Entity-Relationship Approach to Systems Analysis and Design, P.P. Chen (ed.)
North-Holland Publishing Company, 1980
© Int. Conf. on ERA

DECOMPOSITION OF RELATIONS AND SYNTHESIS OF ENTITY-RELATIONSHIP DIAGRAMS

Michel A. Melkanoff* and Carlo Zaniolo†

The purpose of this paper is three-fold:

(a) to present a new decomposition algorithm to decompose a relation according to its functional and multivalued dependencies into well-defined primitives (atomic relations and functional dependencies) which preserve the original information,

(b) to describe the atomic relations and their functional dependencies through labelled graphs called Combined A-Z graphs which exhibit all the information in a succint a convenient diagram,

(c) to indicate how the combined A-Z graphs may be transformed into Chen's Entity-Relationship diagrams.

1. INTRODUCTION

Considerable effort, in database research, has been spent to develop a methodology for designing conceptual schemas satisfying the following requirements:

(1) They must be able to characterize completely and unambiguously the logical relationships existing among the data,

(2) They must be defined in a precise and formal fashion,

(3) They must be expressive and concise.

We will attack this problem using the formal framework of the Relational Data Model [CODD 70, DATE 77]. Requirement (1), which has been called the complete relatability requirement should, strictly speaking, include all the integrity constraints applicable to the data. However, the state-of-the-art regarding the formal analysis of integrity constraints is limited to functional dependencies (abbr. FD's) and multivalued dependencies (abbr. MD's) [FAGI 77, BEER 77, ZANI 79a]. These are capable of describing various relationships between entities and attributes in a database. In particular, they can describe the well-known one-to-one, one-to-many and many-to-many relationships.

We present here a new formal approach to the design of conceptual schemas which satisfies the first two requirements stated above. Furthermore we shall describe a graphical representation of the proposed conceptual schemas which satisfies the last two requirements. This graphical schemas can in turn be related in a natural fashion to Entity-Relationship (abbr. E-R) diagrams [CHEN 76]. The material which

* Computer Science Department, UCLA
Los Angeles, CA 90024
U.S.A.

† Sperry Research Center
Sudbury, MA 01776
U.S.A.

we present here is partly a condensed version of two papers, [ZANI 79a] and [ZANI 79b], where the mathematical justifications and numerous illustrations are given in detail.

The traditional definition of relational schemas is based upon the use of normal forms and relational keys. A usual design approach consists in starting with one or more "universal" relations describing the universe of interest. The desired schema is then obtained by decomposing the relations into various projections satisfying certain normal forms. However, the decomposition rules are not well defined and the resulting schemas often violate requirement (1) above.

In contradistinction, our approach consists in requiring that the complete relatability criterion (CRC) be satisfied at each step of the decomposition. This may be stated more formally by requiring that the relation R(W) only be decomposed into two projections, R[W1] and R[W2], if the following conditions are satisfied:

(a) There exists a non-trivial MD, X ->> Y, in R(W) such that W1 = X U Y and W2 = X U (W-Y);

(b) The FD's of R(W) can be derived from the FD's of R[W1] and R[W2];

(c) The MD's of R(W) can be derived from the combined FD's and MD's of R[W1] and R[W2].

Condition (a) above insures that every instance of R(W) be exactly reconstructable as the natural join of R[W1] and R[W2], thus preserving the content of the relation R(W).

Conditions (b) and (c) insure that the structure of the relation R(W) can be reconstructed from the FD's and MD's of the subrelations R[W1] and R[W2]. These conditions have been formalized in [ZANI 79a] and are described below as the complete relatability conditions, CRC1 and CRC2. Our decomposition process is designed to operate recursively until further decomposition is impossible due to the absence of nontrivial MD's within the resulting subrelations (these will be called atomic subrelations.) This decomposition process yields two sets of objects called the A-structure and the Z-structure. These can be described as follows:

i. The A-structure consists of the attribute sets of the atomic subrelations which may be joined to reconstruct the original relation.

ii. The Z-structure consists of elementary FD's which form a minimal cover for the FD's of the original relation (the definition of elementary FD is also given in the next section.)

These two structures are related by the so-called admissibility condition [ZANI 79a] which will also be discussed in the next section.

