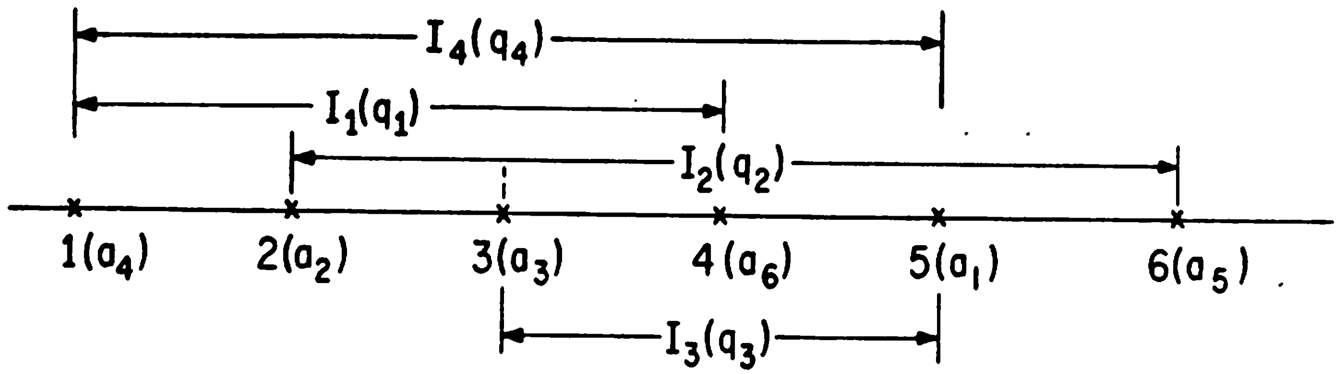

Figure 4.11

P_1 = the left neighbor of a_i in P_2 and for $\forall$ $a_i \in P_1$ such that a_i is not the end node of P_1, the right neighbor of a_i in P_1 = the right neighbor of a_i in P_2.

Let f be a 1-1 function that maps S into R. We know that f totally (linearly) orders the elements of S such that $\forall$ a_i, $a_j \in S$, $i \neq j$, a_i procedes a_j iff $f(a_i) < f(a_j)$. Let the linear ordering defined by f be 0. If there exist intervals I_1, I_2, ..., I_m such that $(f;\ I_2, \ldots, I_m)$ implies that Q has the L.O. property, then we say that the linear ordering 0 implies the L.O. property of Q. The following assertion is stronger than theorem 5, but the proof is essentially the same.

__Theorem 6__: Every distinct linear ordering of the elements $\in$ S, which implies the L.O. property of Q, corresponds to a distinct Hamiltonian path in $\overline{G}(Q)$ and vice versa when $\Omega(Q)$ is complete.

__Proof__: In the proof of the necessity part of theorem 5, we observe that the function f_h defines a linear ordering, say 0, of the elements of S. We found a Hamiltonian path in $\overline{G}(Q)$ that corresponded to 0.

Let $(f_{h'};\ I_1', I_2', \ldots, I_m')$ satisfy the L.O. property of Q and $f_{h'}$ be different from f_h. $f_{h'}$ then gives a total ordering 0' different from 0. Applying the same arguments as in theorem 5, we get a Hamiltonian path h' corresponding to 0'. h' is different from h.

In the proof of the only-if part of theorem 5, we defined a function f_h and intervals I_1, I_2, ..., I_m corresponding to a Hamiltonian path h of $\overline{G}(Q)$ such that $(f_h;\ I_1, I_2, \ldots, I_m)$ implied the L.O. property of Q. Any other Hamiltonian path h' would have resulted in a function

$f_{h'}$, $f_{h'} \neq f_h$, and a set of intervals $I'_1, I'_2, \ldots, I'_m$ such that $(f_{h'}; I'_1, I'_2, \ldots, I'_m)$ implied the L.O. property of Q. Since f_h and $f_{h'}$ are distinct, the linear orderings defined by them are distinct. QED. ****

Lemma 8: If $h = (a_1, a_2, \ldots, a_i, a_{i+1}, \ldots, a_n)$ is a Hamiltonian path in $\bar{G}(Q)$, then $h^R = (a_n, a_{n-1}, \ldots, a_{i+1}, a_i, \ldots, a_1)$ is also a Hamiltonian path of $\bar{G}(Q)$.

Proof: Since there exists a Hamiltonian path in $\bar{G}(Q)$, by Lemma 5 $I = \bigcap_{q_i \in Q} q_i \neq \phi$. By the direction changing property of the nodes corresponding to I and by Lemma 4, we have that h^R is a Hamiltonian path of $\bar{G}(Q)$. ****

Corollary 3: Let $\Omega(Q)$ be complete. Then Q is linearly orderable iff there exist at least two Hamiltonian paths in $\bar{G}(Q)$.

Section 4: Union of Two Linearly Orderable Families Whose Intersection Graphs are Complete.

In the sequel, Q_1 and Q_2 denote two families of sets. $Q_1 \cap Q_2$ need not be empty. $\bar{G}(Q_1)$ and $\bar{G}(Q_2)$ represent the DSG of Q_1 and Q_2 respectively. S_1 denotes the set $\bigcup_{q_i \in Q_1} \{q_i\}$ and S_2 the set $\bigcup_{q_i \in Q_2} \{q_i\}$. $\tilde{S}$ indicates $(S_1 \cup S_2)$.

Lemma 9: Let $Q_1 \cup Q_2$ be a family of linearly orderable sets and $\Omega(Q_1)$, $\Omega(Q_2)$ be complete. Let $I = S_1 \cap S_2$ and f be a function that defines a linear ordering of the elements $\in \tilde{S}$ such that $(Q_1 \cup Q_2)$ has L.O. property. Then there does not exist a_p, a_i,

a_j such that $a_p \in (\tilde{S}-I)$ and $\{a_i, a_j\} \subseteq I$ and $f(a_p)$ is between $f(a_i)$ and $f(a_j)$.

Proof: By contradiction. Suppose that there exist such a_p, a_i and a_j. Without loss of generality, we shall assume that $f(a_i) < f(a_p) < f(a_j)$.

Since $(Q_1 \cup Q_2)$ is linearly orderable, by Lemma 1 Q_1 and Q_2 are linearly orderable. Then by Lemma 3, $I_1 = \bigcap_{q_i \in Q_1} q_i \neq \phi$ and $I_2 = \bigcap_{q_i \in Q_2} q_i \neq \phi$.

As $a_i, a_j \in S_1$, there exist sets $q_c, q_d \in Q_1$ such that $a_i \in q_c$ and $a_j \in q_d$. $I_1 \subseteq q_c$ and $I_1 \subseteq q_d$.

Case (i) $a_i \in I_1$. Then the interval I_d^1 corresponding to q_d contains $f(a_i)$, $f(a_j)$ and hence $f(a_p)$. Since the linear ordering defined by f linearly orders sets in $Q_1 \cup Q_2$, a_p is not foreign to q_d.

Case (ii) $a_j \in I_1$. By the same arguments as in case (i), we have $a_p \in q_c$.

Case (iii) If $a_i, a_j \notin I_1$ then there exists an element $a_\ell \in I_1$ such that either $f(a_\ell) < f(a_i)$ or $f(a_i) < f(a_\ell)$. In either case, $a_p \in q_c$ or q_d. For, if a_p is foreign to both q_c and q_d, we will have a contradiction that f does not linear order $(Q_1 \cup Q_2)$.

By cases (i), (ii) and (iii), we see that there exists a $q_i \in Q_1$ such that $a_p \in q_i$. Thus, $a_p \in S_1$. By similar arguments, we can prove that $a_p \in S_2$. This means that $a_p \in S_1 \cap S_2 = I$. Contradiction. QED. ****

Lemma 10: Let $Q_1 \cup Q_2$ be a family of linearly orderable sets and $\Omega(Q_1)$, $\Omega(Q_2)$ be complete. Let $I = S_1 \cap S_2 \neq \phi$, S_1 or S_2. Let f define a linear ordering of the elements $\in \tilde{S}$ implying the L.O.

property of the sets in $(Q_1 \cup Q_2)$. Let a_c and a_k be such that $f(a_c) = \min_{a_i \in I} f(a_i)$ and $f(a_k) = \max_{a_i \in I} f(a_i)$. Then

(i) for $\forall\, a_i \in (S_1-I)$ either $f(a_i) < f(a_c)$ or $f(a_i) > f(a_k)$ i.e. there does not exist a_p, $a_q \in (S_1-I)$ such that $f(a_p) < f(a_c)$ and $f(a_q) > f(a_k)$.

(ii) $\forall\, a_i \in (S_1-I),\ f(a_i) < f(a_c) \Leftrightarrow \forall\, a_j \in (S_2-I),\ f(a_j) > f(a_k)$

Proof: We shall prove the lemma by contradiction.

Part (i) Let us suppose to the contrary that there exist a_p, $a_q \in (S_1-I)$ such that $f(a_p) < f(a_c)$ and $f(a_q) > f(a_k)$. By Lemma 1 and Lemma 3, we have that $I_1 = \bigcap_{q_i \in Q_1} q_i \neq \phi$. It is easily observed that $I_1 \cap (S_2-I) = \phi$ and $I_2 \cap (S_1-I) = \phi$.

Further since a_p, a_q does not belong to any set in Q_2 and since the linear ordering defined by f implies the L.O. property of $Q_1 \cup Q_2$, we have that, for all $a_j \in S_2$, $f(a_p) < f(a_j) < f(a_q)$. Thus we have:

$$\ldots < f(a_p) < \ldots \quad \ldots \overset{\leftarrow \text{This interval contains images of all elements} \in S_2 \rightarrow}{\underbrace{< f(a_c) < \quad \ldots\ldots \quad < f(a_k)}_{\substack{\text{images of elements} \in I \\ \text{(By Lemma 9)}}}} < \ldots\ldots \ \ldots \quad < f(a_q) < \ldots$$

Now, there exist q_i, $q_j \in Q_1$ such that $q_i \supseteq \{a_p\} \cup I_1$ and $q_j \supseteq \{a_q\} \cup I_1$. Since $I_1 \cap (S_2-I) = \phi$, the interval corresponding to q_i or the interval corresponding to q_j contains images of elements $\in (S_2-I)$ which are foreign to all sets in Q_1. This means that the linear ordering defined by f does not imply L.O. property of $(Q_1 \cup Q_2)$ which is a contradiction.

Part (ii) $\Rightarrow$. We have $f(a_i) < f(a_c)$ for $\forall\, a_i \in (S_1-I)$. Assume to the contrary that there exists an $a_j \in (S_2-I)$ such that $f(a_j) < f(a_k)$. By Lemma 9, $f(a_j) < f(a_c)$. Since a_j is foreign to all sets containing elements $\in (S_1-I)$, we have $f(a_j) < \underset{a_r \in S_1}{\text{Min}}\,(f(a_r))$. There exist a set $\bar{q}_i \in Q_2$ such that $\bar{q}_i \supseteq \{a_j\} \cup I_2$. As $\bar{q}_i \cap (S_1-I) = \phi$, $f(a_\ell) < \underset{a_r \in S_1}{\text{Min}}\,(f(a_r))$ for $\forall\, a_\ell \in I_2$. Now consider any set $\bar{q}_p \in Q_2$ such that $\bar{q}_p \supseteq \{a_k\} \cup I_2$. The interval corresponding to $\bar{q}_p$ contains the images of elements $\in (S_1-I)$ which are foreign to all sets in Q_2. This leads to a contradiction.

