# HORIZONTAL DATA PARTITIONING IN DATABASE DESIGN

S.Ceri - M.Negri - G.Pelagatti

Istituto di Elettrotecnica ed Elettronica
Politecnico di Milano
P.zza Leonardo da Vinci,32
Milano, Italy

## Abstract

In this paper the problem of horizontally partitioning data on a set of resources is considered.

The main optimization parameter is the number of accesses performed by the application programs to different portions of data. The concepts which are required for the determination of relevant portions of data are defined and a methodology for determining the access parameters is proposed.

The formulation of the general partitioning problem in 3 specific application enviroments is shown: distributed database design, file partitioning on a primary and a secondary memory and data distribution on different devices.

It is shown that the analytical models of these specific applications use the parameters of the general formulation.

## 1. Introduction

The problem of optimally partitioning data over a set of facilities is faced in several cases in information system design /1,2/. Consider for example the design of a distributed information system /3-9/, or the allocation of data on different physical areas or devices /10-13/, or the partitioning of a file into a primary and a secondary memory /14-15/: in all these cases, data must be partitioned into subsets and the subsets are associated with the nodes, physical areas, devices or files respectively. Despite the seemingly very different application environments, all these cases share a common problem consisting in the determination of relevant portions of data and of how these portions are accessed by the applications running in the system. In all these application environments, the use of a resource depends on the data which are managed by the resource; therefore the determination of the access pattern of the applications is required. The importance of this problem will encrease with the current technological trend towards distributed systems.

The determination of the access pattern of the applications to different portions of data is not an easy task, because it requires to understand both the logical and the statistical properties of the data and of the applications; this problem is therefore interdisciplinary in nature. Curiously, not much research has been devoted to this problem. For instance, several works exist on file allocation in distributed databases /3-9/, but few of them deal with file fragmentation /9,1o/, and none of them with the characterization

© 1982 ACM 0-89791-073-7/82/006/0128 $00.75

of access patterns to possible fragments.

In this paper, the set of data to be partitioned is considered to be a file consisting of homogeneous records. The basic assumption of this paper is that the probability that a record is accessed in a given time period depends on some property of the record itself and therefore all the records which have the same properties have also the same probability to be accessed. The determination of which properties of the records are relevant with respect to its probability to be accessed is therefore an important aspect of the data partitioning problem.

Given the above assumption, section 2 of this paper defines the logical and statistical concepts which are needed in order to characterize the access pattern of transactions to data; section 3 shows how these concepts can be applied from a methodological viewpoint and presents a result which allows to reduce the number of required parameters; section 4 shows how the parameters which characterize the access pattern to the records are used in the formulation of the optimal partitioning problem for a set of design environments.

## 2. Basic definitions

### 2.1 Predicates

A file F is a set of homogeneous records; records are subdivided into attributes. A predicate $x_i$ is a boolean function defined on F; each predicate can be evaluated on each record. The subset of the records of F for which $x_i$ is true is represented by $\{x_i\}$. Therefore:

$\{x_i\} = \{ o \mid o$ is a record of $F \wedge x_i(o) \}$.

Let $n(x_i)$ be the cardinality of $\{x_i\}$; let also N be the total number of records in the file.

Let AS be the set of attributes of F, $A_j \in$ AS be an attribute of AS and $D_j$ be the domain on which $A_j$ is defined; then a simple predicate is expressed as:

$$x_i : A_j \underline{\text{in}}\ V_{ij},\ V_{ij} \subseteq D_j$$

where $A_j$ is the attribute to which $x_i$ is applied, and $V_{ij}$ is the subset of admissible values for the predicate.

### 2.2 Completeness and minimality of a set of predicates

Let X be a set of simple predicates $x_1, x_2, \ldots x_n$ defined on F. The set Y(X) of minterm predicates $y_1, y_2, \ldots y_m$ associated to X can be defined as follows:

$$y_i = \bigwedge_{x_j \in X} x_j^* \text{ ,where } x_j^* = x_j \text{ or } x_j^* = \neg x_j$$

i.e., each minterm is given by the conjunction of all the predicates of X, where each $x_i$ is either taken in the natural form or negated; clearly, $m = 2^n$. The subset of records of F for which $y_i$ is true is called a minterm fragment of F with respect to X.