2. THE DECOMPOSITION ALGORITHM

The proposed decomposition algorithm is based upon the complete relatability conditions (CRC) which have been derived in [ZANI 79a] using the concepts of elementary FD's and multiple elementary MD's, defined as follows. An FD of R, X -> A, s called elementary if A is not an element of X and R contains no X' -> A where X' C X. The set of attributes {A} U X is called the scope of the elementary FD: X->A. An MD of R, X ->> Y, is called elementary if Y is non-empty and disjoint from X, and R does not contain another MD, say X' ->> Y', where X' C X and Y' C Y. An elementary MD of R is called multiple if R contains other elementary MD's with the same left side; otherwise it is called single.

A relation R(W) may be decomposed in a lossless fashion into two projection R[W1] and R[W2], where W1 U W2 = W, if for some multiple elementary MD, the following two

conditions hold:

$$\text{CRC1:} \quad (F - F_s) \subseteq (F1 \cup F2)^+ \tag{2.1}$$

$$\text{CRC2:} \quad G_m \subseteq (G_F \cup G11 \cup G22)^+ \tag{2.2}$$

where:

F is the set of elementary FD's of R(W),

F_s is the set of elementary FD's in F having W as scope,

F1 is the set of elementary FD's of R[W1],

F2 is the set of elementary FD's of R[W2],

G_m is the set of multiple elementary MD's of R(W),

G11 is the set of elementary MD's of R[W1] which have right side disjoint from W2,

G22 is the set of elementary MD's of R[W2] which have right side disjoint from W1,

G_F denotes the set of MD counterparts of F1 U F2 (the MD counterpart of a FD, X->Y, is the MD: X->>Y.)

F^+ denotes the set of FD's implied by the set of FD's, F. These can be constructed using the following inference rules of FD's: reflexivity, augmentation and transitivity [BEER 77].

G^+ denotes the set of MD's implied by the set of MD's, G. These can be obtained by using the following inference rules of MD's: reflexivity, augmentation, transitivity and complementation [BEER 77].

The decomposition algorithm can now be stated as follows:

Algorithm 2.1: DECOMPOSITION OF A RELATION R(W).

A1. Determine (a) F0, the set of elementary FD's of R,
(b) G0, the set of multiple elementary MD's of R,
(c) G_L, the set of multiple elementary MD's latent in R.

A2. Initialize the set variables ZCOVER and ACOVER to the empty set, and the integer variable L to 1.

A3. DECOMPOSE(W) [Invoke the procedure of Fig. 2.1]

A4. Print ZCOVER and ACOVER

G_L, the set of multiple elementary MD's latent in R [ZANI 79a] must be determined separately for, although these MD's do not hold for R(W), they hold for certain projections and may have to be included in recursive calls of the procedure DECOMPOSE.

Algorithms for determining elementary FD's, elementary MD's and multiple elementary MD's from sets of FD's or MD's and for verifying the CRC's are cited or given in [ZANI 79a], although the initial set of FD's and MD's must be extracted from the users, a task which is not always free of problems.

If the procedure DECOMPOSE completes successfully in all its recursive invocations, the algorithm terminates leaving the atomic relations labelled by the integer L in the set ACOVER; the set ZCOVER holds the FD's characterizing the atomic relations

```
procedure DECOMPOSE(W) comment a recursive procedure to decompose R(W);
  begin STEP1: DETERMINE (F, G_m);
        STEP2: FLAG <- false;
               for each X -> A ∈ F do
               if X U {A} = W then begin FLAG <- true;
                                         ZCOVER <- ZCOVER U { L:X -> A};
                                         F <- F - {X -> A};
                                   end STEP2;
               if G_m = ∅
               then STEP3: begin ACOVER <- ACOVER U {L:W};
                                 L <- L + 1
                           end STEP3
               else STEP4: begin NOTFOUND <- true;
                                 for each X ->> Y ∈ G_m while NOTFOUND do
                                 begin W1 <- X U Y; W2 <- W-Y;
                                       COMPUTE (F1, F2, G_F, G11, G22);
                                       if (F-Fs) ⊆ (F1 U F2)+
                                       G_m ⊆ (G_F U G11 U G22)+ then
                                       begin if FLAG then L <- L + 1;
                                             DECOMPOSE(W1);
                                             DECOMPOSE(W2);
                                             NOTFOUND <- false;
                                       end
                                 end;
                           if NOTFOUND then REPORTFAILURE
               end STEP4
  end DECOMPOSE;
```

Fig. 2.1. An ALGOL-like description for the recursive procedure DECOMPOSE. (Declarations are omitted. All variables but ZCOVER, ACOVER, L, F0, G0, G_L and W are local to this procedure).

and labelled by the appropriate values of L.

