International Journal of Management Science
Vol 7, No 1, May 2001

THE MULTI-PROJECTIVE MODEL: AN OBJECT-ORIENTED LOGICAL MODEL

Taeho Roh
Department of Industrial/Information Systems Engineering
Soongsil University, Seoul, 156-743, Korea

Insoo Choi
Department of Industrial/Information Systems Engineering
Soongsil University, Seoul, 156-743, Korea

(Received August 2000 ; Revised April 2001 ; Accepted April 2001)

ABSTRACT

The multi-projective model considers attributes and the relationships among attributes called projections. The critical features of the multi-projective model are the way of relating attributes in the description of the system, the way of reasoning incomplete projections, and the determination of connected patterns between projections. In order to get a full picture of the system, we build a set of projections. The multi-projective model can be thought of as projections of a multi-dimensional reality onto simplified "model space". The multi-projective database modeling approach used in this paper unifies the ideas and terminology of various database models. Most importantly, the multi-projective modeling is presented as a tool of database design in the relational and other database models.

1. INTRODUCTION

Underlying the structure of a database is the concept of a data model, a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints [1]. The various data models can be viewed or classified in several ways. The most useful classification scheme consists of the record-based logical model and the object-based logical model. Any data model can be exclusively categorized one way or the other. Recently, data models have tended to be object-based logical data models.

Record-based logical models are used in describing data at the conceptual and view levels, and are used both to specify the overall logical structure of a database and to provide a high level description of its implementation. These models require fixed format records of several types. Each record type defines a fixed number of fields, and each field is usually a fixed length. The representative of the class of record-based logical models is a relational model.

In contrast to record-based logical models, object-based models, which are also used in describing data at the conceptual and views levels, provide fairly flexible structuring capabilities and allow data constraints to be specified explicitly. Some of the more widely known ones are the object oriented model, the semantic data model, and the multidimensional model [1].

In this paper, we assume that the entity relationship (E-R) model can be defined as existing on a continuum between two categories, namely, the record-based logical model and the object-based logical model. This concept, as applicable to three categories, is illustrated below Figure 1.

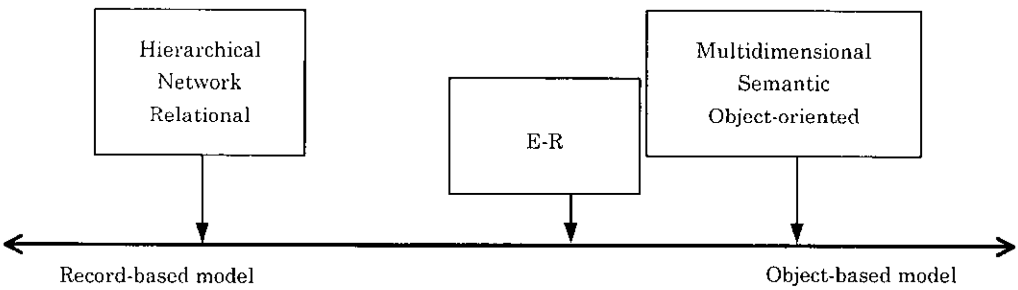


Figure 1. Level of Logical Models

Even though E-R models are based on a perception of a real world that consists of a collection of entities and relationships among these entities, an entity is distinguished from another entity by assigning values to a set of attributes. Alternatively, object-based logical models are based on a collection of objects. Unlike entities in E-R models, each object, for example, in the object-oriented model has its own unique identity independent of the values it contains. Consequently, two objects containing the same values are nevertheless distinct. In other words, object-based models are based purely on a perception of a real world that consists of a collection of objects or dimensions and relationships among them. Therefore, objects are distinct from each other in the database and the user will have a model of the 'universe' that the database is supposed to represent.

Despite the numerous amount of research in the area of database model design, to our knowledge there has been very little work that considers whether tuples in the database are incomplete [4]. In this paper, we address a new logical model, which is called the Multi-Projective Model. Figure 2 illustrates the characteristics of this new logical model among other logical models and the relationships between real world systems (or conceptual level) and real time systems (or functional level).