A set X of simple predicates is said to be complete if and only if any two records belonging to the same minterm fragment can not be distinguished from the viewpoint of the number of accesses, i.e. the accesses to the records of the same minterm fragment are homogeneusly distributed.

The completeness of a set of simple predicates is a primitive concept in this paper and the following definitions are given assuming that the considered set X is complete.

First, it is possible to define when a complete set of predicates is also minimal. Informally, minimality requires to exclude from the set X those predicates which are not useful for the description of dishomogeneity of accesses to the records.

Let $a(y_i)$ represent the number of accesses to the records of a minterm fragment $\{y_i\}$ in a given time period; let also A be the total number of accesses to the records of F in the same time period.

A simple predicate $x_k \in X$ is relevant if there exist at least two minterm fragments whose expression differs only in the predicate $x_k$ itself (which appears in natural form in one of them, and negated in the other one), having a different ratio between the accesses to the minterm fragment and its cardinality; therefore:

$x_k$ is relevant <--> $\exists y_i, y_j \in Y(X)$ |
$a(y_i)/n(y_i) \neq a(y_j)/n(y_j) \wedge (y_i \vee x_k = y_j \vee \neg x_k)$

A complete set X of predicates is minimal if and only if all its elements are relevant.

Let X be a minimal complete set of predicates, with m=|X|; any boolean expression B of them is itself a predicate, and {B} represents the subset of records for which B is true, called a candidate fragment of F. The number of different candidate fragments that can be defined is $2^m$ (two fragments are different if they are different sets; equivalent boolean expressions represent therefore the same fragment).

The general problem of characterizing the dishomogeneity of accesses of transactions to the records of a file is now properly formulated as the problem of determining the accesses of transactions to any of its candidate fragments. This problem is equivalent to the determination of accesses to minterm fragments, because, once that accesses to each minterm fragment are known, it is possible to determine the accesses to any candidate fragment by expressing the corresponding boolean expression as the disjunction of minterm predicates, and then summing the accesses which are made to the corresponding minterm fragments.

## 2.3 Implications between predicates

Implications between predicates are relevant to the determination of the accesses to candidate fragments. Implications exist not only between simple predicates but also between general boolean expressions of simple predicates; for instance:

DEPT='d1' $\wedge$ (STATUS='mgr' $\vee$ QUAL='Progr') ->
-> JOB='instructor' $\wedge$ SALARY > '50.000'

If an implication exists between two expressions $B^k$ and $B^l$, then they have to respect the following obvious rules on the number of accesses to the corresponding candidate fragments:

$$(B^k \rightarrow B^l) \rightarrow a(B^k \wedge B^l) = a(B^k)$$
$$a(B^k \vee B^l) = a(B^l)$$

The existence of an implication between two expressions has the consequence that some minterm fragments can be void. Let $mint(B^k)$ represent the set of minterm predicates which imply $B^k$ i.e. :

$$mint(B^k) = \{ y_i \mid y_i \rightarrow B^k).$$

Then, given an implication I: $B^k \rightarrow B^l$, all the minimal fragments expressed by the minterms belonging to the set:

$$V(I) = mint(B^k) \cap mint(\neg B^l)$$

are void. In fact, let $y_i \in V(I)$ be such a minterm. Suppose that a record $o \in \{y_i\}$ exists, then the expression $B^k$ is true and the expression $B^l$ is false for the record, thus contraddicting the implication.

Thus, the implications are used for determining and eliminating from the analysis those minterm fragments which are void.

## 2.4 Access probabilities

The statistical behaviour of the accesses to the records of F is described referring to the model of figure 1. The file F is represented as a set , whose elements are the records of F; the set is partitioned into disjoint minterm fragments, as resulting from the definition of a complete and minimal set of simple predicates, and the analysis of all the implications between them.