The sets ACOVER and ZCOVER generated by the algorithm satisfy the admissibility condition which can be stated as follows:

(1) If the ZCOVER contains an elementary FD with scope X, then it also includes every elementary FD of R with scope X. Moreover if R[X] is atomic then X is also contained in the ACOVER.

(2) If the ACOVER contains X, then every elementary FD of R[X] with scope X is also included in the ZCOVER.

It may be noticed that the admissibility condition (1) allows us to include within the ZCOVER elementary FD's with scope X where R[X] is decomposable.

The procedure DECOMPOSE is shown in Fig. 2.1 where it is written using ALGOL like syntactic constructs extended to set variables in an obvious fashion. It may be seen that the complete relatability conditions of (2.1) and (2.2) represent the core of the procedure. Under certain conditions, which will be discussed later, the procedure may fail thereby calling the special procedure REPORTFAILURE which may provide the appropriate warning.

The complete algorithm has been programmed in slightly modified form as part of a program designed to help extract the FD's and MD's from the user by displaying minimal contrived instances of a relation [SILV 78], [SILV 79].

We shall next illustrate the operation of the algorithm by describing an example

given in [ZANI 79a]. For convenience in what follows we define:

$$Fp = F1 \cup F2,$$
$$Gp = G_F \cup G11 \cup G22.$$

3. EXAMPLE: THE RELATION DMV

We consider here a relation, DMV, which might be used by a Department of Motor Vehicle to store data concerning vehicles, driver licenses and traffic violations. The attributes are:

LIC: License numbers of motor vehicles
MAKE: Manufacturers of motor vehicles
MODEL: Models of vehicle
YEAR: The year in which the vehicle was manufactured
VALUE: The current value of the vehicle
OWNER: The unique owner of a vehicle for simplicity we only use the family name
DRVL: The driving license number
VIOL: The code number of traffic violations
DATE: Month, day and year of the violation.

Thus one may start with a relation DMV (LIC, MAKE, MODEL, YEAR, VALUE, OWNER, DRVL, VIOL, DATE) with sample content shown in Table 2.1.

TABLE 2.1
A Sample Situation for the Relation DMV

DMV (LIC,	MAKE	MODEL	YEAR	VALUE	OWNER	DRVL	VIOL	DATE)
2LL 235	FORD	CAPRI	1972	1550	OWEN	A771235	22520	4/5/75
							22351	1/9/75
AFF 255	GM	NOVA	1975	2300	OWEN	A771235	22520	4/5/75
							22351	1/9/75
QPR 937	FIAT	128	1973	1900	WEST	C991837	21803	9/5/74
							22100	2/12/75
LFV 222	GM	VEGA	1976	2550	MANN	D331973	0000	0/00/00

Note that the various violations on each driver' record have been grouped together to provide a representation that is more concise and expressive than a true First Normal Form. The corresponding 1NF is obvious.

The dependencies in relation DMV are found to be as follows:

D1: LIC -> MAKE
D2: LIC -> YEAR
D3: LIC -> MODEL
D4: LIC -> VALUE
D5: LIC -> OWNER
D6: LIC -> DRVL
D7: LIC ->> {VIOL, DATE}
D8: {MAKE, YEAR, MODEL} -> VALUE

D9: {MAKE, YEAR, MODEL} ->> {LIC, OWNER, DRVL, VIOL, DATE}
D10: OWNER -> DRVL
D11: OWNER ->> {VIOL, DATE}
D12: OWNER ->> {LIC, MAKE, YEAR, MODEL, VALUE}
D13: DRVL -> OWNER
D14: DRVL ->> {VIOL, DATE}
D15: DRVL ->> {LIC, MAKE, YEAR, MODEL, VALUE}

For the sake of brevity, the MD counterparts of the FD's shown above (obtained by replacing the single by the double arrow) are not listed although they do belong in the list. Whenever we need them we shall refer to them by underlining the corresponding FD. Thus the above list is also assumed to include the MD's: D1, D2, D3, D4, D5, D6, D8, D10, D13.