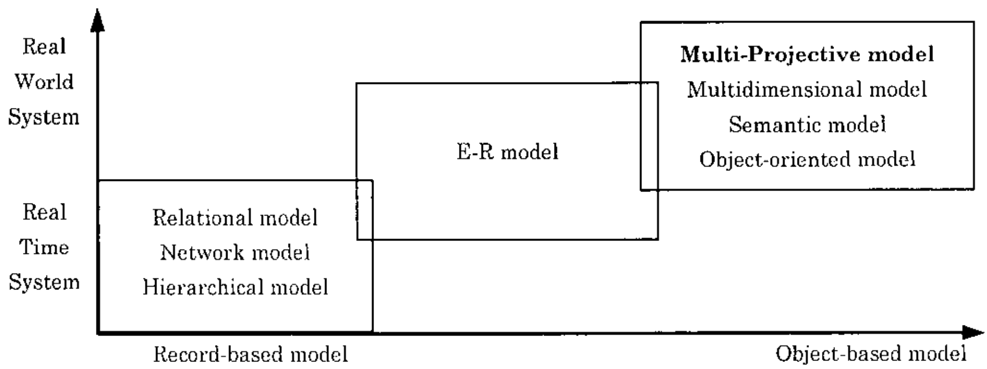
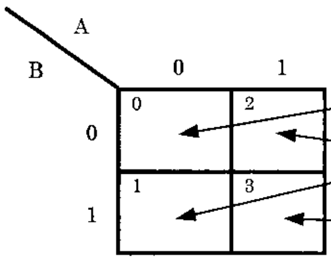


Figure 2. Characteristic of Multi-Projective Model

2. DEVELOPMENT OF PROJECTIVE MAPS

This section introduces concepts of projective maps (or simply P maps) which aim to development a new data model. Throughout this section the concepts and properties of the Karnaugh maps (or simply K map) [2] are briefly discussed as a refresher and in order to demonstrate their generalized application to P maps. K map is a geometrical figure which provides one region (box) for each row in a truth table. There is a one-to-one correspondence between truth table rows and potential minterms (complete products, see 2-1) or maxterms (complete sums, see 2-2). Also, there is a one-to-one correspondence between K map boxes and minterms and between such boxes and maxterms. A two-variable truth table and a two-variable K map are shown in Figure 3.

To identify the boxes, for example, Box 2 is not only labeled 2 but also located at intersection of $A = 1$ and $B = 0$, which are entries in row 2 in the columns for A and B, respectively.



Row No.	A	B	F(A, B)
0	0	0	
1	0	1	
2	1	0	
3	1	1	

Figure 3. A two-variable truth table and its K map

A particular function is defined as follow:

$$f(A,B) = A^cB^c + AB = m_0 + m_3 \qquad (2\text{-}1)$$

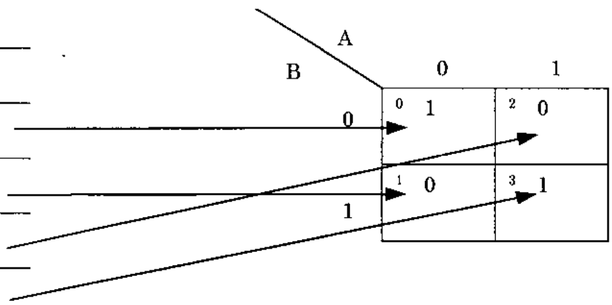
$$= (A + B^c)(A^c + B) = M_1 \cdot M_2 \qquad (2\text{-}2)$$

where A^c means the complement of A and B^c means the complement of B.

Then the function is readily defined in Figure 4.

Row No.	A	B	f(A, B)
0	0	0	1
1	0	1	0
2	1	0	0
3	1	1	1

(a)



(b)

Figure 4. (a) A truth table defining a function
(b) Truth-table definition represented on K map

The above function, which is expressed as a sum of minterms m_0 and m_3 is represented by 1s in the K map boxes 0 and 3; i.e., box 0 is associated with m_0 and box 3 with m_3. Similarly, since the function is expressible also as the product of maxterms M_1 and M_2, we find 0s in boxes 1 and 2. We shall now discuss a P map. Figure 5 shows a geometrical figure of the two-projection P map corresponding to the two-variable K map shown in Figure 3.