$\Leftarrow$. The same arguments as above direct us to the conclusion that if for $\forall\, a_j \in (S_2-I)$, $f(a_j) > f(a_k)$, then for $\forall\, a_j \in (S_1-I)$, $f(a_j) < f(a_c)$.

QED ****

Let h_1 and h_2 be Hamiltonian paths in $\bar{G}(Q_1)$ and $\bar{G}(Q_2)$ respectively. Let $I = S_1 \cap S_2$. We see that h_1 induces a subpath in the set of nodes that correspond to I in $\bar{G}(Q_1)$. The starting and end nodes of this subpath are nodes that correspond to some elements in I and the subpath contains all the nodes which correspond to the elements $\in$ I. Let h_1^I denote this subpath. Similarly h_2^I is the subpath induced by h_2 in the set of nodes that correspond to I in $\bar{G}(Q_2)$. We say that the Hamiltonian paths h_1 and h_2 are <u>consistent ($h_1 \sim h_2$)</u> iff exactly one of the following holds:

(i) $S_1 \cap S_2 = I = \phi$

(ii) $h_1^I = h_2^I$ and the left neighbour of the starting node of h_1^I is empty in h_1 or h_2 and the right neighbour of the end node of h_1^I is empty in h_1 or h_2.

Example 4.9: Consider the following Hamiltonian paths h_1 and h_2. Let $h_1 = \langle a_1, a_2, a_3, a_4, a_5 \rangle$ and $h_2 = \langle a_3, a_4, a_5, a_6, a_7, a_8 \rangle$

$$S_1 = \{a_1, a_2, a_3, a_4, a_5\}$$

$$S_2 = \{a_3, a_4, a_5, a_6, a_7, a_8\}$$

$$I = S_1 \cap S_2 = \{a_3, a_4, a_5\}$$

$$h_1^I = \langle a_3, a_4, a_5 \rangle = h_2^I$$

The starting node of $h_1^I = a_3$ and the end node of $h_1^I = a_5$. The left neighbour of a_3 is empty in h_2 and the right neighbour of a_5 is empty in h_1. Hence $h_1 - h_2$. ****

Theorem 7: Let $\Omega(Q_1)$ and $\Omega(Q_2)$ be complete. If $Q_1 \cup Q_2$ has L.O. property, then there exist Hamiltonian paths h_1 in $\bar{G}(Q_1)$ and h_2 in $\bar{G}(Q_2)$ such that h_1 and h_2 are consistent.

Proof:

By Lemma 1, Q_1 and Q_2 have L.O. property. Since $\Omega(Q_1)$ and $\Omega(Q_2)$ are also complete, by Theorem 5 there exist Hamiltonian paths in $\bar{G}(Q_1)$ and in $\bar{G}(Q_2)$.

Case (i): $I = S_1 \cap S_2 = \phi$. Any Hamiltonian path in $\bar{G}(Q_1)$ is consistent with any Hamiltonian path in $\bar{G}(Q_2)$.

Case (ii): $I \neq \phi$. Let f be a function that defines a linear ordering of the elements $\in \tilde{S}$ implying the L.O. property of $Q_1 \cup Q_2$. We have the following situations:

(1) $I = S_2$, i.e. $S_2 \subseteq S_1$ and $\tilde{S} = S_1$

Define $f_1(a_i) = f(a_i) \; \forall \; a_i \in S_1$

$f_2(a_i) = f(a_i) \; \forall \; a_i \in S_2$

Clearly, f_1 defines a linear ordering, say O_1, of the elements $\in S_1$ that establishes the L.O. property of Q_1. So does f_2 w.r.t. Q_2. Let the linear ordering defined by f_2 be O_2. Since $f_2(a_i) = f_1(a_i)$ for $\forall \; a_i \in S_2$, we have that, for $\forall \; a_\ell, a_k \in S_2$, a_ℓ precedes a_k in O_2 iff a_ℓ precedes a_k in O_1. By Lemma 9, there does not exist an $a_p \in (S_1-S_2)$ and $a_c, a_d \in S_2$ such that $f_1(a_p)$ is in between $f_1(a_c)$ and $f_1(a_d)$. Hence, if h_1 and h_2 are Hamiltonian paths in $\bar{G}(Q_1)$ and $\bar{G}(Q_2)$ corresponding to O_1 and O_2 respectively (see proof of Theorem 5), then $h_2 = h_2^I = h_1^I$. The left neighbor of the starting node of h_2 and the right neighbor of the end node of h_2 are both empty. Then $h_1 \sim h_2$.

(2) $I = S_1$, i.e. $S_1 \subseteq S_2$. The proof is similar to that of (1), above.

(3) $I \neq \phi$ or S_1 or S_2. Let $I = \{a_1, a_2, \ldots, a_k\}$. Without loss of generality, we can let f be such that $f(a_1) < f(a_2) < \ldots < f(a_k)$. By Lemma 10, we can assume without any loss in generality that for $\forall \; a_i \in (S_1-I)$, $f(a_i) < f(a_1)$ and $\forall \; a_i \in (S_2-I)$, $f(a_i) > f(a_k)$.

Now, define f_1, f_2:

$f_1(a_i) = f(a_i) \quad \forall \; a_i \in S_1$

$f_2(a_i) = f(a_i) \quad \forall \; a_i \in S_2$

Clearly, f_1 and f_2 define linear orderings, say 0_1 and 0_2, of the elements belonging to S_1 and S_2 respectively such that the L.O. property of Q_1 and Q_2 are implied. Let h_1 and h_2 be the Hamiltonian paths corresponding to 0_1 and 0_2 in $\bar{G}(Q_1)$ and $\bar{G}(Q_2)$ respectively (see the proof of Theorem 5). Then $h_1^I = \langle a_1, a_2, \ldots, a_k \rangle = h_2^I$. a_1 is the starting node of h_1^I and its left neighbor is empty in h_2. The right neighbor of a_k, the end node of h_1^I, is empty in h_1. We thus have $h_1 \sim h_2$. QED ****

Section 5: Union of Linearly Orderable Families whose Intersection Graphs are Complete.

Let $Q = \{Q_1, Q_2, \ldots, Q_m\}$ be a set of families of sets with $Q_i \cap Q_j$ not necessarily empty. For $1 \leq i \leq m$, let $\Omega(Q_i)$ be complete and S_i denote the set $\bigcup_{q_j \in Q_i} \{q_j\}$. $\tilde{S}$ indicates $\bigcup_{i=1,\ldots,m} S_i = \{a_1, a_2, \ldots, a_n\}$. Let $h_1, h_2, \ldots, h_m$ be pair-wise consistent Hamiltonian paths in $\bar{G}(Q_1), \bar{G}(Q_2), \ldots, \bar{G}(Q_m)$ respectively where, for $1 \leq i \leq m$, $\bar{G}(Q_i)$ is the DSG of Q_i. Define a directed graph $\tilde{G}(Q) = [\tilde{S}', \tilde{R}]$. $\tilde{S}'$ is the set of nodes of $\tilde{G}(Q)$ and is $\{a_1, a_2, \ldots, a_i, \ldots, a_n\}$ where node a_i corresponds to element $a_i \in \tilde{S}$ and vice versa. $a_i \; \tilde{R} \; a_j$ iff there exists a Hamiltonian path h_k, $1 \leq k \leq m$, in which a_i precedes a_j. $a_i \; \tilde{R} \; a_j \Leftrightarrow (a_i, a_j)$ is an edge of $\tilde{G}(Q)$. $\tilde{G}(Q)$ is called a <u>Partial Order (P.O.)</u> graph of Q corresponding to $Q_1, Q_2, \ldots, Q_m$.

An <u>undirected path</u> or simply a <u>path</u> in a directed graph G is a sequence of distinct nodes $a_1, a_2, \ldots, a_k$ such that for $i = 1, 2, \ldots, (k-1)$, $\{a_i, a_{i+1}\}$ are edges of G. Note that we ignore the direction of the edges in G. A connected-directed graph is a directed graph in which there is a path between every pair of distinct nodes. A component G' of a directed graph G is a subgraph of G such that G' is a connected-directed graph and is not properly contained in any other connected-directed subgraph of G. A directed path in a directed graph G is a sequence of distinct nodes a_1, a_2, ..., a_k such that for $1 \leq i \leq k-1$, (a_i, a_{i+1}) are edges of G. A Hamiltonian path in a directed graph G is a directed path that passes through all the nodes of G. If $P = \langle a_1, a_2, \ldots, a_k \rangle$ is a path in G and (a_k, a_1) is also an edge of G, then P is also a directed cycle. We can distinguish by context whether by P, we mean a directed path or a directed cycle. If G does not have any directed cycles, then G is called acyclic. The length of a cycle is the number of nodes in the cycle. All these definitions are more or less standard in graph theory and may be found in [8], [11] or [12].

Example 4.10: Consider the directed graph G in Figure 4.12. It has two components. If R is the connectivity relation of G, then the components are $[\{a_1, a_2, a_5, a_6\}, R]$ and $[\{a_3, a_4\}, R]$. $\langle a_1, a_2, a_5 \rangle$ is a directed cycle of G and is of length 3. ****

Let G be an undirected graph. $G_1, G_2, \ldots, G_m$ be a set of complete subgraphs of G(i.e. $G_1, G_2, \ldots, G_m$ are subgraphs of G and are complete) such that every node and edge of G is in at least one of them. Then G is said to be covered by $G_1, G_2, \ldots, G_m$. This is exemplified in example 4.11. We are now ready to prove a major result concerning the L.O. property of an arbitrary

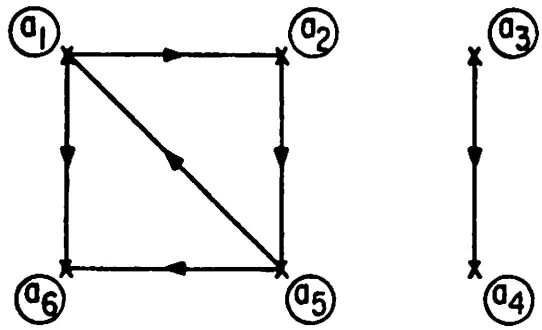

Figure 4.12

family of sets.

Theorem 8: Let $G_1, G_2, \ldots, G_m$ be a set of complete subgraphs of $\Omega(Q)$ that cover $\Omega(Q)$. Let $Q_i \subseteq Q$ be such that $G_i = \Omega(Q_i)$ for $1 \leq i \leq m$. Q has L.O. property iff there exists a P.O. graph $\tilde{G}(Q)$ corresponding to $Q_1, Q_2, \ldots, Q_m$ and any $\tilde{G}(Q)$ acyclic.

