Közlemények 30/1984 91-99

MAPPING TO STORAGE OF A NETWORK STRUCTURE

Dr. GY. MEZEI

National Technical Information Centre
and Library

The scope of this paper does not allow us to describe (see [1]) ideas about the whole mapping. So we are going to deal only with segments and area design. After the functional (see [2]) analysis of the normalized data model it is usual to have an entity-type directly transformed into a record-type, and relation-types into sets. Then for the sake of efficiency some of the recordtypes should be divided into disjoint (see [3]) segments (or groups according to the CODASYL DBTG's term) (see [4]), and in a next phase these segments should be melted together into areas.

Both these phases apply cluster analysis. Together with the former phase of designing a conceptual data model which forms homogenous clusters, these three phases may be seen as a three-level cluster analysis method. The first level is conceptual data model design the second is segments design and the third level is area design. In the following we will deal with the peculiarities of the second and third level. Similarly in the frames of another approach we can see segments and area design as a transformation of a starting cluster--structure into an object cluster-structure (see [5]).

1. PECULIARITIES OF THE TRANSFORMATION PROCESS

The transformation tries only to improve the chances of the physical DB design. That is why reducing space or improve availability and recovery (all of them are important performance measures) are not in the focus of this paper. But we concentrate to response time which is the most important factor concerning the users of an information system.

For this purpose as a performance measure the following formula (or a similar one) can be created

$$CF = \sum_{i=1}^{r} \left[S_i \sum_{k=1}^{m} (M(i,k) \quad K(i,k)) \cdot \delta_{ik} \right], \qquad \text{where}$$

$$\delta_{i,k} = \begin{cases} 1 & if \quad E_{SZ_k} \cap E_{f_i} \neq \emptyset \\ 0 & else \end{cases}, \qquad \text{where}$$

$F = \{f_1, f_2, \ldots, f_r\}$ is the set of functions managing the DB

and E_{f_i} (where $i=1,2,\ldots,r$) is the subset of the attributes belonging to the f_i-th function.

$SZ = \{SZ_1, SZ_2, \ldots, SZ_m\}$ is the set of the DB segments types and

SZ_k (where $k=1,2,\ldots,m$) is the group of attributes belonging to the k-th segment type.

$\exists [(SZ_k \cap SZ_\ell \neq \emptyset) \wedge (SZ_k \nsubseteq SZ_\ell) \wedge (SZ_\ell \nsubseteq SZ_k)]$, where $k \neq \ell$,

$SZ_k, SZ_\ell \in SZ$ and $\ell=1,2,\ldots,m$.

$K(i,k)$ is the access time of SZ_k on a mass storage device.

$M(i,k)$ is the time of data handling of SZ_k in the central memory.

S_i is the estimated relative frequency of f_i

There are several factors which affect response time.

Important ones:

- Type of access: it must be distinguished retrieval time and update time

- Mode of access: sequential, random etc.
- Query complexity
- Frequency of reference

File designers who know something about the expected factors may be able to design more effective file organizations. But more essential design factors to be aware of decisions which hardware and software products, specially DBMS-components are going to be used. Optimizing the function CF above (or a similar one) can be only if functions $M(i,k)$ and $K(i,k)$ are thoroughly known.
And in practice those formerly mentioned decisions sometimes are taken later than having started segments design. Furthermore unfortunately $M(i,k)$ and $K(i,k)$ functions can be estimated well only after having taken decisions on file structures in a later phase of the DB design. And just because of this nature of the mapping to storage of a network structure that is essentially a feedback-oriented task (see *Figure 1)* which there always must be enough opportunity for the correcting decisions of the data administrator in.