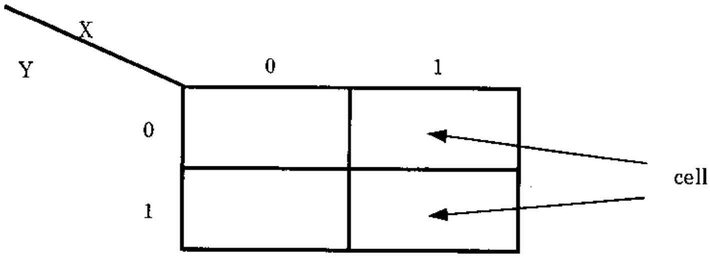


Figure 5. Two variables P map

The P map uses projections, which are viewpoints of the database designer or functional dependencies (or simply FDs) of the database, as input variables. The projection X or Y in Figure 5 corresponds to the input variable A or B in Figure 4, respectively. The cells on the P map correspond to boxes (regions) on K maps. The method of identifying cells follows the K maps' method. This, as we shall see, is an essential feature of the ordering. That is, the cells geometrically adjacent on the P map are also logically adjacent. While the minterms or maxterms are entered into boxes on the K map, attributes, which are the members of projections, are entered into cells on the P map. There is a clear difference between K-map and P-map. In K-map, every box may be assigned by minterm or maxterm. In P-map, however, the Cell No.0 (refer to Figure 13) does not need to be assigned by any attribute. In order to illustrate the P map, this section uses a relation in Figure 6 that will be decomposed to avoid logical inconsistencies, which might occur, from update operations later on.

Plant Name	Equipment Name	Plant Manager	Equipment Manufacturer	Manufacturer Address
Ethylene	Final cooler, Feed heater	Jim Smith	ABC exchanger	1247 Locust
Styrene	Feed pump	Bill Gunn	XYZ pumps	432 Broadway
Styrene	Feed heater	Bill Gunn	ABC exchanger	1247 Locust

Figure 6. An illustrated example

The relation is decomposed into following three relations (Relation 1, 2, 3) in 3^{rd} normal form [3].

Projection1 (P1): (PN → PM)

plant_name (PN) → plant_manager (PM)

Plant_name	Plant_manager
Ethylene	Jim Smith
Styrene	Bill Gunn

Figure 7. Relation 1

Projection2 (P2): (EM → MA)

equipment_manufacturer (EM) → manufacturer_address (MA)

Equipment_manufaturer	Manufacturer_address
ABC exchanger	1247 Locust
XYZ pumps	432 Broadway

Figure 8. Relation 2

Projection3 (P3): (PN EN → EM)

plant_name (PN), equipment_name (EN) → equipment_manufacturer (EM)

Plant_name	Equipment_name	Equipment_manufaturer
Ethylene	Final cooler	ABC exchanger
Ethylene	Feed heater	ABC exchanger
Styrene	Feed pump	XYZ pumps
Styrene	Feed heater	ABC exchanger

Figure 9. Relation 3

As we have noted, the attributes of the above projections are minterms of the K map. From this we can readily verify the result of the above projections in Figure 10.

- Attribute PN is in LHS (left hands side) of the arrow of projection 3 (P3) and projection 1 (P1).
- Attribute EN is in LHS of the arrow of projection 3 (P3).
- Attribute PM is in RHS (right hand side) of the arrow of projection 1 (P1).
- Attribute EM is in LHS of the arrow of projection 2 (P2) and in RHS (right

hand side) of the arrow of projection 3 (P3).

- Attribute MA is in RHS of the arrow of projection 2 (P2).