Proof: The if-part of the theorem: We first show that there exists a P.O. graph $\tilde{G}(Q)$ of Q. Let $\bar{G}(Q_i)$ be the DSG of Q_i. If $\bar{G}(Q_i)$ does not have a Hamiltonian path, then by theorem 5, the sets in Q_i are not linearly orderable. By Lemma 1, this implies that Q does not have L.O. property which is a contradiction. Hence, every one of $\bar{G}(Q_i)$, $i = 1, 2, \ldots, m$, has a Hamiltonian path. If $\bar{G}(Q_i)$ and $\bar{G}(Q_j)$, $i \neq j$, does not have Hamiltonian paths that are consistent, then by theorem 7, $Q_i \cup Q_j$ does not possess the linear ordering property. Again by Lemma 1, this means that sets in Q are not linearly orderable which is not true. Hence there exist Hamiltonian paths h_1, $h_2, \ldots, h_m$ in $\bar{G}(Q_1), \bar{G}(Q_2), \ldots, \bar{G}(Q_m)$ such that they are pair-wise consistent, i.e., a $\tilde{G}(Q)$ exists.

Suppose to the contrary that there exists a $\tilde{G}(Q)$ containing directed cycles. Let $C = a_1, a_2, \ldots, a_i, a_{i+1}, \ldots, a_k$ be a directed cycle of minimum length in $\tilde{G}(Q)$. Since $h_1, h_2, \ldots, h_m$ are pair-wise consistent, length of $C \neq 2$. Then, let the length of $C \geq 3$. We have $S_i = \bigcup_{q_j \in Q} \{q_j\}$ for $1 \leq i \leq m$ and $\tilde{S} = \bigcup_{1 \leq i \leq m} S_i$. Either (a_i, a_j) or (a_j, a_i) is an edge of $\tilde{G}(Q) \Leftrightarrow$ there exists a S_p, $1 \leq p \leq m$, such that $S_p \supseteq \{a_i, a_j\}$. Since C is of minimum length, there does not exist a S_p, $1 \le p \le m$, such that S_p contains more than 2 elements that correspond to nodes in C (see figure 4.13). Hence without loss in generality, we can assume that a_i, $a_{i+1} \in S_i$ for $1 \le i \le (k-1)$ and a_k, $a_1 \in S_k$.

Since Q has L.O. property, there exists a function f that defines a linear ordering of the elements $\in \tilde{S}$ implying the L.O. property of Q. Now, consider $\Omega(Q_i)$. Since a_1, a_2, ..., a_{i-1}, a_{i+2}, ..., a_k are foreign to all sets in Q_i and a_i, $a_{i+1} \in S_i$, $f(a_\ell)$ is not between $f(a_i)$ and $f(a_{i+1})$ for $\ell = 1, 2, \ldots, i-1, i+2, \ldots, k$. (This can be seen by similar arguments as in Lemma 9.)

Applying the above contention to $\Omega(Q_1)$, $\Omega(Q_2)$, ..., $\Omega(Q_{k-1})$ and $\Omega(Q_k)$, we get the contradiction that $f(a_2)$, $f(a_3)$, $f(a_4)$, ..., $f(a_{k-1})$ are between $f(a_1)$ and $f(a_k)$, and $f(a_2)$, $f(a_3)$, $f(a_4)$, ..., $f(a_{k-1})$ are not between $f(a_k)$ and $f(a_1)$. Thus, Q has L.O. property $\Rightarrow \tilde{G}(Q)$ exists and any $\tilde{G}(Q)$ is acyclic.

To prove the sufficiency of the conditions, we shall show how to construct for any family Q satisfying the conditions, a function f and a set of intervals implying that Q is linearly orderable.

Let $\tilde{G}_1$ $\tilde{G}_2$, ..., $\tilde{G}_p$ be the components of $\tilde{G}(Q)$. Define function f:

(i) for $\forall$ a_i, $a_j \in \tilde{G}_k$, $1 \le k \le p$, $f(a_i) < f(a_j)$ iff there exists a directed path from a_i to a_j. (ii) for $\forall\, a_i \in \tilde{G}_k$ and $\forall\, a_j \in \tilde{G}_\ell$, $k < \ell$, $f(a_i) < f(a_j)$. Since $\tilde{G}(Q)$ is acyclic, such a function exists. For each $q_i \in Q$, define $I_i = [\min_{a_p \in q_i} f(a_p), \max_{a_p \in q_i} f(a_p)]$.

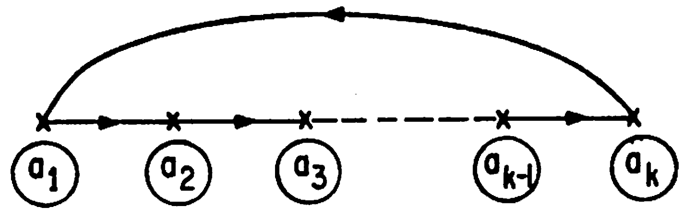

Figure 4.13

Interval I_i contains the images of all elements $\in q_i$.

To see that I_i does not contain images of any foreign elements w.r.t. q_i, we suppose to the contrary that it does and show that it leads to a contradiction. Let there exist a_b, $a_d \in q_i$, a_{c_1}, a_{c_2}, ..., $a_{c_j} \in (\tilde{S}-q_i)$ such that for $1 \leq k \leq j$, $f(a_{c_k})$ is between $f(a_b)$ and $f(a_d)$. Further let a_b, a_d be such that there does not exist an $a_p \in q_i$ and $f(a_p)$ is between $f(a_b)$ and $f(a_d)$. Without loss in generality, we can assume that $f(a_b) < f(a_{c_1}) < \cdots < f(a_{c_j}) < f(a_d)$. Then, (a_b, a_{c_1}) is an edge of $\tilde{G}(Q)$ and a_{c_1} is the right neighbor of a_b in some Hamiltonian path h_t, used to define $\tilde{G}(Q)$.

q_i belongs to at least one complete subgraph, say G_ℓ, that was chosen to cover $\Omega(Q)$. Let Q_ℓ be such that $\Omega(Q_\ell) = G_\ell$ and h_ℓ be the Hamiltonian path in $\tilde{G}(Q_\ell)$ that was used in the definition of $\tilde{G}(Q)$. If a_d precedes a_b in h_ℓ, then (a_d, a_b) will be an edge of $\tilde{G}(Q)$ and hence a_b, a_{c_1}, ..., a_{c_k}, ..., a_{c_j}, a_d will be a directed cycle of $\tilde{G}(Q)$ which is not possible. Hence, let a_b precede a_d in h_ℓ. Since a_{c_1} is foreign to q_i, by Lemma 6 a_{c_1} is not between a_b and a_d in h_ℓ. Hence $h_\ell \neq h_t$.

The right neighbor of a_b is not empty in both h_ℓ and h_t. The right neighbor of a_b in $h_t = a_{c_1}$ which is not the right neighbor of a_b in h_ℓ since a_b precedes a_d in h_ℓ and a_{c_1} is not between a_b and a_d in h_ℓ. This leads us to the contradiction that h_ℓ and h_t are not consistent. <u>QED</u>. ****

<u>Example 4.11</u>:

Let $q_1 = \{a_1, a_2, a_3\}$

$q_2 = \{a_2, a_3, a_4, a_5\}$

$q_3 = \{a_2, a_3, a_4\}$

$q_4 = \{a_4, a_5, a_6\}$

$q_5 = \{a_4, a_5, a_6, b_1\}$

$q_6 = \{b_1, b_2\}$

$q_7 = \{a_7, a_8\}$

and $q_8 = \{a_7, a_9\}$

Let $Q = \{q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8\}$.

$\Omega(Q)$ is given in <u>figure 4.14</u>. Let R be the connectivity relation of $\Omega(Q)$

Let $G_1 = [\{q_1, q_2, q_3\}, R]$

$G_2 = [\{q_2, q_3, q_4, q_5\}, R]$

$G_3 = [\{q_5, q_6\}, R]$

and $G_4 = [\{q_7, q_8\}, R]$

G_1, G_2, G_3, G_4 are complete subgraphs of $\Omega(Q)$ that cover $\Omega(Q)$. We have $Q_1 = \{q_1, q_2, q_3\}$, $Q_2 = \{q_2, q_3, q_4, q_5\}$, $Q_3 = \{q_5, q_6\}$ and $Q_4 = \{q_7, q_8\}$. $\overline{G}(Q_1)$, $\overline{G}(Q_2)$, $\overline{G}(Q_3)$ and $\overline{G}(Q_4)$ are given in <u>figures 4.15, 4.16, 4.17 and 4.18</u> respectively.

$h_1 = \langle a_1, a_2, a_3, a_4, a_5 \rangle$

$h_2 = \langle a_2, a_3, a_4, a_5, a_6, b_1 \rangle$

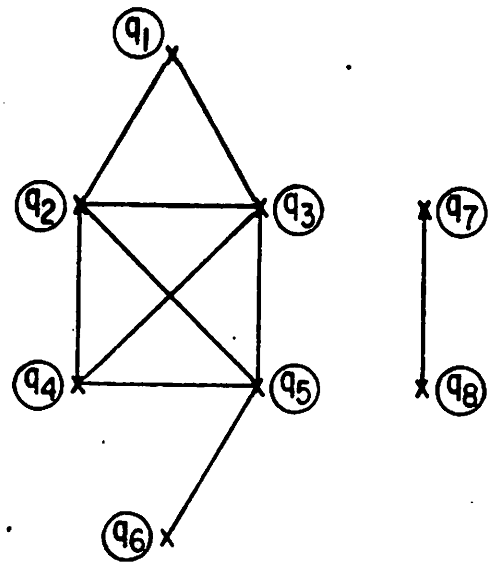

Figure 4.14

$h_3 = \langle a_4, a_5, a_6, b_1, b_2 \rangle$

$h_4 = \langle a_8, a_7, a_9 \rangle$

h_1, h_2, h_3, h_4 are pair wise consistent Hamiltonian paths in $\bar{G}(Q_1)$, $\bar{G}(Q_2)$, $\bar{G}(Q_3)$ and $\bar{G}(Q_4)$ respectively. Let h_1, h_2, h_3 and h_4 define $\tilde{G}(Q)$. See figure 4.19. For the sake of clarity, we have not shown in Fig. 4.19 all the edges of $\tilde{G}(Q)$ which is directed-cycle free. Hence Q has the L.O. property. Let $\tilde{R}$ be the connectivity relation of $\tilde{G}(Q)$. We note that there are two components of $\tilde{G}(Q)$, namely $\tilde{G}_1$ and $\tilde{G}_2$ where $\tilde{G}_1 = [\{a_1, a_2, a_3, a_4, a_5, a_5, b_1, b_2\}, \tilde{R}]$ and $\tilde{G}_2 = [\{a_7, a_8, a_9\}, \tilde{R}]$. Let f be: $f(a_1) = 1$, $f(a_2) = 2$, $f(a_3) = 3$, $f(a_4) = 4$, $f(a_5) = 5$, $f(a_6) = 6$, $f(b_1) = 7$, $f(b_2) = 8$, $f(a_8) = 9$, $f(a_7) = 10$ and $f(a_9) = 11$. The intervals corresponding to q_i, $1 \leq i \leq 8$ are shown in figure 4.20. From Figure 4.20 the L.O. property of Q is evident. ****

Lemma 11: Let G_1 and G_2 be complete subgraphs of $\Omega(Q)$ such that G_1 and G_2 cover $\Omega(Q)$. Let $Q_1 \subseteq Q$, $Q_2 \subseteq Q$ be such that $\Omega(Q_1) = G_1$ and $\Omega(Q_2) = G_2$. Let h_1 and h_2 be consistent Hamiltonian paths in $\bar{G}(Q_1)$ and $\bar{G}(Q_2)$. Then $\tilde{G}(Q)$ defined by h_1 and h_2 is acyclic.