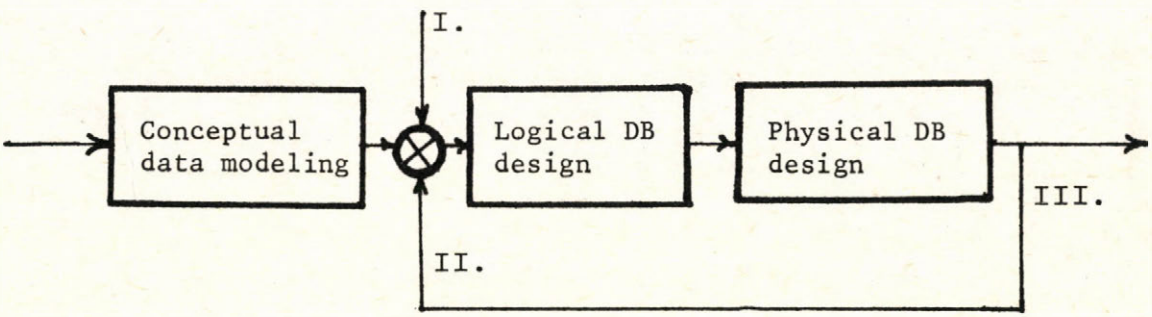

Figure 1

In the *Figure 1* two classes of entries can be seen.
Entry I: regeneration of the DB in as much as the information needs:
- set of functions (accesses)
- frequency of functions
- priority of functions

or the hardware /software environment varies to a great extent.

Entry II: recombination (correction) of the structure of the segments since the decisions of the file organisation was not foreseeble.

The II-III. loop represents an iterative segments recombination process. Instead of using the rough model of *Figure 1.* there is use in avoiding its repetitive steps by splitting the logical design process into three consecutive tasks:

- segments design
- area design
- impact of DBMS applied

and modify the model of the process (see *Figure 2.*)

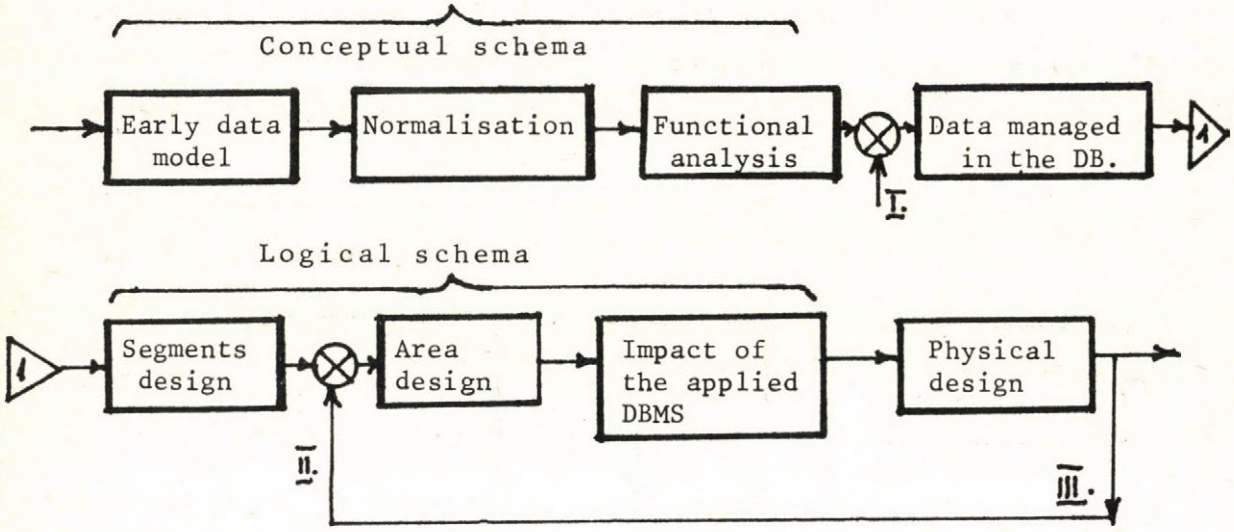

Figure 2.

If we apply that kind of segments-design method which is invariant (or nearly invariant) to small changes in the information needs of the organisation we may omitt segments-design from the loop.
So the data administrator has to frequently recombine the structure of the DB only at area level.
In the scope of 2. and 3. segments design and area design are discussed.