The FD's, D1 to D6, follows from the fact that the license of a vehicle is unique so that it determines its other properties. The FD, D8, implies that the make, model and the year of a car determine its value. The FD's D10 and D13 imply that the car owner and the driver whose license was recorded during a traffic violation are one and the same. This is clearly not a realistic condition, but it has been adopted to illustrate a one-to-one correspondence. D7 is is inferrable by transitivity of MD's from D5 and D14. D9 is inferrrable by complementation from D8.

We shall now consider the application of the decomposition algorithm to the relation DMV.

A1: F0 = {D1, D2, D3, D4, D5, D6, D8, D10, D13}
G0 = {D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15}
G_L = Ø

A2: ZCOVER, ACOVER <- Ø; L<-1.

A3: DECOMPOSE(W)
where, W = {LIC, MAKE, MODEL, YEAR, VALUE, OWNER, DRVL, VIOL, DATE}.

We now follow the action of DECOMPOSE(W). STEP1 consists of a call to the procedure DETERMINE which, having verified that this is the first decomposition step, sets F = F0 and G_m = G0. STEP2 does not perform any calculation as none of the FD's in F spans the whole set W. STEP3 is skipped since G_m = Ø. STEP4 consists basically in testing the CRC's for each MD in G_m. We shall now examine this step assuming that the MD's in G_m are tested in the order they are listed.

D1: LIC ->> MAKE is tested first. We have W1 = {LIC, MAKE}, W2 = {LIC, MODEL, YEAR, VALUE, OWNER, DRVL, VIOL, DATE}. The procedure COMPUTE now computes F1, F2, F_S, G_F, G11, G22 which are found to be:

F1 = {D1}
F2 = {D2, D3, D4, D5, D6, D10, D13},
F_S = Ø
G_F = {D1, D2, D3, D4, D5, D6, D10, D13},
G11= {D1 },
G22 = {D2, D3, D4, D5, D6, D7, D10, D11, D13, D14}.

As the CRC's are tested we find that D12 and D15 are derived through complementation on D11 and through partitionability on the complement of D11 and on D13; however D8 and D9 cannot be derived (indeed it is always true that a nontrivial FD or MD with left side X can only be inferred from a set of dependencies if that set contains some other dependency having as left X or a subset of it.) Thus CRC2 is also violated. The case of D2 and D3 is quite analogous to that of D1: the CRC's fail in the same way (indeed it may be seen from the set of dependencies that the attributes MAKE, YEAR and MODEL are quite

symmetrical). If D4 is used, D8 is missing in Fp^+ and CRC1 fails; moreover D8 and D9 are missing in Gp^+ and CRC2 also fails. If D5 is used, D10 and D13 are missing in Fp^+ and CRC1 fails; moreover D10, D11 and D12 are missing in Gp^+, and CRC2 also fails. With D7 we find that D11, D12, D14 and D15 are missing from Gp^+, thus CRC2 fails even though CRC1 succeeds. Using D8 we find that while D4 is missing from Fp, it is inferrable in Fp^+ from D1, D2, D3 and D8, thus CRC1 succeeds. We also find that while D4, D12 and D15 are missing from Gp, they can be inferred in Gp^+, thus CRC2 also succeeds.

Following the successful decomposition of DMV into DMV [MAKE, YEAR, MODEL, VALUE] and DMV [LIC, MAKE, YEAR, MODEL, OWNER, DRVL, VIOL, DATE], the procedure DECOMPOSE is applied to the former. This time STEP2 is carried out while STEP3 places into ACOVER the atomic subrelation "1: {MAKE, YEAR, MODEL, VALUE}".

The decomposition of DMV [LIC, MAKE, YEAR, MODEL, OWNER, DRVL, VIOL, DATE] is attempted next. In STEP1, the procedure DETERMINE computes F and G_m to be:

F = {D1, D2, D3, D5, D6, D10, D13},
G_m = {D1, D2.D3, D5, D6, D7, D10, D11, D13, D14, D15'}.

where D12' and D15' denote the MD's constructed from D12 and D15 by projecting out VALUE. Note that D9 is now left out of G_m because it is a single elementary MD due to the disappearance of D8.