A second set, called request set R, represents all the accesses to records of F made by application programs in a given period of time.

Accesses in the request set are mapped to the corresponding records of the file F by a function f : R -> F; in fact, each access in the set R refers to a particular record of F, while the same record of r can be accessed several times, possibly by the same application program on different activations.

As a consequence of the distinction between data and access set, it is possible to define the statistical properties of the different sets. The following definitions are useful:

$p(x_i(o)) = \mathbf{n(x_i)/N}$ : probability that the predicate $x_i$ holds on an object which is taken randomly in F.

$p(x_i(f(r))) = a(x_i)/\mathbf{A}$ : probability that the predicate $x_i$ holds on the object associated to a request r taken randomly in R.

The probabilities $p(y_i(o))$ and $p(y_i(f(r)))$, related to minterm predicates, are similarly defined.

## 2.5 Statistical assumptions on the accesses to minterm fragments

In some cases the probability $p(y(f(r)))$ can be derived from the probabilities $p(x(f(r)))$, i.e. the probability that a minterm predicate y holds on a record can be derived from the probability that simple predicates hold on the same record. Of course, this is possible only if the access pattern of the application programs satisfy some statistical assumptions.

In the following, these assumptions are discussed and exemplified; 2 predicates $x_1$ and $x_2$ are considered, and the assumptions are formulated for their conjunction $x_1 \wedge x_2$.

**Assumption (a) Independence of predicates in the request set.**

This assumption requires that the predicates $x_1$ and $x_2$ are independent when observed on the request set; note that the independence of properties $x_1$ and $x_2$ in the data set is not required. This assumption is formulated as follows:

$$p(x_1 \wedge x_2(f(r))) = p(x_1(f(r))) \cdot p(x_2(f(r))) =$$
$$= a(x_1) \cdot a(x_2) \; / \; A^2.$$

**Assumption (b) The access set reflects the**

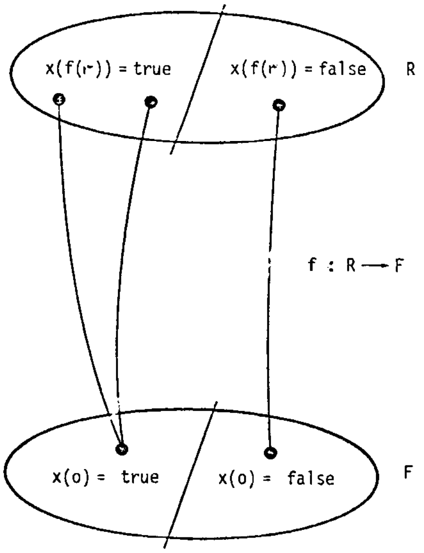
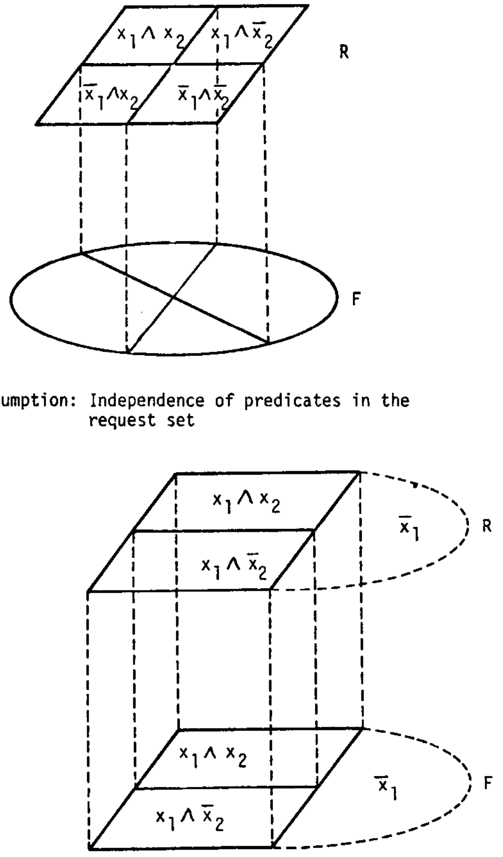