2. SEGMENTS DESIGN

2.1. THE STRUCTURE OF THE STARTING CLUSTERS

Before starting with segments we are given the product of the former process of data modeling. This product can be (see [6]) seen as a system of homogenous and generally overlapping clusters. The nucleus of such a cluster is the respective entity type and a cluster contains all the attributes related to that very entity type (that is why it is homogenous). The content and the number of these clusters is known.

2.2. CLUSTERING EFFICIENCY

In the case of a medium-size information system the number of the entity types is between 10-100 and that of the attributes (R) referring to one entity type is 10-30 (say R≈20), so the volume of the attributes altogether is generally some thousand, (say m=3000). It would be possible to cluster all the attributes together in one pass. But is well known that the bulk of the clustering methods is between $O(m \cdot \log m)$ and $O(m^2)$. So there seems to be use here in applying divide and conquer philosophy. It means that at a time the cluster analysis will be applied only for a homogenous subset of the attributes given by former data modeling (see 2.1).

By means of that simple trick the volume of attributes of larger systems and one pass clustering is transformed into a sequence of passes of clusterings dealing with moderate amounts of attributes. The number of the passes is equal to the number of entity types (i.e. the number of the formerly given clusters by data modeling).

2.3. THE FORESEEBLE STRUCTUREOF THE PRODUCED CLUSTERS

Segments design will produce an unforeseeble number of segmenttypes (clusters) the content of which is also unknown. Segmenttypes related to the same entity are disjoint ones. The segments (by definition) have an inner hierarchical structure. So in a clustering pass (see 2.2) we can make use of some kind of hierarchical clustering. Because of efficiency agglomerative methods seem to be advisable (see [8]).

Segmenttypes related to different entities might ovelap. This feature will be taken into consideration later only during area design (i.e. after having finished the sequence of the hierarchical dustering passes).

2.4. SELECTION OF THE HIERARCHICAL AGGLOMERATIVE CLUSTERING METHODS

2.4.1. Scale of variables

It determines to some extent the implementation of the hierarchical agglomerative clustering method. Stored similarity matrix approach is advisable, because of easy updating. Furthermore the DB managing functions (accesses) are seen as variables. The number of them is between N=100-1000. We concentrate only to the important ones, so $N \approx 100$. By weighting and standardising variables by the dmeanded estimated relative frequency of the accesses the scale of variables remain an interval one. But may be used subjective weighting as well and so the scale might become ordinal so stored similarity matrix is better (see [7]). This approach is effective when the number of the samples to be clustered (R) is less then the number of the (N) variables. That requirement is met in this case, since $R \approx 20$ and $n \approx 100$, so $R<N$ indeed. (see 2.2 as well).

2.4.2. Efficiency of the logical-physical design loop (see Figure 2)

In the class of hierarchical agglomerative clustering methods can be seen:
- linkage methods
- centroid methods
- variancia methods

Because for segments design such a method matches best which is invariant (or nearly invariant) to small changes in the information needs of the organisation, (see 1.) single-link methods are chosen (see [9]).
That subset of the single-link methods is preferable, which there is no need for cut-off level parameter and/or easy to program in.

2.4.3. Frequency of reference and mode of access

The relationship between the variables and the samples (here: attributes) reflects the frequency of reference for a sample (here: attribute).

To be frank sample is an unproper term here, because we can see attribute types here instead of samples. In a somewhat similar case of the leafs of the same tree where each occurence has got differences from the other ones probability based cluster analysis methods can be used. But in our case no such differences can be realized between the occurences of the same type. Furthermore the mode of access and the demanded subset of a particular attribute type in relation with a variable can be seen as a weighting factor of the type. Since the object-term matrix is essentially a non-binary one, the similarity coefficients (which are based on binary contigency tables) cannot be used here. This does not make good to the efficiency (space) of the segments design algorithm.

2.5. TAXONOMIC MEASURE

We can use only distance matrix with a distance measure which reflects asymmetry as well. The metric distance measure cannot reflect asymmetry so we choose a proper non-metric one. Because formely the Dice-coefficient was found as a proper similarity measure it comes handy Lance-Williams non-metric distance measure which is the inverse of the Dice-coefficient and can be used in the case of a non-binary object-term matrix (see [7]).