This time the use of D1 in STEP4 is found to satisfy both CRC's (D12' and D15' are inferrable from D10, D11, and from D13, D14.) The subrelation DMV [LIC, MAKE] is found to be atomic; thus L, ACOVER and ZCOVER are augmented and the projection DMV [LIC, MODEL, OWNER, DRVL, VIOL, DATE] is decomposed next. Here D2 is found to satisfy both CRC`s and the subrelation DMV [LIC, MODEL] is found to be atomic; thus L, ACOVER and ZCOVER are augmented and the subrelation DMV [LIC, OWNER, DRVL, VIOL, DATE] is decomposed next. Here STEP1 yields:

F = {D5, D6, D10, D13}, and
G_m = {D5, D6, D7, D10, D11, D12", D13, D14, D15"},

where, D12": OWNER ->> LIC and D15": DRVL ->> LIC.

Utilizing D5 in STEP4 we find that D10 and D13 are missing from Fp^+ while D10, D11 and D12" are missing from Gp^+. So both CRC's fail. Symmetrically, utilizing D6, we find that D10 and D13 are again missing from Gp^+ while now D13, D14 and D15" are missing from Gp^+, thus both CRC's fail again. When we try D7 we find that while CRC1 is satisfied, dependencies D11, D12", D14 and D15" are missing from Gp^+, thus CRC2 fails.

Finally D10 yields an acceptable decomposition into DMV [OWNER, DRVL] and DMV [OWNER, LIC, VIOL, DATE]. The invocation of DECOMPOSE upon the former enters the elementary FD's, D10 and D13, in ZCOVER under the same label "5", and then adds "5: {OWNER, DRVL}" to ACOVER. The decomposition of DMV [OWNER, LIC, VIOL, DATE] is attempted next. STEP1 yields,

F = {D5}
G_m = {D5, D7, D11, D12"}.

In STEP4, D5 and D7 both fail CRC2 as in both cases Gp^+ misses D11 and D12". However D11 passes the CRC's yielding the the two atomic subrelations, "6: {OWNER, VIOL, DATE}" and "7: {LIC, OWNER}" and the elementary FD, "7: LIC -> OWNER".

The algorithm terminates successfully leaving the following subrelations in ACOVER and ZCOVER:

Label	ACOVER	ZCOVER
1	{VALUE, MAKE, MODEL, YEAR}	{MAKE, YEAR, MODEL} -> VALUE
2	{LIC, MAKE}	LIC -> MAKE
3	{LIC, YEAR}	LIC -> YEAR
4	{LIC, MODEL}	LIC -> MODEL
5	{OWNER, DRVL}	OWNER -> DRVL
5	{OWNER, DRVL}	DRVL -> OWNER
6	{OWNER, VIOL, DATE}	None
7	{LIC, OWNER}	LIC -> OWNER

4. DISCUSSION

The decomposition algorithm described above has the following beneficial properties:

(1) It decomposes a complex relation pair-wise into simple well-defined primitives,

(2) It preserves all the information contained in the original relation,

(3) It minimizes redundancy in a certain sense.

The primitives produced by the decomposition consist of,

I. the attribute sets of atomic subrelations called the A-structure which are displayed in ACOVER, and

II. a set of elementary FD's called the Z-structure which are displayed in ZCOVER.

The final set of subrelations generated by the decomposition are atomic in the sense that it is not possible to decompose them in a pair-wise fashion without losing information contained in he original relation.

The decomposition preserves the structural information and the content of the original relation, since:

(a) The original relation is reconstructable from the atomic relations by a sequence of natural joins;

(b) Those original nontrivial FD's which are not included in the Z-structure can all be derived from the latter by the inference rules of FD's. Indeed it has been shown that ZCOVER constitutes a minimal cover for the FD's of the original relation [ZANI 79a];

(c) The original nontrivial MD's which disappear during the decomposition are all regenerated during the reconstruction of the original relation and all its original FD's.

Finally the decomposition algorithm minimizes redundancy both in terms of atomic subprojections and elementary FD's resulting from the decomposition. It should be noted that we do not claim minimization in terms of storage. Indeed efficient implementations would usually combine several atomic relations such as DMV [LIC, MAKE], DMV [LIC, YEAR] and DMV [LIC, MODEL]. But we do claim that the atomic relations generated by the decomposition algorithm provide a basic, fundamental view of the database of minimum granularity. Indeed if we realize that the tuples of a relation represent declarative sentences describing the universe of interest, atomic relations consist of atomic sentences in the logical sense, such as f(a,b), whereas non-atomic relations consist of molecular sentences such as "f(a,b) and g(a,c)". Furthermore, as we shall see later, entities may be abstracted from the primitives in a simple and natural fashion.