Proof: Suppose that $\tilde{G}(Q)$ is not acyclic. Let $C = a_1, a_2, \ldots a_k$ be a cycle of minimum length in $\tilde{G}(Q)$. First, we observe that the length of $C \leq 3$. This can be seen by arguments similar to the ones in Theorem 8. If the length of C is 2, then (a_1, a_2), (a_2, a_1) are edges of $\tilde{G}(Q)$. This implies that a_1 precedes a_2 in $h_2(h_1)$ and a_2 precedes a_1 in $h_1(h_2)$. We then have a contradiction that h_1 and h_2 are not consistent. If the length of C is 3, then in $h_1(h_2)$

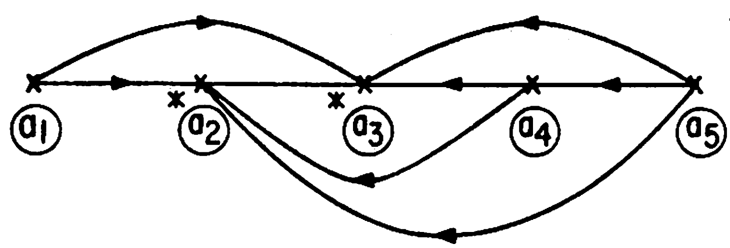

Figure 4.15

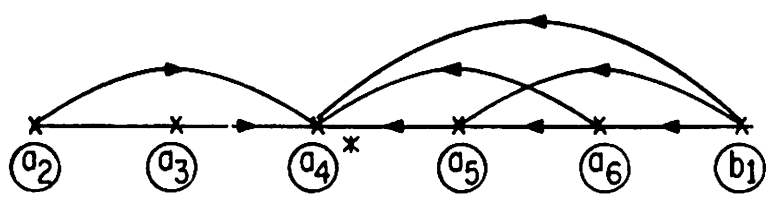

Figure 4.16

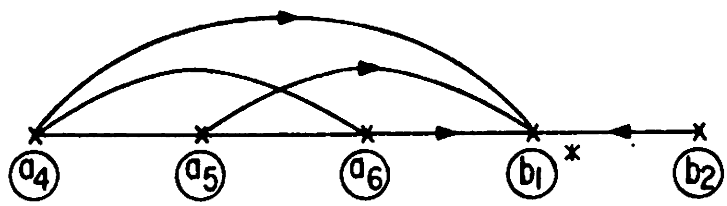

Figure 4.17

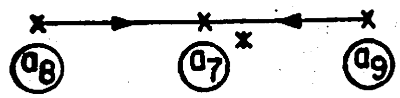

Figure 4.18

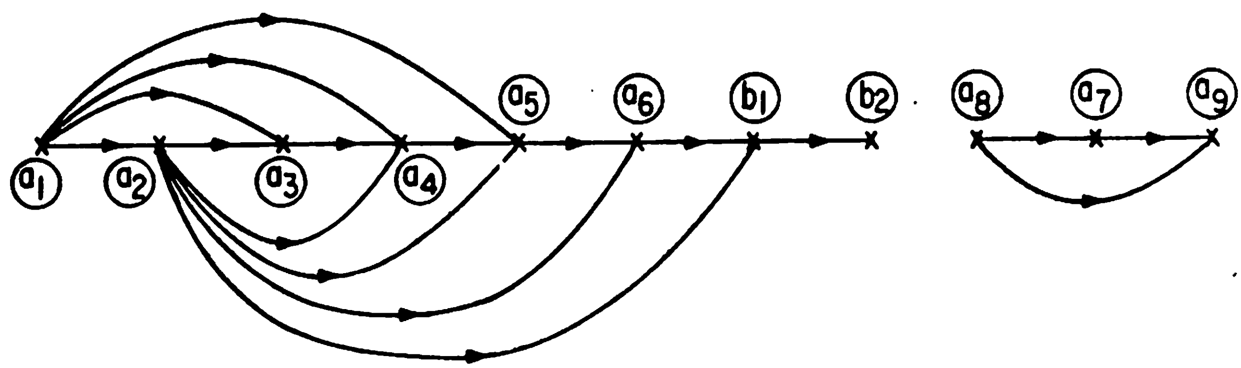

Figure 4.19

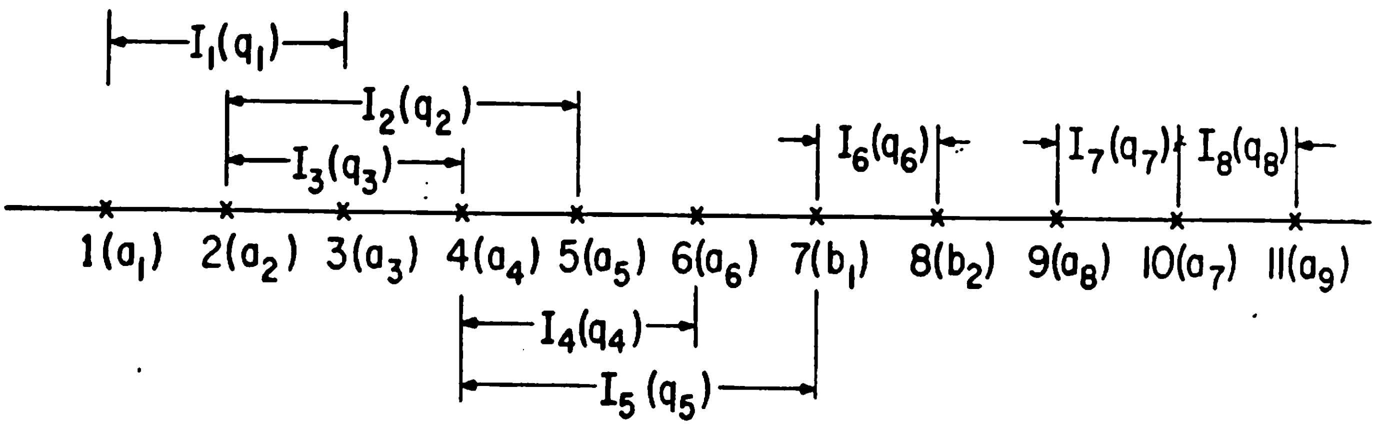
I1(q1)
I2(q2)
I3(q3)
I6(q6)
I7(q7)
I8(q8)
1(a1)
2(a2)
3(a3)
4(a4)
5(a5)
6(a6)
7(b1)
8(b2)
9(a8)
10(a7)
11(a9)
I4(q4)
I5(q5)

Figure 4.20

a_1 precedes a_2 and a_3, and a_2 precedes a_3. In $h_2(h_1)$, a_3 precedes a_1. This also leads to the contradiction that h_1 is not consistent with h_2. QED ****

Theorem 9: If G_1 and G_2 are complete subgraphs of $\Omega(Q)$ such that G_1 and G_2 cover $\Omega(Q)$, then Q possesses the L.O. property iff there exist consistent Hamiltonian paths in $\bar{G}(Q_1)$ and $\bar{G}(Q_2)$ where $Q_1 \subseteq Q$, $Q_2 \subseteq Q$ and $\Omega(Q_1) = G_1$, $\Omega(Q_2) = G_2$.

Proof: The necessity is theorem 7. If there exist consistent Hamiltonian paths in $\bar{G}(Q_1)$ and $\bar{G}(Q_2)$, then there exists a $\tilde{G}(Q)$. By Lemma 11, $\tilde{G}(Q)$ is acyclic. Then, by Theorem 8, Q has the L.O. property. Thus we have the sufficiency. QED ****

Based on the fact that every pair of Hamiltonian paths used in the definition of a P.O. graph $\tilde{G}(Q)$ of Q is consistent, we have the following lemma.

Lemma 13: There exists a directed path between every pair of nodes in each component of $\tilde{G}(Q)$ - (i.e. if a_i and a_j are two nodes belonging to the same component in $\tilde{G}(Q)$ then there is a path from a_i to a_j or from a_j to a_i.)

Proof: Let $\tilde{G}(Q) = [\tilde{S}', \tilde{R}]$. Assume to the contrary that the lemma is untrue. Then, there exists a component $\tilde{G}_k$ of $\tilde{G}(Q)$ and nodes a_i, a_j, a_p in $\tilde{G}_k$ such that a_i and a_j are not comparable w.r.t. $\tilde{R}$ and either

(i) (a_i, a_p), (a_j, a_p) are edges of $\tilde{G}(Q)$ or

(ii) (a_p, a_i), (a_p, a_j) are edges of $\tilde{G}(Q)$.

Case (i): a_i and a_j are not $\tilde{R}$-related and (a_i , a_p), (a_j , a_p) are edges of $\tilde{G}_k$. Then a_i precedes a_p in some Hamiltonian path h_ℓ and a_j precedes a_p is some other Hamiltonian path h_t when h_ℓ and h_t are among the Hamiltonian paths used to define $\tilde{G}(Q)$. Then, since the left neighbor of a_p is not empty in h_ℓ and h_t and $a_i \neq a_j$, we are led to the contradiction that h_1 and h_2 are not consistent.

Case (ii): a_i and a_j are not $\tilde{R}$-related and (a_p, a_i) and (a_p, a_j) are edges of $\tilde{G}_k$. By similar arguments as in case (i), we get the same contradiction. QED ****

Lemma 14: Let G be a directed graph satisfying that there exists a directed path between every pair of nodes of G. If G is acyclic, then there exists one and only one Hamiltonian path in G.

Proof: First, we observe that there exists a single node s in G with no edge directed to it and a single node t with no edge directed from it [12]. s is called the source and t the sink. Let $S_1 = \{s\}$. We delete from G the node s and the edges leading from it and obtain the graph $\bar{G}$.