3. AREA DESIGN

3.1. THE STRUCTURE OF THE STARTING CLUSTERS

The structure of the starting clusters is written in 2.3. Useful additional information on conceptual data model is a list of those attributes which represent relations between two entity types.
It is important to know which segments contain these attributes. And from the point of view of implementation of the DB the distance measure components of these attributes should also be known. Furthermore necessary to be aware of the precedency (hierarchy) of the respective segment types when meeting the information requirement of each DB function.

3.2. THE FORESEEBLE STRUCTURE OF THE PRODUCED CLUSTERS

Area design is a necessary step in logical DB design, because (generally) none of the segment types can ensure the access of all the data required for a function of the organisation at a time. So between segment types either direct or indirect relations should be developed. Direct relations are developed first in the phase of area design. Are design will produce an unforeseeble number of area types (overlapping clusters). The content and the number of the elements (of these clusters) is also unknown.

3.3. THE TRANSFORMATION PROCESS

The transformation process should have the following features:

- harmonize clusters of objects (here: segment types) and clusters of variables at a time.
- easy to detect clusters by visualizing a display or hardcopy.
- quickly to select a representative subset of the segment types of each separate cluster.

Each representative subset determines an area. To meet the requirements above data-rearranging methods seen to be adequate.

LITERATURE

[1] Demetrovics, J., E. Knuth and P. Radó: Specification Meta Systems, IEEE 1982. May.

[2] Demetrovics, J., Gy. Gyepesi: Relációs adatmodell funkcionális függőségeinek általánosítása. MTA Alk. Mat. Lapok 6(1980) 313-322.

[3 CODASYL Systems Committee (1969): A Survey of Generalized DBMS May 1969. Report. New York.

[4] CODASYL systems Committee (1971): Feature Analysis of Generalized DBMS Technical Report. May 1971, New York - London - Amstardam

[5] Füstöss, A klaszteranalizis módszerei, MTA Szoc. Kut. Int. Módszertani füzetek (1977/1). Budapest.

[6] Halassy, B.: Adatmodellezés, adatbázis-tervezés, 1980. SZÁMOK, Budapest.

[7] Anderberg : Cluster Analysis for Applications
Academic Press Inc. New York - London, 1973.

[8] Van Rijsbergen: Information Retrieval. 1979.
Butterworths.

[9] Jardine - Sibson: Mathematical Taxonomy. 1971.
Wiley New York.

ÖSSZEFOGLALÁS

HÁLÓS ADATBÁZIS LEKÉPEZÉSE

Dr. Mezei Gyula

A cikk adatbázis szegmens- és area-tervezésével foglalkozik. A normalizált és funkcionálisan elemzett elvi adatmodell egyedtipusait rekordtipusokká, illetve részrekordokká /azaz szegmensekké/ képezik le, majd később e rekord /részrekord/ tipusokat nagyobb egységekbe /areak/ fogják össze. A cikk olyan módszert ismertet, ahol mindkét lépés során klaszternalizist használnak.

A szegmenstervezéshez agglomerativ hierarchikus klaszterálást és egy nem-metrikus távolságmértéket, az area- tervezéshez pedig táblázat-átrendező eljárást alkalmaznak.

ПРОЕКТИРОВАНИЕ СЕТЕВОЙ БАЗЫ ДАННЫХ

Д-р Дюла Мезеи

В статье рассматриваются вопросы проектирования сегментов и область /арэа/ базы данных. Отдельные типы нормализованной функциально проанализированной принципиальной модели данных преобразуются в рекорды либо в части рекордов /сегменты/, затем рекорды /части рекордов/ объединяются в большие группы /области/ /арэа/. Статья описывает также алгоритмы, которые используют методы кластерного анализа. Для построения сегментов использовались агглометративная иерархическая кластеризация и неметрическое измерение расстояний. При проектировании областей использовался метод перегруппировки таблиц.