Fig. 1 - Reference model for the statistical analysis of access requests to file F



a) Assumption: Independence of predicates in the request set

b) Assumption: the request set reflects the data set

Fig. 2 - Graphical representation of statistical assumptions

**data set.**

This assumption requires the conservation, in the request set, of proportionalities existing between subsets of records in the data set. Let us take the predicate $x_1$ as a reference, and consider the subdivision of the subset of accesses to records for which $x_1$ holds into two further subsets in which the minterm predicates: $x_1 \wedge x_2$, $x_1 \wedge \neg x_2$ hold respectively. Then, this assumption requires that these two subsets are in the same proportion as the corresponding minterm fragments in the data set. The assumption is formulated as follows:

$$p(x_1 \wedge x_2(f(r))) \;/\; p(x_1 \wedge \neg x_2(f(r))) = p(x_1 \wedge x_2(o)) \;/\; p(x_1 \wedge \neg x_2(o))$$

By applying simple algebric transformations, we obtain:

$$p(x_1 \wedge x_2(f(r))) = p(x_1(f(r)))\; p(x_2(o)/x_1(o)) = = (a(x_1)/A)(n(x_1 \wedge x_2)/\; n(x_1)),$$

where $p(x/y)$ is the well-known expression of the probability of event x conditioned by the event y.

Note that if the symmetrical assumption holds taking the predicate $x_2$ as a reference, then the following facts also hold:

1) the ratios between accesses to minterm fragments and their cardinality are equal, i.e.

$$p(y_i(f(r))) \;/\; p(y_j(f(r))) = = p(y_i(o)) \;/\; p(y_j(o))$$

2) neither $x_1$ nor $x_2$ are relevant according to the definition in section 2, and in fact accesses are homogeneously distributed in the data set.

A graphical representation of the two assumptions is shown in figure 2.

## 3. Methodological problems

The concepts which have been defined in the previous sections suggest to perform the following steps when determining the access pattern of applications to data:

1) Classification of transactions.
2) Specification of a complete and (possibly) minimal set of predicates.
3) Determination of implications between predicates.
4) Analysis of statistical assumptions.
5) Determination of accesses to minterm fragments.
6) Integration of access requirements of different classes.

The major problems which arise in performing the above steps are analysed in /21/; this paper adresses only a few of them.

In performing the design,care must be taken of the fact that relevant predicates cannot always be deduced from the text of transactions.

Consider for example a bank transaction consisting in the request of the current amount of the bank account of a customer. The transaction needs for the identification of the customer the specification of either (a) his name, or (b) his office number and account number. Assume that the bank has a distributed data base system, and that each office of the bank can issue the transaction, with a different frequency. In order to describe adequately the access properties of transaction activations from a particular office, say office 1, one of the relevant predicates is clearly:

x : bank_office_number = 1

In fact, while each customer can use any office for making the transaction, the probability that at office 1 the transaction is issued by a customer whose bank_account_number is also 1 is very high, as each customer uses his office more often than any other office. In case (a) the predicate cannot be deduced by the text of the transaction, while case (b) uses the predicate explicitely as a selection criteria. In case (a), the designer should investigate on the general features of the transaction and characterize the common properties of the considered transaction activations, in order to determine the appropriate predicate.