Let S_2 be the set of sources in $\bar{G}$. Suppose to the contrary that there exist distinct nodes i, j belonging to S_2. Since G is acyclic, the path between i and j does not include s. Hence, there is a path between i and j in $\bar{G}$. This implies that one of {i,j} is not a source which is a contradiction. We know that there is at least one source in $\bar{G}$ (See [12], page 64.) Hence $|S_2| = 1$. Let $S_2 = \{\ell\}$. Note that the path between s and ℓ is from s to ℓ in G. Assume to the contrary that there exists an intermediate node k in the path from s to ℓ in G. Then there is a path from k to ℓ in $\bar{G}$. This contradicts the fact that ℓ is the source in $\bar{G}$. Hence (S,ℓ) is an edge in G.

By repeating this argument for $1 \leq i < n$, we get the disjoint singleton sets S_i such that for $1 \leq j < n$ if $S_j = \{p\}$ and S_{j+1} $\{q\}$, then (p,q) is an edge of G. $S_n = \{t\}$ and there is no edge directed from t. This means that there is a path from s to t including all the nodes in G which is Hamiltonian.

Let H_1 be a Hamiltonian path in G. Since G is acyclic, for all nodes i, j in G, i precedes j in H_1 implies that i precedes j in any directed path passing through i and j. This fact establishes the uniqueness of H_1.

QED ****

From lemmas 13 and 14 the following theorem is immediate.

Theorem 10: There exists one and only one Hamiltonian path in every component of $\tilde{G}(Q)$, if $\tilde{G}(Q)$ is acyclic. ****

We now present an algorithm to obtain a function f and a set of intervals $\{I_1, I_2, \ldots, I_m\}$ such that $(f; I_1, I_2, \ldots, I_m)$ implies the L.O. property of Q.

Algorithm (Al):

(i) Obtain a P.O. graph $\tilde{G}(Q)$ of Q. If there does not exist a $\tilde{G}(Q)$ or if $\tilde{G}(Q)$ is not acyclic, Q does not possess the property of linear ordering.

(ii) Let $\tilde{G}_1, \tilde{G}_2, \ldots, \tilde{G}_p$ be the components of $\tilde{G}(Q)$. Get the Hamiltonian paths $H_1, H_2, \ldots, H_p$ in $\tilde{G}_1, \tilde{G}_2, \ldots, \tilde{G}_p$.

(iii) Define function f: (1) $\forall\ a_i \in \tilde{G}_k,\ a_j \in \tilde{G}_r,\ k < r,\ f(a_i) < f(a_j)$ and (2) $\forall\ a_i, a_j \in \tilde{G}_k,\ i \neq j,\ 1 \leq k \leq p,\ f(a_i) < f(a_j)$ iff a_i precedes a_j in H_k.

(iv) $\forall\ q_i \in Q$, define $I_i = [\min_{a_j \in q_i} f(a_j), \max_{a_j \in q_i} f(a_j)]$

Theorem 11: The function and the intervals defined by the above algorithm imply that the sets in the family Q are linearly orderable.

Proof: By Theorem 10, $H_1, H_2, \ldots, H_p$ exist. The rest of the proof is the same as the proof of the sufficiency part of Theorem 8. ****

Section 6. Complexity:

We have given necessary and sufficient conditions for a family of sets Q to have the property of linear ordering. We also gave an algorithm. The problem may also be formulated as a property of a matrix. Let M be an n×m matrix whose rows correspond to the n elements of S and whose columns correspond to the m rows of the sets in the family Q. $M_{ij} = 1$ if $a_i \in q_j$ and is zero otherwise. The linear ordering property of Q can be stated as a property of M. If there exist a permutation of the rows of M for which the "ones" in every column of M are in consecutive positions, then Q has the L.O. property. The "consecutive-one property" of a 0-1 matrix has been studied by mathematicians for special cases. The most general solution of this problem is given by Fulkerson and Gross [13]. On having written this thesis, the author became aware of this. Their approach is not graph theoretic. They also do not treat the problem from the information retrieval point of view [5].

In our solution there were five specific steps:

step (i): Find $G_1, G_2, \ldots, G_k$ so that G_i for $1 \leq i \leq k$ are complete subgraphs of $\Omega(Q)$ and $G_1, G_2, \ldots, G_k$ cover $\Omega(Q)$.

step (ii): See if each one of the subfamilies corresponding to these subgraphs has the L.O. property.

step (iii): Find consistent Hamiltonian paths in the directed semantic graphs of these subfamilies. Define the partial order graph $\tilde{G}(Q)$.

step (iv): Check if $\tilde{G}(Q)$ is acyclic.

step (v): Find Hamiltonian paths in every component of $\tilde{G}(Q)$.

For each edge (q_i, q_j) of $\Omega(Q)$, define G_{ij} to be the subgraph $[\{q_i, q_j\}, R]$ where R is the connectivity relation of $\Omega(Q)$. Then the subgraphs corresponding to all the edges of $\Omega(Q)$ are complete and cover $\Omega(Q)$. Hence, step (i) of the procedure does not require any work. In order to find a Hamiltonian path in the DSG $\bar{G}(Q)$ of Q whose intersection graph is complete, we partition the set of non-direction-changer nodes into P_1 and P_2 so that if $a_i \in P_1$ (P_2) then a_i is not comparable to any node in P_2 (P_1). Existence of such a partition is evident from theorem 4 and Corollary 2. If $h = \langle a_1, a_2, \ldots, a_i, \ldots, a_j, \ldots, a_n \rangle$ is a Hamiltonian path of $\bar{G}(Q)$, then (i) h consists of all the nodes of $\bar{G}(Q)$ and (ii) if a_i precedes a_j in h and $a_i, a_j \in P_1$ (P_2), then degree of a_i is less than or equal to (greater than or equal to) the degree of a_j. Degree of a node a_i is the number of edges incident on a_i [8]. Hence to get a Hamiltonian path in a DSG (step(ii)), all that is required is to sort the nodes of P_1 and P_2 on their degree. This is of complexity $O(n \log n)$ where n is the number of nodes of the DSG. Step (iii) is the most difficult one. Step (iv) is of complexity $O(n^2)$ where n is the number of nodes of $\bar{G}(Q)$ when we use the depth-first-search technique [14]. We use the depth-first-search technique and see if the number of strong components of $\tilde{G}(Q)$ is equal to the number of nodes of $\tilde{G}(Q)$. Finding a Hamiltonian path in a component of $\tilde{G}(Q)$ is same as finding a longest or critical path in the component which is acyclic. This needs $O(n^2)$ steps where n is again the number of nodes of $\tilde{G}(Q)$.

Though we have not established the complexity of our algorithm, we have included this discussion to throw some light on the amount of work involved at each step of the procedure.

CHAPTER 5.

UPDATING A CR FILE ORGANIZATION

In Chapter 4 we assumed that the file is time invariant and obtained necessary and sufficient conditions for a query family Q to be consecutively retrievable with respect to a file F. We also got an algorithm to find a CR file organization for Q w.r.t. F. Let CR(F,Q) be a CR file organization for the family Q w.r.t. F. In this chapter, we shall consider the problem of updating CR(F,Q).

First let us introduce some notations. If A and B are two sets then in the sequel we shall let A-B denote the set of elements belonging to A and not to B. If q_i is a query in the family Q and if $\rho(q_i)$ is the set of reply records for q_i, for simplicity of notation, we shall let q_i denote $\rho(q_i)$. It should be clear from the usage whether we mean the query q_i or the set of reply records for q_i. If $CR_1 = r_1\ r_2\ \ldots\ r_m$ and $CR_2 = \bar{r}_1\ \bar{r}_2\ \ldots\ \bar{r}_n$ are two CR organizations with no records in common then $CR_1 \cdot CR_2 = r_1\ r_2\ \ldots\ r_m\ \bar{r}_1\ \bar{r}_2\ \ldots\ \bar{r}_n$ is a CR organization.

Insertion and deletion are the two situations of updating. There are two cases of deletion: (i) deletion of records from the file F. Let R be the set of records being deleted from F. Define $\bar{Q} = \{\bar{q}_i \mid \bar{q}_i = q_i - R\}$. We need to analyse $\bar{Q}$ with respect to F-R. (ii) deletion of queries from the family Q of queries.

If $Q_1 \subset Q$ is the family of queries being deleted, then we want to examine $Q-Q_1$ with respect to $\bigcup_{q_i \in Q-Q_1} \{q_i\}$. Similarly there are two cases of insertion: (i) Adding queries whose pertinent records are in F. (ii) Adding a new set of records R to the file F and updating the replies to the queries in Q, i.e. for each query $q_i \in Q$, we add zero or more records from R as reply records.

If a query q_i with a set of reply records not contained in F is added to the system, then $CR(F,Q) \cdot q_i$ is a CR organization for $Q \cup \{q_i\}$ with respect to $F \cup q_i$. We can consider adding a query q with reply records $R_1 \cup R_2$ where $R_1 \cap R_2 = \phi$ and R_1 is a subset of records in file F and R_2 is a set of records not in F in two steps: First step is adding a query q_1 with reply records R_1 and analyse the CR organization for $Q \cup \{q_1\}$ w.r.t. $F \cup R_1$. The second step is to add the set of records R_2 to $(F \cup R_1)$ and update the reply of query q_1 in the family $(Q \cup \{q_1\})$.

When records or queries are deleted from a CR organization, the CR organization remains invariant. Let f be a function that maps F into storage locations of the linear storage medium and s_i be the sequence of consecutive locations corresponding to $q_i \in Q$. Suppose we delete a set R of records from the file F. Then the same function f and sequences $\{s_i\}$ satisfy the CR property of updated set of queries w.r.t. F-R. In practice, we may have some gaps created due to the deletion of records and need compacting. When a query q is deleted from a CR organization, we may (i) delete the query q from the directory and (ii) delete from the organization the records that are pertinent only to q.

Example 5.1: Let $Q_0 = \{q_0, q_1, q_2, q_3\}$ where $q_0 = \{r_0, r_1, r_2\}$, $q_1 = \{r_1, r_2, r_3\}$, $q_2 = \{r_2, r_3, r_4\}$ and $q_3 = \{r_0, r_1, r_2, r_3\}$. $r_0\ r_1\ r_2\ r_3\ r_4$ is a CR organization for Q_0.

If we delete the record r_2 from the file, then q_0 becomes $\{r_0, r_1\}$, q_1 becomes $\{r_1, r_3\}$, q_2 gets updated to $\{r_3, r_4\}$ and q_3 to $\{r_0, r_1, r_3\}$. $r_0\ r_1\ r_3\ r_4$ is the organization obtained by deleting r_2 from the CR organization for Q_0 and squeezing the gaps. $r_0\ r_1\ r_3\ r_4$ is a CR organization for the family of queries with updated set of replies.