Another point of interest concerns the possible non-uniqueness of atomic

decompositions. The decomposition process itself may be described by an unordered binary tree where the branch nodes of the tree denote a decomposition step whereas the leaf node denote the final atomic subrelations. Thus figure 4.1 denote the decompositions tree corresponding to the process described in section 3. However, the order in which the elements of G_m are considered in the decomposition algorithm is not specified, therefore the MD's of G_m may be treated in arbitrary sequence. The key question is whether the resulting ACOVER and ZCOVER change depending on the order in which the MD's are considered. In general several decomposition trees may yield the same ACOVER and ZCOVER (although the label values will permute according to the order in which elementary FD's and atomic subrelations are generated). However, the presence of a one-to-one relationship, such as the one implied by the two FD's, A -> B and B -> A, yields multiple decomposition corresponding to various possible exchanges between A and B. As we will see later therefore, for relation DMV, there exists four atomic decompositions corresponding to various interchanges of

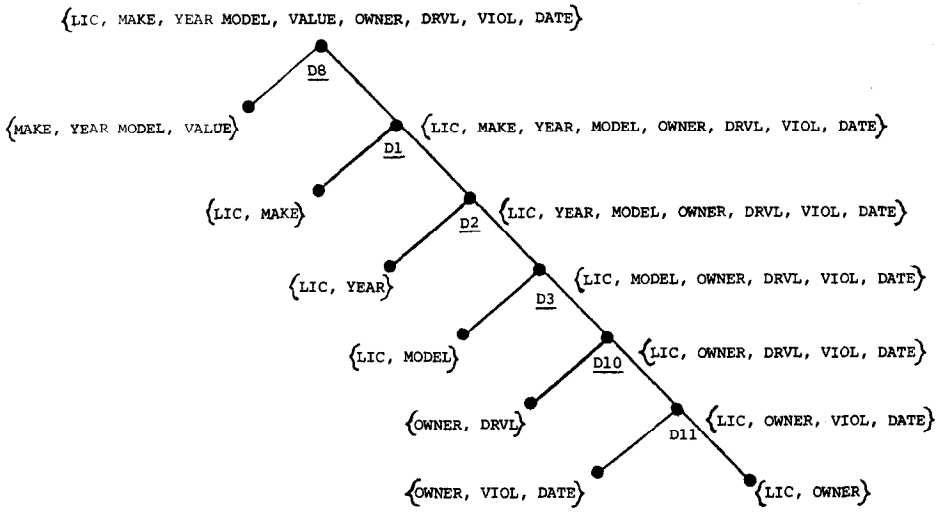


Fig. 4.1. A First Decomposition Tree for DMV.

OWNER and DRVL.

5. PROBLEMS

There are several problems which are still present in the decomposition algorithm. First of all, there is no formal proof that the decomposition step produces a non-redundant atomic decomposition (i.e. one where if any subrelation is dropped the join of the remaining ones will fail to return the original relation). The generation of a redundant decomposition by Algorithm 2.1 actually represents a very rare occurrence which seems connected with the situation where a relation can be losslessly decomposed in three but not in two projections. Therefore the designer may want to verify non-redundancy. Reference [ZANI 79b] provides two fairly effective methods to do this. One is based on the algorithm presented in [AHO 77] for testing the lossless join property; the other is based upon certain results of hypergraph theory applied to the CAZ-diagrams.

Another problem concerns cases where the decomposition algorithm yields a call to REPORTFAILURE thereby signalling that no (pairwise) decomposition is acceptable.

We have encountered at least two such situations which are described in [ZANI 79a]. In the first case, the addition of a new attribute significantly clarifies the picture and yields satisfactory decomposition. In the second case, a decomposition into three projections preserves the Z-structure, at the price of introducing redundancy. We are currently studying these interesting cases.

6. CAZ-GRAPHS

A CAZ-graph (short for combined A- and Z-graph) is a graph representation of the schema of a relation having as vertices the attributes of this relation. Every arc of the graph is labelled and belongs to one of three types: one-to-one, one-to-many, or many-to-many. These are constructed as follows:

one-to-one arcs: There is a one-to-one arc labelled "i" between vertices A and B if ZCOVER contains two FD's with common scope Y_i and with right side A and B, respectively; such an arc is denoted by two opposite arrowheads.