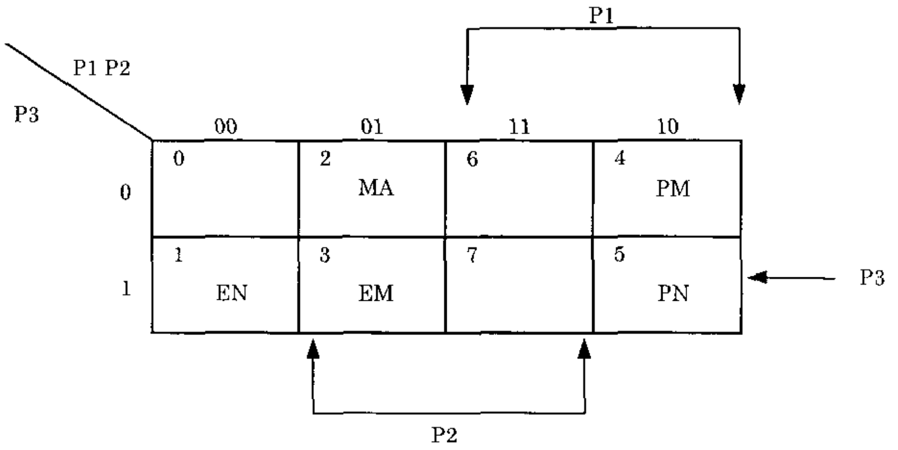


Figure 10. Example of the P map

Attributes PN, EN, PM, EM, and MA are values of each cell in the P map in Figure 10. Also, the sequence of cells follows the method of the K map. With the above results, we can represent the relationship between attributes and projections in Figure 11. In Figure 11, the letter 'L' means that the existence of an attribute on the LHS of the projection, 'R' means that the existence of an attribute on the RHS of the projection and '∅' means that an attribute does not exist on any side of the projection. Then the number '1' means that an attribute exists on any side of the projection and the number '0' means that no attribute exists on any side of the projection. Thus we can generate the binary representation ***g*** (P1, P2, P3) from the second column. And in the fourth column we have replaced this binary numbers by their decimal equivalents.

	P1 P2 P3	g (P1,P2,P3)	Cell no.
PN	L ∅ L	101	5
EN	∅ ∅ L	001	1
PM	R ∅ ∅	100	4
EM	∅ L R	011	3
MA	∅ R ∅	010	2

Figure 11. Relationship between attributes and projections

3. ADJOINING CELLS IN P MAPS

In this paper, we treated projections as viewpoints of the database designer or FDs of relations. Using database designers' common sense, they define projections as the groups of attributes that are formed to be relation schemata. In P maps, projections are used as input variables. However, we did not have any formal measure of why one grouping of attributes into a relation schema may be better than another. There was no measure of appropriateness or of the quality of the design except for the intuition of the designer. Before going on, we introduce a property of P maps. The essential feature of P maps is that the adjoining cells, horizontally or vertically (but not diagonally), is that the cells differ in only a single projection. This single projection will appear complemented in one cell and uncomplemented in the other.

To see the benefit of this feature, consider the attributes EN (C_1) and EM (C_3). These are going to adjoin horizontally in the P map of Figure 10. We have

$$EN = P1^c P2^c P3 \qquad \text{(cell no. 1 = 001)} \qquad (3\text{-}1)$$

$$EM = P1^c P2 P3 \qquad \text{(cell no. 3 = 011)} \qquad (3\text{-}2)$$

where, $P1^c$ and $P2^c$ of Eq. (3-1) are complement of P1 and P2, respectively.

These two attributes differ only in that the projection P2 appears complemented in EN and uncomplemented in EM. They can be combined to yield

$$P1^c P2^c P3 + P1^c P2P3 = P1^c P3(P2^c + P2) = P1^c P3 \qquad (3\text{-}3)$$

Thus the above two attributes, each involving three projections, have been replaced by a single 'term' involving two projections. The projection P2 that appeared complemented in EN and uncomplemented in EM has been eliminated. Another example is that attributes MA(C_2) and EM(C_3) are going to adjoin vertically in the P map of Figure 10. We have

$$MA = P1^c P2P3^c \qquad \text{(cell no. 2 = 010)} \qquad (3\text{-}4)$$

$$EM = P1^c P2P3 \qquad \text{(cell no. 3 = 011)} \qquad (3\text{-}5)$$

These two attributes differ only in that the projection P3 appears complemented in MA and uncomplemented in EM. They can be combined to yield

$$P1^cP2P3^c + P1^cP2P3 = P1^cP2(P3^c + P3) = P1^cP2 \quad (3\text{-}6)$$

Thus the above two attributes, each involving three projections, have been replaced by a single term involving two projections. The projection P3 that appeared complemented in MA and uncomplemented in EM has been eliminated. In the similar way we would obtain:

$$PM + PN = P1P2^cP3^c + P1P2^cP3 = P1P2^c \quad (3\text{-}7)$$

The great merit of the P map is that it permits easy recognition through geometric visualization of combinations of attributes that can be combined into simpler projections. A general principle, then, which applies to a P map is that any pair of adjoining attributes can be combined into a single term involving one projection fewer than do the attributes themselves. We have seen that two adjoin P map cells can be combined, yielding a term from which one projection has been eliminated. In a similar way, whenever 2^n cells adjoin, they can be combined to yield a single term from which n projections have been eliminated. So far, we have noted that attributes geometrically adjacent on a P map are also logically adjacent; i.e., the attributes differ in just a single projection. However, there are cases in which the cells are not geometrically adjacent but are nonetheless logically adjacent. As can be readily verified, each cell in the leftmost column is logically adjacent to the cell in the rightmost column on the same row. In Figure 10, the combination EN + PN yields

$$EN + PN = P1^cP2^cP3 + P1P2^cP3 \quad (3\text{-}8)$$

So that

$$EN + PN = P1^cP2^cP3 + P1P2^cP3 = P2^cP3(P1 + P1^c) = P2^cP3 \quad (3\text{-}9)$$

The result would have been the same, of course, if the grouping had been made in the order (PN + EN). Reading the group EN + PN in a direct manner would proceed as follows. We note that two attributes fall in the row corresponding to $P2 = 0$ and $P3 = 1$. Hence these projections will remain in complemented and uncomplemented form, respectively. However the attributes are found in columns corresponding to $P1 = 0$ and to $P1 = 1$. Hence this projection is eliminated. In other words, we can visualize a geometrical as well as a logical adjacency between the left and right column by imagining the P map wrapped around a vertical cylinder. We can visualize a geometrical as well as a logical adjacency between the top and bottom rows by imagining the P map wrapped around a horizontal cylinder. Both left and right column adjacencies and top and bottom row adjacencies can be visualized simultaneously by imagining the P map wrapped around a doughnut. We can note the following points:

1. The number of P map cells that are to be read as a group must be a power of 2. That is, we may read $2^0=1$, $2^1=2$, $2^2=4$, $2^3=8$, etc. We may not group three cells, for example, even if they are all adjacent.

2. Suppose we have a situation as in Figure 12. Here we have combined C_6 (means cell no. = 6) + C_{14} and C_{13} + C_{15}. May we now also combine these groups of two into a single group of four? The answer is "no", in spite of the fact that these two groups appear to be adjacent; for one group was formed by a horizontal combination while the other group was formed by a vertical combination. Hence the projection P1 eliminated from C_6 + C_{14} (=P2P3P4) is different from the projection P3 eliminated from C_{13} + C_{15} (=P1P2P4). Hence no further combining is possible [1].

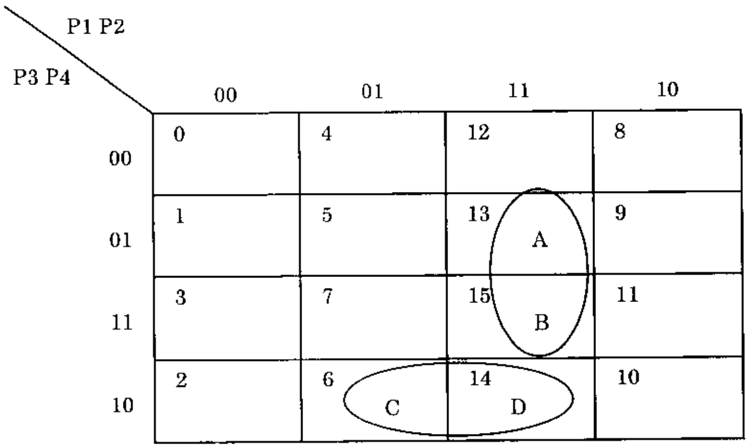


Figure 12. Example of physically but not logically adjacent four cells

4. DEGREE OF CELL IMPORTANCE IN P MAPS

In this section we want to investigate the importance of cells in P maps. In Figure

10, the four cells (Cell No. 4, 5, 6 and 7) in which P1 = 1 are bracketed are marked P1. We have correspondingly bracketed and marked four cells (Cell No. 2,3,6 and 7) in which P2 = 1 and the four cells (Cell No. 1, 3, 5 and 7) correspond to P3 = 1.