Referring to step 5 of the design, the most obvious algorithm for determining the accesses to minterm predicates is the following: Let $X^k$ be a set of k simple predicates, $Y(X^k)$ be the associated set of minterms predicates, as defined in section 2.2. Let $x_{k+1}$ be a simple predicate which does not belong to $X^k$, and let $X^{k+1}=X^k \cup x_{k+1}$. The accesses to minterm fragments of all the sets $X^2$, $X^3$,....,$X^n$ can be progressively computed using the assumptions introduced in section 2.5; if no property applies,the number of accesses to the new minterm fragment $\{y_i \wedge x_{k+1}\}$ being considered is explicitely determined. This algorithm allows to determine the accesses to minterms, but its application is rather heavy; in order to complete, it is necessery to analyse $3^n$- 2n- 1 intermediate minterms. The computation of this number is given in /21/. It is possible, however, to follow a different algorithm, which consists in analysing for each set $X^k$ only the minterm $x_1 \wedge x_2 \wedge .... \wedge x_k$ instead of the complete set $Y(X^k)$; the determination of accesses to the other minterm fragments is given then by the solution of a linear system of equations. This algorithm requires the analysis of statistical assumptions only for $n^2 - n - 1$ pairs, and the solution of a system of $n^2$ equations, in the $n^2$ variables corresponding to accesses to minterms. The equations are of the form:

$$a(B^k) = \sum_{y_i \in mint(B^k)} a(y_i)$$

where $a(y_i)$ are unknown, and $a(B^k)$ are extimated; in facts, there are:

(1) the general equation expressing that the accesses to all minterms are A

(2) n equations expressing the accesses to simple predicates

(3) $2^n - n - 1$ equations, each expressing the accesses to the conjunctions of predicates in the natural form; each equation is produced by one of the above pairs.

In /21/ the proof is given that this algorithm requires to analyze $n^2$-n-1 pairs of predicates and the associated equations are linearly independent.

During step 6 of the design, in order to integrate the access requirements of different classes, all the simple predicates produced for each class are merged into one set S. The global minterm fragments $w_j$ that are generated using the simple predicates of S are a finer decomposition of F than those determined by the minimal fragments of each single class; the accesses to a minterm $y_i$ of a given class are homogeneously distributed on the global minterms $w_j$ such that $w_j$ --> $y_i$ ; therefore:

$$a(w_j)=a(y_i)\frac{n(w_j)}{n(y_i)}$$

## 4. Formulation of optimal partitioning problems for a set of design environments.

In this section, the parameters which characterize the access pattern of transactions are used in the formulation of the optimal partitioning problem for the following design environments: optimal fragmentation and allocation of a distributed database, horizontal partitioning of a file into a primary and a secondary memory, allocation of a file on several devices for assigning a given distribution of work load. The main goal of this section is not to provide and discuss complex optimization models, but to show that the problem formulations for these applications require exactly those concepts and parameters which have been introduced in the previous sections. The emphasis is therefore on the problem formulation, not on its solution.

### 4.1 Optimal fragmentation and allocation of a distributed database.

Recent researches on distributed databases /17-19/ have shown that it is generally convenient to partition a file into fragments, which are the smallest units of allocation. Clearly, it is not possible to solve the fragmentation problem independently before the allocation problem, because the optimal fragmentation can be determined only with respect to the optimal allocation of fragments.

The mixed fragmentation and allocation problem is treated using as goal function (to be minimized) the number of remote accesses of transactions to the records of the file; the criterion is sufficiently high level and general, does not require the extimation of many tedious parameters for modelling the distributed resources, and seems to be sufficiently effective in real cases.

The global minterm fragments introduced in section 3 are the proper units of allocation, since they contain records which are homogeneously accessed by all transactions; transactions must be divided into classes according to their activation nodes. Let $a(w_i/n_j)$ be the number of accesses to the global minterm fragment $w_i$ from the transactions which are activated at node $n_j$, and let $r_{ij}$ and $u_{ij}$ represent the fractions of these accesses which are retrievals and updates respectively, with $r_{ij} + u_{ij} = 1$.

Two distinct allocation problems are considered: the non-redundant problem, in which exactly one copy of each minterm fragment is allocated, and the redundant problem, in which at least one copy of each minterm fragment is allocated. In both cases, the final partitioning of the file into fragments is obtained by collecting in the same fragment all those minterm fragments whose resulting allocation is on the same node.