If the query q_2 is deleted from Q_0 we may delete r_4 from the CR organization for Q_0 and obtain $r_0\ r_1\ r_2\ r_3$. This is a CR organization for the updated family of queries $\{q_0, q_1, q_3\}$. r_4 was deleted from the CR organization for Q_0 since r_4 belonged only to q_2, the query being deleted from the system. ****

Let us now consider adding a query q to a CR organization. The records pertinent to q are already in the file. Let Q_0 denote the old family of queries and $Q_N = Q_0 \cup \{q\}$ the new family of queries. We are looking for a CR organization for Q_N w.r.t. F. The following theorem gives some of the situations where a CR organization for Q_N may be obtained by a few changes to the existing CR organization for Q_0. We need the following definitions: Consider an undirected graph G. A path in G is a sequence of distinct nodes $i_1, \ldots i_j, i_{j+1}, \ldots i_k$ such that for $1 \leq j \leq k$, i_j is a node of G and for $1 \leq j \leq k, \{i_j, i_{j+1}\}$ is an edge of G. G is called connected if there is a path between every pair of nodes of G.

A <u>component</u> of G is a subgraph of G that is connected and is not properly contained in any connected subgraph of G [8].

<u>Example 5.2</u>: Consider the graph in <u>figure 4.14</u>. It is not a connected graph since there is no path from say q_5 to q_7. If R is the connectivity relation of the graph, then $[\{q_1, q_2, q_3, q_4, q_5, q_6\}, R]$ and $[\{q_7, q_8\}, R]$ are the components of the graph. ****

<u>Theorem 12</u>: If q satisfies any one of the following conditions, then Q_N has a CR organization:

(i) q covers every query $\in Q_0$, i.e. $q \supseteq q_j$ for $\forall q_j \in Q_0$.

(ii) $q \subseteq \bigcap_{q_i \in Q_0} q_i$

(iii) $q = \bigcap_{q_i \in Q'} q_i$ where $Q' \subseteq Q_0$.

(iv) $q \subseteq \bar{q}$ where $\bar{q} \in Q_0$ and there exist at most one $\check{q} \in Q_0$ such that $\check{q} \neq q$ and $\check{q} \cap \bar{q} \neq \phi$, i.e. in $\Omega(Q_0)$, zero or one edge is incident on $\bar{q}$, the node corresponding to $\bar{q}$.

(v) $q = \bigcup_{q_i \in \bar{Q}} \{q_i\}$ where $\bar{Q} \subseteq Q_0$ and $\Omega(\bar{Q})$ is connected.

(vi) $q = R_1 \cup R_2$, $R_1 \cap R_2 = \phi$ and there exist $Q_1 \subseteq Q_0$, $Q_2 \subseteq Q_0$ such that $\Omega(Q_1)$ and $\Omega(Q_2)$ are components of $\Omega(Q_0)$ and $R_1 = \bigcup_{q_i \in Q_1} \{q_i\}$ and $R_2 = \bigcup_{q_j \in Q_2} \{q_j\}$.

(vii) q can be partitioned into $R_1, R_2, \ldots, R_k$ such that for $1 \leq i \leq k, \exists Q_j \subseteq Q_0$, $\Omega(Q_j)$ is a component of $\Omega(Q_0)$ and $R_i = \bigcup_{q_\ell \in Q_j} \{q_\ell\}$.

<u>Proof:</u> <u>Part (i)</u> is obvious.

<u>Part (ii)</u>: In the CR organization for Q_0 w.r.t. F, the records $\in \bigcap_{q_i \in Q_0} q_i$ are consecutive and rearranging these records among themselves does not affect the CR property of Q_0. Rearranging these records such that the records $\in q$ are consecutive, we obtain a CR organization for Q_N.

<u>Part (iii)</u>: Observe that for any subfamily $Q' \subseteq Q_0$ the records $\in \bigcap_{q_i \in Q'} q_i$ are consecutive in the CR organization for Q_0. Then the CR organization for Q_0 is a CR organization for Q_N.

<u>Part (iv)</u>: If there is no query $\tilde{q}$ in Q_0 such that $\tilde{q} \neq \bar{q}$ and $\tilde{q} \cap \bar{q} \neq \phi$ then the records belonging to $\bar{q}$ can be arranged among themselves without affecting the CR property of the family Q_0. By rearranging the records belonging to $\bar{q}$ in the current CR organization for Q_0 such that records belonging to q are consecutive, we have a CR organization for Q_N.

If there exists a $\tilde{q} \in Q_0$, $\tilde{q}$ different from $\bar{q}$ and $\tilde{q} \cap \bar{q} \neq \phi$, we note that rearranging the records $\in (\tilde{q} \cap \bar{q})$ among themselves and records $\in (\bar{q} - \tilde{q})$ among themselves in the present CR organization for Q_0 do not affect the CR property of Q_0. Rearranging the records $\in (\bar{q} - \tilde{q})$ among themselves and records $\in (\tilde{q} \cap \bar{q})$ among themselves such that records $\in q$ are consecutive, we get a CR organization for Q_N.

Part (v): All we have to do is to prove that the records pertinent to queries in $\bar{Q}$ are consecutive in the CR organization for Q_0. Let R_1 and R_2 be some partitions of records $\in \bigcup_{q_i \in \bar{Q}} \{q_i\}$.

For the sake of proof, assume to the contrary that the CR organization for Q_0 is ------ $R_1 \bar{R} R_2$ ------ where $\bar{R}$ is a non-empty set of records, i.e. assume that the records $\in \bigcup_{q_i \in \bar{Q}} \{q_i\}$ are not consecutive in the CR organization for Q_0.

Then there is no query $q_i \in \bar{Q}$ such that $q_i \supseteq \{r_1, r_2\}$ where $r_1 \in R_1$ and $r_2 \in R_2$. Thus for $\forall q_i \in \bar{Q}$, either q_i contains records $\in R_1$ or records $\in R_2$. This means that in $\Omega(Q)$ there is no path between a node corresponding to a query $\in \bar{Q}$ that has pertinent records belonging only to R_1 and a node corresponding to a query $\in \bar{Q}$ that has relevant records belonging only to R_2. This contradicts the hypothesis that $\Omega(\bar{Q})$ is connected. Now that we have proved that the records relevant to queries $\in \bar{Q}$ are consecutive in the CR organization for Q_0, it is obvious that the same organization is a CR organization for Q_N.

Part (vi): From the proof of part (v), we observe that records $\in R_1$ must be consecutive in the CR organization for Q_0. So do the records $\in R_2$. Let the CR organization for Q_0 be ---- R_1 R R_2 ----. If $R = \phi$ then the records that are relevant to q are consecutive in this organization which then is a CR organization for Q_N. If $R \neq \phi$, arrange the records such that the new organization is ---- R_1 R_2 R ----. Since $\Omega(Q_1)$ is a component, there is no query $\bar{q} \in Q_0$ such that $\bar{q} \cap R_1 \neq \phi$ and $\bar{q} \cap (F-R_1) \neq \phi$ (Recall that $F = \bigcup_{q_i \in Q_0} \{q_i\}$.) Similarly, there does exist a $\bar{q} \in Q_0$ having $\bar{q} \cap R_2 \neq \phi$ and $\bar{q} \cap (F-R_2) \neq \phi$. Then the new organization is a CR organization for Q_0. But in the new organization, records $\in R_1$ and R_2 which are precisely the records that are relevant to q are consecutive. Hence the new organization is a CR organization for Q_N.

Part (vii): This is just a generalization of part (vi). By repeated application of the procedure outlined in the proof of part (vi), we obtain a CR organization for Q_N. QED ****

We considered in the above theorem some special cases of adding a new query to the system. There is no elegant solution for the general case. However, a few lines to reduce the amount of work to be done may be worth mentioning. Let $G_1, G_2, \ldots, G_m$ be the complete subgraphs of $\Omega(Q_0)$ that were chosen to cover $\Omega(Q_0)$ in the application of algorithm (A1). For $1 \leq i \leq m$, let Q_i be so that $\Omega(Q_i) = G_i$ and $\bar{G}(Q_i)$ be the DSG of Q_i. Let H_0 be the family of sets $\{H_0^1, H_0^2, \ldots, H_0^k\}$ where H_0^i for $i = 1, 2, \ldots, k$ is a set of pair wise consistent Hamiltonian paths in $\bar{G}(Q_1), \bar{G}(Q_2), \ldots, \bar{G}(Q_m)$. In $\Omega(Q_N)$, the node q corresponding to the query q being added and the edges that are incident on q are not in any of the subgraphs $G_1, G_2, \ldots, G_m$. Let $G_U^1, G_U^2, \ldots, G_U^p$ be a set of complete subgraphs of $\Omega(Q_N)$, distinct from $G_1, G_2, \ldots, G_m$, such that $G_1, G_2, \ldots, G_m$, $G_U^1, G_U^2, \ldots, G_U^p$ cover $\Omega(Q_N)$. Q_U^i be such that $\Omega(Q_U^i) = G_U^i$ for

$1 \leq i \leq p$ and $\bar{G}(Q_U^i)$ denotes as usual the DSG of Q_U^i. We now obtain H_U, the family of sets of pair wise consistent Hamiltonian paths in the directed semantic graphs $\bar{G}(Q_U^i)$. Now that we have H_O and H_U, it is easy to obtain H_N, the family of sets of pair wise consistent Hamiltonian paths in the directed semantic graphs of the subfamilies corresponding to some complete subgraphs covering $\Omega(Q_N)$. If $H_N \neq \phi$, we proceed by choosing any member of H_N, defining the P.O. graph $\tilde{G}(Q_N)$ and applying the rest of the algorithm (A1). Thus, we see that if at every stage we keep a family of sets of consistent Hamiltonian paths, the problem is a little simplified when we add a new query to the system.

Let us now consider the addition of a set of new records to a CR file organization. When a set of new records is added to a CR organization and replies to one or more queries in Q_O are updated, we essentially have a new family of queries Q_N and a new file. We may reorganize the new file w.r.t. the new set of queries and obtain a CR file organization from scratch. Apart from this trivial and inefficient solution, we may consider some efficient methods that follow.

If we can anticipate the semantics of the new records, we can include dummy records as pertinent to queries that may have some of the new records as reply records. When the actual records are added to the system, we simply replace the dummy records by the actual ones. Till the addition of new records, the dummy records must be removed from the set of replies to all queries.

If it is difficult to anticipate the semantics of new records and if we can relax the requirements of a CR file organization for Q_N, there is an elegant method. Q_0 is the family of queries in the system. Let $R = \{r_1, r_2, \ldots, r_k\}$ be the set of new records added and Q_N the family of queries with updated set of replies. Now, define $Q_U = \{\bar{q}_1, \bar{q}_2, \ldots, \bar{q}_p\}$ such that for $\forall\, \bar{q}_i \in Q_U$, $\bar{q}_i \subseteq R$ and for $\forall q \in Q_N$, $q \cap R \neq \phi \Rightarrow \exists\, q_j \in Q_0$ and $\bar{q}_i \in Q_U$ such that $q = q_j \cup \bar{q}_i$.

<u>Theorem 13</u>: If Q_U does not have a CR organization, then Q_N does not have a CR organization.