In order to talk about the "universe" that the database is supposed to represent, at least three projections are required in Figure 10. With Figure 10, we can reproduce Figure 13 as follows:

Projection	Cell Number							
	0	1	2	3	4	5	6	7
P1					⮟	⮟	⮟	⮟
P2			⮟	⮟			⮟	⮟
P3		⮟		⮟		⮟		⮟
Degree of Cell Importance	0	1	1	2	1	2	2	3

Figure 13. Degree of Cell Importance

In Figure 13, the sum of marked symbols (⮟), which means the degree of importance of each cell, is represented in the bottom row. If the projection 1 is used to see some part of the universe, we shall need four attributes that must be assigned to the four cells (Cell No. 4, 5, 6 and 7). If the projection 2 is used to see some part of the universe, we shall need four attributes that must be assigned to the four cells (Cell No. 2, 3, 6 and 7), and with the projection 3, four attributes will be needed which must be assigned to the four cells (Cell No. 1, 3, 5 and 7). In this paper, to understand the universe of our given example in Section 2, the database designers must have the three projections. In order to understand the universe with the projection 1, 2 and 3, the attribute that is assigned to Cell No. 7 should be considered first when database designers design a database. Cell No. 7 is a common cell to figure out the universe with any one of the three projections. We also defined the degree of importance of Cell No. 7 as three. It is natural that database designers, then, consider the Cells No. 3, 5 and 6 as less common cells than Cell No. 7. Because the degree of the three cells importance are two in Figure 13, it is argued that "universe" can overcome certain semantic difficulties with incomplete attributes in the database. Reasoning incomplete attributes of projections is one of the subjects of this paper. Unfortunately, the attribute of Cell No. 7 is undefined even though the attribute is more important than any other. In practice, such incomplete specification arises in two ways. Database designers do not care what attributes are assumed by projections. On other occasions, they may know certain attributes will never occur. In this case, we may pretend those attributes never occur, since the net effect is same. With incomplete projections, of which attributes are not assigned to the common cells, all projections should be required to understand universe. We now discuss the unknown attribute with our example of Figure 6 where the relation has three projections. With Figure 13, it is easily seen which attributes should be assigned to the common cells or have higher degrees of cell importance. Moreover, we know that an attribute of the Cell No.7 is extremely necessary. We can make the name of this attribute as "Specifications". To describe the algorithm of the multi-projective model developed here is as follows.

[Step 1] Allocates projections into the adequate P map.
[Step 2] Allocates attributes into cells of the P map.
[Step 3] If necessary, find out unknown attributes.

5. CONCLUDING REMARKS

We have proposed a formal way that projecting universe onto database with multi-projections. We believe that our approach has the potential of developing into an important design tool, but this approach calls for further analysis. First, our results clearly indicate some quantitative properties of optimal design. These properties provide explicit design rules which would guide database designers in performing this function. Second, using the representation of P maps, the database designers could optimize the design of databases. It is harmful to normalize ill-structured or incomplete projections. With the approach of the optimal and formal way, this paper let database designers design relations with correct structure. One of the purposes of this paper is to discover such an attribute that has to be assigned to the Cell No.7 in our example and is to design projections with precise structure. In summary, our contribution is providing a comprehensive, formal, and precise method in designing databases.

REFERENCES

[1] Henry F. Korth, Abraham Silberschatz, Database System Concepts. 2nd ed. McGraw-Hill Computer Science Series. 1991.

[2] Herbert Taub. Digital Circuits and Micro-Processors, McGraw-Hill (July, 1985), 57-81.

[3] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen. Object-oriented modeling and design, Prentice-Hall, Englewwod Cliffs, N.J., (1991), 371-372.

[4] Mark Levene, George Loizou. Database Design for Incomplete Relations. ACM Trans. Database Systems, 24,1 (Mar 1999), 80-126.