In the non-redundant problem, the minimization of the number of remote accesses of the transaction to the records of the file requires that each record be allocated on the node where it is accessed more often; therefore:

$$\{w_i\} \text{ is allocated on } n_j \quad \texttt{<-->}$$
$$a(w_i/n_j) = \max_{\text{all } n_k} \{a(w_i/n_k)\}$$

In the redundant problem it is necessary to distinguish between retrieval and update accesses, as updates have to be directed to all the copies, while retrievals are directed to only one copy (possibly, to a local copy) /4-9/. Then, the cost $c_{ij}$ of allocating a copy of the minterm $w_i$ on the node $n_j$ is given by the difference between the cost of updating this copy from all other nodes, and the benefit for retrieval transactions activated at node $n_j$, which found a local copy; it is:

$$c_{ij} = \sum_{\text{all } n_k \neq n_j} u_{ik} * a(w_i/n_k) - r_{ij} * a(w_i/n_j)$$

Then, a copy of the fragment is allocated on the node $n_j$ if the corresponding cost is non negative; when all costs are negative, the node with minimum cost is selected. It is:

$$\{w_i\} \text{ is allocated on } n_j \quad \texttt{<-->}$$
$$c_{ij} < 0 \vee c_{ij} = \min_{\text{all } n_k} \{c_{ik}\}$$

In /8/ it was proved on a similar formulation of the problem that the node with minimal cost is the same as the node given by the optimal non-redundant allocation.

### 4.2 Horizontal partitioning of a file into a primary and a secondary memory.

This problem is addressed in /13-14/ for the vertical partitioning (or segmentation) of a file; the approach consists in determining the mostly accessed projections of data to be stored on a primary (faster) memory. The same idea is applied here to the horizontal partitioning of the file into two fragments, corresponding to the more accessed records and the less accessed records.

The problem has the same formulation as a "knapsack" problem /20/, where a knapsack which can hold a maximum weight is packed, maximizing the total value of the contained goods; clearly, the maximum weight corresponds to the capacity C of the fast memory, and the value is maximized when most of the accesses are to records allocated in the fast memory. Again, global minterms introduced in section 3 are the proper units of allocation; the value $v_j$ associated to each minterm $w_j$ is given by the ratio $a(w_j)/n(w_j)$. The problem has a linear integer formulation by associating to each global minterm a binary variable $z_j$:

$$\max\left(\sum_{\text{all } j} z_j \; v_j\right)$$

under the constraint:

$$\sum_{\text{all } j} z_j \; n(w_j) <= C$$

The constraint of integrality can be relaxed by allowing the partitioning of a global fragment between the primary and the secondary memory, having $0 <= z_j <= 1$; in this case, the problem can be solved by inspection, ordering the minterm fragments by ascending values, and filling the primary memory with records taken ordinately from them.

### 4.3 Distribution of a file on devices in order to obtain a given distribution of work load.

A typical physical design problem is the allocation of a file on several devices aiming at obtaining a fixed distribution of work load on them (considering as a measure of work load the number of accesses to the device). Let $h(D_i)$ represent the fraction of work load which should be assigned to a given device $D_i$ of capacity $C_i$. The decision variables of the problem are the fractions $z_{ij}$ of each minterm fragment $w_j$ which are allocated on each device $D_i$, with $0 <= z_{ij} <= 1$. The goal function consists in the minimization of the difference between $h(D_i)$ and the effective work load assigned to $D_i$ because of the values assumed by the decision variables; the constraints impose that the capacity of each device is not exceeded, and that the global minterms are completely allocated. We have:

$$\min\left(\sum_{\text{all } i} \mid h(D_i) - \left(\sum_{\text{all } j} z_{ij} \; a(w_j) \;/\; A\right) \mid\right)$$

under the constraints:

(a) $\sum_{\text{all } j} z_{ij} \; n(w_j) <= C_i$ , for each i

(b) $\sum_{\text{all } i} z_{ij} = 1$ , for each j

Note that the simple problem of assigning an homogeneous work load on ND identical devices is solved by assigning the value (1/ND) to all decision variables, i.e. distributing homogeneously each global minterm on each device. In this case, the determination of accesses to global minterm fragments is not required for the solution of the problem.