<u>Proof:</u> If Q_N has a CR organization, then we can delete the records not belonging to any $\bar{q}_i \in Q_U$ from the CR organization for Q_N. The resultant is a CR organization for Q_U. This is imminent from the fact established earlier in this chapter that deletion of a set of records from a CR organization does not affect the CR property of the family consisting of queries with updated set of reply records. Thus, Q_N has a CR organization implies that Q_U has a CR organization.

<u>Example 5.3</u>: Let $Q_0 = \{q_0, q_1, q_2, q_3\}$

$$\begin{aligned} \text{where } q_0 &= \{r_0, r_1, r_2\}, \\ q_1 &= \{r_1, r_2\}, \\ q_2 &= \{r_2, r_3\}, \\ \text{and } q_3 &= \{r_3, r_4\}. \end{aligned}$$

Let $R = \{r_5, r_6, r_7\}$ be the set of records added to the system.

Let $Q_N = \{q_0, \tilde{q}_1, \tilde{q}_2, \tilde{q}_3\}$ where $\tilde{q}_1 = q_1 \cup \bar{q}_1 = \{r_1, r_2\} \cup \{r_5\}$

$$\tilde{q}_2 = q_2 \cup \bar{q}_2 = \{r_2, r_3\} \cup \{r_5, r_7\}$$

and $\tilde{q}_3 = q_3 \cup \bar{q}_3 = \{r_3, r_4\} \cup \{r_6, r_7\}$

We note that $r_0\ r_1\ r_2\ r_3\ r_4$ is a CR organization for Q_0 and $r_0\ r_1\ r_2\ r_5\ r_3\ r_7\ r_4\ r_6$ is a CR organization for Q_N. Then, $Q_U = \{\bar{q}_1, \bar{q}_2, \bar{q}_3\}$ has a CR organization which is $r_5\ r_7\ r_6$. ****

Unfortunately, theorem 13 does not work the other way, i.e. if Q_U has a CR organization, then Q_N need not possess a CR organization.

Consider the example below:

Example 5.4: Let $Q_0 = \{q_1, q_2, q_3\}$

where $q_1 = \{p, q\}$

$q_2 = \{r, s\}$

$q_3 = \{t, u\}$.

Let $R = \{a, b, c, d\}$ and $Q_U = \{\bar{q}_1, \bar{q}_2, \bar{q}_3\}$ where $\bar{q}_1 = \{a, b\}$, $\bar{q}_2 = \{b, c\}$, $\bar{q}_3 = \{b, c, d\}$. Let $Q_N = \{\tilde{q}_1, \tilde{q}_2, \tilde{q}_3\}$ where $\tilde{q}_i = \bar{q}_i \cup q_i$ for $1 \leq i \leq 3$. pqrstu is a CR organization for Q_0 and abcd is a CR organization for Q_U. But Q_N does not have a CR organization. ****

The two examples above lead us to a general solution when (i) Q_N has CR organization, but we opt not to do a reorganization and obtain a CR organization for Q_N and (ii) Q_N does not have a CR organization, but Q_U has a CR organization. We form a CR organization for Q_U. Each query $q \in Q_N$ is $q_i \cup \bar{q}$ for some $q_i \in Q_0$ and $\bar{q}$ is either empty or some query $\in Q_U$. In order to obtain the relevant records for the query $q \in Q_N$, we obtain from the CR organization for Q_0 the records relevant to q_i and if $\bar{q} = \phi$, we stop. If $q \neq \phi$, as the second step, we retrieve records pertinent to $\bar{q}$ from the CR organization for Q_U. This method may be extended so that we can define several update families Q_{U_1}, Q_{U_2}, ... such that each query $q \in Q_N$ may be expressed as a union of one query $\in Q_0$ and zero or more queries each belonging to some update family Q_{U_i}.

CHAPTER 6

RAMIFICATIONS OF CONSECUTIVE RETRIEVAL FILE ORGANIZATION

In this chapter, we consider some file organizations (on linear storage media) which are offshoots of the Consecutive Retrieval File Organization (CRFO). All of them are interesting both theoretically and practically. We also indicate a few problems for future research.

In Chapter 2, we defined a Query Inverted File Organization (QIFO). It is a file organization in which records relevant to each query are stored in consecutive storage locations. We also saw that in a QIFO, we may have to store a record more than once (i.e. we may have to have redundant records). Let m' be the number of records in a QIFO and m the number of records in the file having QIFO. Then m' - m is called the redundancy of the QIFO. Note that our definition of redundancy implicitly assumes that all records are of the same length. This may not be adequate in some situations.

Example 6.1: Consider the QIFO in figure 3.2 (page 23). The redundancy of the organization is = 4 - 3 = 1.

Naturally, the question that arises in one's mind is that if Q does not possess a CRFO (i.e. Q does not have a QIFO with zero redundancy), what is the QIFO with minimum redundancy?

This problem is not solved here and is a nice problem to look into in the future. The problem may be broken into two parts: (i) What is the minimum redundancy of a QIFO of an arbitrary family of queries (ii) how to find such an organization. Dr. S. P. Ghosh establishes some bounds on redundancy in [15].

One obvious way to reduce redundancy in a QIFO for Q is to partition Q into k subfamilies, $k < m$, such that the subfamilies have CR organizations where m is the number of queries in Q. It is easily seen that any pair of queries have a CR organization and hence, such a k exists. Indeed $k \leq \left\lceil \frac{m}{2} \right\rceil$ where $\left\lceil \frac{m}{2} \right\rceil$ stands for the smallest integer greater than or equal to $\frac{m}{2}$. If the CR organizations for these subfamilies are CR_1, CR_2, ... and CR_k respectively, then $CR_1 \cdot CR_2 \cdot \ldots \cdot CR_k$ is a QIFO for Q. Suppose QI_i and QI_j are two QIFO with some (>0) records common at one of their ends. Then, QI_i and QI_j may be stored consecutively with common records at their ends stored only once. For example, if $r_0\ r_1\ r_2\ r_3\ r_1$ is a QI organization for Q_1 and $r_0\ r_4\ r_1\ r_5\ r_6$ is a QI organization for Q_2 then $r_6\ r_5\ r_1\ r_4\ r_0\ r_1\ r_2\ r_3\ r_1$ is a QIFO for $Q_1 \cup Q_2$.

In a query inverted file organization, we assumed that the storage space is cheap and we looked for a file organization with maximum retrieval efficiency and as less a storage (redundancy) as possible. We can also consider the converse problem: obtain a storage organization (on a linear storage medium), in which each record is stored only once (i.e. guaranteed minimum storage space) and as high a retrieval efficiency as possible. The problem may be formulated as possible:

(i) Let Q be a family of queries regarding a file F. That is, $F = \bigcup_{q_i \in Q} \{q_i\}$. Note that, as before, we let q_i denote both the query and set of replies to the query. All the queries in Q are equally likely. For each query, the file is accessed from one end of the organization.

(ii) There is no cost to reach the first relevant record of a query.

(iii) There is no cost after reading the last relevant record of a query.

(iv) It cost a dollar to skip a record

(v) Find an organization (mapping or permutation) of F in the linear storage medium (real line) in which each record is stored only once (mapping is a one to one function) such that the cost of retrieving is minimized.

Assumption (ii) is valid since we can do a fast wind to reach the first relevant record of a query even if all the queries are not equally likely. Assumption (iii) is also valid since we do not care what happens after retrieving all the relevant records of a query. Assumption (iv) treats all the records equally. In reality, the cost of skipping a record should be a function of the length of the record. Before we give a solution based on dynamic programming [16], let us give an example.

<u>Example 6.2</u>: Consider the family $Q = \{q_1, q_2, q_3, q_4\}$ where $q_1 = \{r_1, r_2, r_4\}$, $q_2 = \{r_2, r_3\}$, $q_3 = \{r_1, r_3\}$ and $q_4 = \{r_1, r_2, r_3, r_4\}$. Consider the following permutation π: $1 \to r_1$, $2 \to r_2$,

$3 \to r_3$ and $4 \to r_4$. In the arrangement $r_1\ r_2\ r_3\ r_4$, when we retrieve replies for q_1, it costs a dollar (since we skip the record r_3), for q_2 is costs nothing, for q_3 it costs a dollar and for q_4 it costs nothing. Hence the total cost is two dollars. Now, consider the permutation, $1 \to r_1$, $2 \to r_3$, $3 \to r_2$ and $4 \to r_4$. In this arrangement ($r_1\ r_3\ r_2\ r_4$) the total cost is one dollar. We know that this is the best we can do. (Q does not possess the CR property. $\Omega(Q)$ is complete and $\bigcap_{q_i \in Q} q_i = \phi$. By Lemma 3, Q is not consecutively retrievable.)

We can give this problem a matrix representation. Let M be a zero - one matrix, whose m columns correspond to queries, n rows to records and $M_{ij} = 1$ iff record r_i is relevant to query q_j and $M_{ij} = 0$ otherwise. The cost of the matrix M = $\sum_{1 \le j \le m}$ Number of zeros between the top and bottom "ones" in column j. The problem then is to find a permutation of rows such that the total cost is minimum.

<u>Example 6.3</u>: Consider the same query family in example 6.2. The matrix M representing Q is given below for the arrangement $r_1\ r_2\ r_3\ r_4$.

$$M = \begin{array}{c|cccc} & q_1 & q_2 & q_3 & q_4 \\ \hline r_1 & 1 & 0 & 1 & 1 \\ r_2 & 1 & 1 & 0 & 1 \\ r_3 & 0 & 1 & 1 & 1 \\ r_4 & 1 & 0 & 0 & 1 \end{array}$$

$$\text{Cost of } M = \sum_j \text{cost of column } j$$

$$= 1 + 0 + 1 + 0 = \underline{\underline{2}}$$

(Cost of column 1 = number of zeros between the first (M_{11}) and last (M_{41}) "one" in column one.)

$$M' = \begin{array}{c|cccc} & q_1 & q_2 & q_3 & q_4 \\ \hline r_1 & 1 & 0 & 1 & 1 \\ r_3 & 0 & 1 & 1 & 1 \\ r_2 & 1 & 1 & 0 & 1 \\ r_4 & 1 & 0 & 0 & 1 \end{array}$$

M' is the same as M except that rows two and three are interchanged. Cost of M' = 1. We proved in example 6.2 that this cost was minimum.

We now give a dynamic programming solution to this problem and analyze the complexity of the solution.

Let B be a subset of rows in the matrix M. OR(B) denotes the inclusive OR of the row vectors in B. For example if 1 0 1 0 and 1 1 0 1 are the two row vectors in B then OR(B) is the vector 1 1 1 1. OR ($\{\phi\}$) = 0 where $\{\phi\}$ is the empty set. If $a = (a_0, a_1, \ldots, a_i, \ldots, a_n)$ and $b = (b_0, b_1, \ldots, b_i, \ldots, b_n)$ are two row vectors of zeros and ones, then $a * b$ is the row vector $(a_0 \times b_0, \ldots., a_i \times b_i, \ldots., a_n \times b_n)$. $|a| = \sum_{i=0,n} a_i =$ number of "ones" in the vector a. $\bar{a}$ is the complement of a which is $(\bar{a}_0, \ldots, \bar{a}_i, \ldots, \bar{a}_n)$ where $\bar{a}_i = 1$ if $a_i = 0$ and $\bar{a}_i = 0$ if

$a_i = 1$. If A denotes a set of rows in M, then A-B denotes the set of rows in A and not in B.