## 5. Conclusions

In this paper, the data base design problem consisting in the horizontal partitioning of an homogeneous file has been considered. The concepts and the tools which are useful for the characterization of the access pattern of transactions have been defined, and a methodology for determining the access parameters has been proposed. The optimal partitioning problem has been formulated for several application environments, showing that the solution models require exactly the introduced concepts and parameters.

This work was supported by the National Research Council of Italy, Progetto Finalizzato Informatica.

## References

1. V.Y.Lum et al: "1978 New Orleans Database Design Workshop Report", Proc. Fifth Int. Conf. on Very Large Data Bases, Rio de Janeiro, Oct. 1979.

2. S. B. Navathe: "Logical Database Design", Panel on Logical Database Design,

Sec. 5, Florida, 1980.

3. W. W. Chu: "Optimal File Allocation in a Multiple Computer System", IEEE - TC, vol. C - 18, no. 10, 1969.

4. K. P. Eswaran: "Placement of Records in a File and File Allocation in a Computer Network", Proc. IFIP Congress , North Holland, 1974.

5. S. Mahmoud, J. S. Riordon: "Optimal Allocation of Resources in Distributed Information Networks", ACM - TODS , vol. 1, no. 1, 1976.

6. H. L. Morgan, J. D. Levin: "Optimal Program and Data Location in Computer Networks", CACM , vol. 20, no. 5, 1977.

7. P. P. S. Chen, J. Akoka: "Optimal Design of Distributed Information Systems", IEEE - TSE, vol. SE-6, no. 12, Dec. 1980.

8. S. Ceri, G. Martella, G. Pelagatti: "Optimal File Allocation on a Network of Minicomputers", Proc. Int. Conf. on Databases , Heyden Pub., Aberdeen, July 1980.

9. S. Ceri, S. B. Navathe, G. Wiederhold: "Optimal Design of Distributed Databases", Working paper, Stanford University, 1981 (Submitted for publication).

10. J. A. Hoffer: "An Integer Programming Formulation of Computer Database Design Problems", Information Science , vol. 11, July 1976.

11. J. A. Hoffer, D. G. Severance: "The Use of Cluster Analysis in Physical Database Design", Proc. First Int. Conf. on Very Large Data Bases , Framingham, 1975.

12. M. Schkolnick: "A Clustering Algorithm for Hierarchical Structures", ACM - TODS , vol. 2, no. 1, March 1977.

13. S. T. March, D. G. Severance: "The Determination of Efficient Record Segmentation and Blocking Factors for Shared Data Files", ACM - TODS , vol. 2, no. 3, Sept. 1977.

14. M. J. Eisner, D. G. Severance: "Mathematical Techniques for Efficient Record Segmentation in Large Shared Databases", JACM , vol. 23, no. 4, October 1976.

15. M. Hammer, B. Niamir: "A Heuristic Approach to Attribute Partitioning", Proc. ACM - SIGMOD , 1979.

16. S. K. Chang, W. H. Cheng: "A Methodology for Structured Database Decomposition", IEEE - TSE , vol. SE-6, no. 2, Mar. 1980.

17. J. B. Rothnie et al: "Introduction to a System for Distributed Databases (SDD-1) ACM - TODS , vol. 5 no. 1, Mar. 1980.

18. B. G. Lindsay, P. G. Selinger et al: "Notes on Distributed Databases", RJ2571 (33471), IBM Res. Lab., San Jose, Jul. 1979.

19. S. Ceri, G.Pelagatti: "The Allocation of Operations in Distributed Database Access", IEEE - TC , Feb 1982.

20. D. J. Luenberger: "Introduction to Linear and Nonlinear Programming", Addison-Wesley, 1973.

21. S. Ceri, M. Negri, G. Pelagatti: "Problems in Horizontal data partitioning", Internal Report, IEEEPM n.82-5, April 1982.