Define: $f(r_i; B)$ is the minimum cost of placing row r_i in M immediately after the rows in the set B.

$$f(r_i; B) = \min_{r_j \in B} [f(r_j; B-\{r_j\})] + d_{B,\, r_i} \text{ where}$$

$$d_{B,r_i} = |\, OR(B) * \bar{r}_i * OR(F-B \cup \{r_i\}) \,|$$

in which F denotes the set of rows in M. Boundary condition is $f(r_i; \{\phi\}) = 0$. The minimum cost of M is $\min_{r_i} f(r_i; F-\{r_i\})$.

As a by-product of this calculation, we obtain a minimum cost permutation.

<u>Example 6.4</u>: For example, let us calculate $f(r_4; \{r_1,r_2,r_3\})$ for the example 6.3..

$$d_{\{r_1,r_2,r_3\},\, r_4} = |\, OR\,(\{r_1,r_2,r_3\}) * \bar{r}_4 * OR\,(\{\phi\}) \,| = 0$$

$$d_{\{r_1,r_3\},\, r_2} = |\, OR\,(\{r_1,r_3\}) * \bar{r}_2 * OR\,(r_4) \,|$$

$$= |\, 1\,1\,1\,1 * 0\,0\,1\,0 * 1\,0\,0\,1 \,|$$

$$= |\, 0\,0\,0\,0 \,| = 0$$

$$d_{\{r_3\},\, r_1} = |\, OR\,(\{r_3\}) * \bar{r}_1 * OR\,(\{r_2,r_4\}) \,|$$

$$= |\, 0\,1\,0\,0 \,| = 1$$

Similarly, $d_{r_1,\, \{r_3\}} = 1$

$$f(r_1;\{r_3\}) = f(r_3;\{\phi\}) + d_{\{r_3\},\ r_1} = 0 + 1 = 1$$

$$f(r_3,\{r_1\}) = f(r_1;\{\phi\}) + d_{\{r_1\},\ r_3} = 0 + 1 = 1$$

$$f(r_2;\{r_1,r_3\}) = \text{Min}\begin{cases} f(r_1;\{r_3\}) + d_{\{r_1,r_3\},\ r_2} \\ f(r_3;\{r_1\}) + d_{\{r_1,r_3\},\ r_2} \end{cases}$$

$$= 1 + 0 = 1$$

Similarly

$$f(r_1;\{r_2,r_3\}) = 1$$

$$f(r_3;\{r_1,r_2\}) = 2$$

$$f(r_4;\{r_1,r_2,r_3\}) = \text{Min}\begin{cases} f(r_1;\{r_2,r_3\}) + d_{\{r_1,r_2,r_3\},\ r_4} \\ f(r_2;\{r_3,r_1\}) + \quad '' \\ f(r_3;\{r_1,r_2\}) + \quad '' \end{cases}$$

$$= 1. \qquad ****$$

Now, let us look into the complexity of our algorithm. The answer is $\underset{r_i}{\text{Min}}\ f(r_i;F-\{r_i\})$. There are n choices for r_i where n is the number of rows of M(n is the number of records in the file). Consider the computational step $f(r_j;B)$. There are (n-1) choices for r_j. If the number of row vectors in B is p, then the number of ways of choosing B is $\binom{n-2}{p}$. OR(B) involves "oring" p vectors, OR(F-BU$\{r_j\}$) involves "oring" n - p - 1 vectors. There are two row multiplications. Hence the number of operations in d_{B,r_i} is (n+1). Therefore, the total number of

$$\text{operations} = n \cdot \sum_{p=0}^{n-2} (n-1) \cdot \binom{n-2}{p} \cdot (n+1)$$

which is $\sim n^3 \cdot \sum_{0}^{(n-2)} \binom{n-2}{p}$ where $\sim$ stands for "of the order of".

$$= n^3 \cdot \sum_{0}^{(n-2)} \frac{(n-2)\,!}{(n-2-p)\,!\,\cdot p!}$$

$$= n^3 \cdot \sum_{p=0}^{m} \frac{m!}{(m-p)!\,\cdot p!} = n^3 \cdot (1+1)^m = n^3 \cdot 2^{n-2} \sim \underline{\underline{2^n}}$$

The solution has complexity of the order of 2^n which is exponential. I strongly suspect that this problem is polynomial complete [17]. If this conjecture is true, it means that if we can solve this problem in polynomial time then we can solve in polynomial time several problems which are strongly suspected to be unsolvable in polynomial time (like the traveling salesman problem [18], [21], the satisfiability problem [17], etc.). Proving that the conjecture is true or false is a good problem for future research.

Another interesting problem is to assign weights or probabilities to the queries in Q and find CRFO and QIFO with maximum retrieval efficiency and minimum redundancy.

Consider the following problem. Let R = {1,2,...,n}. Find a shortest length sequence of elements belonging to R containing every k out of n combinational subsequences. A combinational subsequence is a sequence of consecutive locations in which we ignore the ordering of elements. For example, 1231 is a shortest length sequence of {1,2,3} containing all the two-out-of three combinations which are {1,2}, {1,3} and {2,3}. This problem also seems (?) to be polynomial complete. If R corresponds to the set of records of a file and the family of queries Q regarding the file is $\{q_i \mid$ reply for q_i is k out of n records in the file$\}$, then solving the above problem is equivalent to finding a QIFO for Q with minimum redundancy (of course, assuming that the records are of the same length) [15]. In addition to QIFO, the above problem seems to have applications in coding theory, placement algorithm and information theory. Giving a solution to the problem that works in polynomial time or proving that the problem is polynomial complete is a good research topic.

CHAPTER 7

SUMMARY AND CONCLUSIONS

The work reported here was in the nature of a general optimization problem applied to storage organization techniques. We assumed that we knew in advance a family of queries Q regarding a file F. We wanted to store the file on a linear storage medium so that the retrieval efficiency was maximum and the amount of storage used was minimum. We analysed the four existing file organizations, namely, the sequential file organization, index sequential file organization, inverted file organization and multilist file organization. Among these, the inverted file organization possessed greater flexibility and higher efficiency of retrieval. Because of often unwanted generality, the efficiency of retrieval in an inverted file organization was not fast enough.

We introduced the notions of QIFO and CRFO. QIFO provided minimum overall retrieval time for all queries in Q and CRFO guaranteed minimum storage space in addition to minimum overall retrieval time. The CR property of a family of queries was generalized to the L.O. property of a family of sets. With this generalization, the model could be applied to many storage organizations of interest. Apart from storage organization in a computer, book organization in a library shelf, inventory organization in a warehouse, placement and maintenance of equipments, switches, gadgets, etc. are some of the several examples of the applications of the model.

We took a graph theoretic approach to investigate the problem. We got some results connecting the L.O. property of a family of sets and an interval graph. We obtained necessary and sufficient conditions for a family of sets to be linearly orderable. We gave an algorithm to find a linear ordering if one existed and analysed the complexity of the algorithm.

All these examinations were done on the assumption that the file was time invariant. In Chapter 5, we addressed ourselves to the problems of updating a file with CRFO and obtained some nice results. Several file organizations that have similar flavour to CRFO were exposed in Chapter 6. Some open problems were also cited.

The idea of CRFO is based on the assumption that we know in advance the set of queries concerning the particular file F. It is my belief that the different number of queries asked by different users regarding a file at different times are not too many. These queries may be obtained by the data management system by monitoring and audit logging the accesses to the file. Among the queries frequently asked by users regarding a file F, let Q denote the maximal subset having the CR property. We build a CR organization for Q. Whenever a query q not in Q is asked by a user, since the CRFO is also a sequential file organization (if we ignore the directory part), we can fetch the records relevant to q as if the file has the sequential organization. When a query $\in$ Q is asked, we access the file making use of the CR organization.

REFERENCES

[1] D. Hsiao and F. Harary, A Formal System for Information Retrieval from Files, Communications of ACM, 2, February, 1970.

[2] D. E. Knuth, The Art of Computer Programming, volume 1, Addison Wesley, Reading, Mass., 1968.

[3] D. K. Hsiao, A Generalized Record Organization, IEEE Transactions on Computers, 12, December 1971.

[4] W. C. McGee, Generalized File Processing, Annual Review in Automatic Programming, 5, 13, Pergamon Press, New York, 1969.

[5] S. P. Ghosh, File Organization: The Consecutive Retrieval Property, Communications of ACM, 9, September, 1972.

[6] S. P. Ghosh, File Organization: Consecutive Storage of Relevant Records on a Drum Type Storage, RJ 895, IBM Research Report, July, 1971.

[7] E. F. Codd, A Relational Model of Data for Large Shared Data Banks, Communications of ACM, 6, June, 1970.

[8] F. Harary, Graph Theory, Addison Wesley, Reading, Mass., 1969.

[9] P. C. Gilmore and A. J. Hoffman, A Characterization of Comparability Graphs and of Interval Graphs, Canadian Journal of Math., 16, 1964.

[10] A. Pnueli, A. Lampel and S. Even, Transitive Orientation of Graphs and Identification of Permutation Graphs, Canadian Journal of Math., 23, 1971.

[11] C. Berge, Theory of Graphs and its Applications, Wiley, New York, 1962.

[12] F. Harary, R. Z. Norman, and D. Cartwright, Structural Models: An Introduction to the Theory of Directed Graphs, Wiley, New York, 1965.

[13] D. R. Fulkerson and O. A. Gross, Incidence Matrices and Internal Graphs, Pacific Journal of Math., 15, 1965.

[14] R. Tarjan, Depth-First Search and Linear Graph Algorithms, SIAM Journal on Computing, 2, 1972.

[15] S. P. Ghosh, Consecutive Storage of Relevant Records with Redundancy, RJ 933, IBM Research Report.

[16] R. E. Bellman and S. E. Dreyfus, Applied Dynamic Programming, Princeton Univ. Press, 1962.

[17] R. M. Karp, Reducibility among Combinatorial Problems, Complexity of Computations, Plenum Press, 1972.

[18] M. Held and R. M. Karp, The Construction of Discrete Dynamic Programming Algorithms, IBM Systems Journal, 4, 2, 1965.

[19] K. Eswaran, A Graph Theoretic Approach to Linearly Orderable Sets, Electronics Research Laboratory, Univ. of California, Berkeley, research report, M 369, 1972.

[20] K. Eswaran, Faithful Representation of a Family of Sets by a Set of Intervals, to appear in the Journal of Information Sciences.

[21] V. R. Pratt, A n log n Algorithm to Distribute n records Optimally in a Sequentially Access File, Complexity of Computations, Plenum Press, 1972.