Many-to-one arcs: There is a many-to-one arc labelled "i" between A and B if:

(a) ZCOVER contains a FD of scope Y_i whose left side contains A and whose right side contains B, and

(b) there is no one-to-one edge with label "i" between A and B;

such an arc is denoted by an arrowhead pointing from A to B.

Many-to-many arcs: There is a many-to-many arc labelled "i" between A and B if:

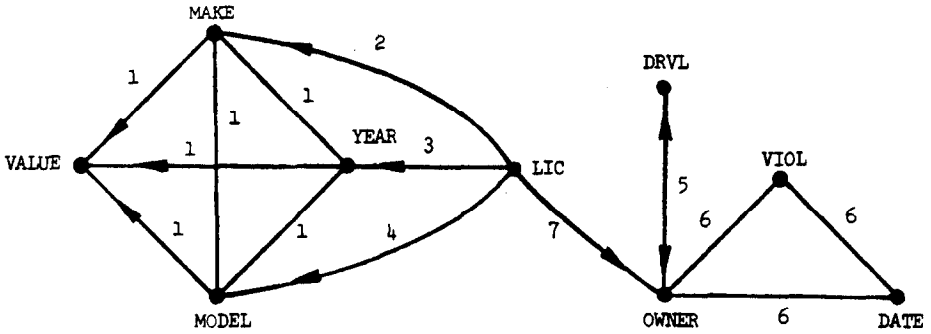
(a) ZCOVER contains Y_i which contains both A and B; and

(b) There are no one-to-one nor one-to-many arcs with label "i" between A and B;

such an arc is denoted by a plain line with no arrowheads.

Thus the CAZ-graph for the relation DMV, corresponding to the decomposition presented in section 3, is given in figure 6.1. The following points may be noted:

- Both atomic relations and FD's are conveniently labelled with the same label L.
- Binary atomic relations appear as single arcs.
- N-ary atomic relations appear as identically labelled complete subgraphs (cliques) consisting of N(N-1)/2 arcs.
- Multiple arcs between two edges are possible (although none appear in figure 6.1).
- Edge "5 denotes a one-to-one arc between OWNER and DRVL as evidenced by the two FD's, OWNER -> DRVL and DRVL -> OWNER, appearing under the same label.
- Edge 7 denotes a one-to-many arc between LIC and OWNER (indicating that a single owner may own several cars, but not vice-versa) as evidenced by the FD, LIC->OWNER.
- The three edges labelled 6 denote many-to-many arcs between OWNER, VIOL and DATE. This may be interpreted in the usual fashion by realizing that, fixing the value of one of the attributes, the two remaining ones exhibit the usual binary many-to-many relationship. Thus for example a given owner may have several different violations on the same date or the same violation on different dates (note that we have made the assumption that the owner of a vehicle is also its driver, at the date of the violation).



Label	ACOVER	ZCOVER
1	{VALUE, MAKE, MODEL, YEAR}	{MAKE,YEAR,MODEL} -> VALUE
2	{LIC, MAKE}	LIC -> MAKE
3	{LIC, YEAR}	LIC -> YEAR
4	{LIC, MODEL}	LIC -> MODEL
5	{OWNER, DRVL}	OWNER -> DRVL
"	"	DRVL -> OWNER
6	{OWNER, VIOL, DATE}	none
7	{LIC, OWNER}	LIC -> OWNER

Fig. 6.1. A relation schema and its CAZ-graph representation for DMV, corresponding to the decomposition described in section 3.

- The six edges labelled 1 denote a clique representing the atomic subrelation, DMV [VALUE, MAKE, MODEL, YEAR], having 4x3/2 = 6 edges. Three of these edges having a single arrowhead represent the FD, {MAKE, YEAR, MODEL} -> VALUE; these are one-to-many arcs interpreted in the usual fashion by fixing two of the three attributes on the left-hand side. The other three edges labelled 1 but bearing no arrowhead denote that the three arcs are of the many-to-many types interpreted as described above.

Because of the one-to-one correspondence, DVRL<->OWNER, there exists four different CAZ-graphs for relation DMV. Figure 6.2 shows one of these graphs, obtained from the one of figure 6.1 by replacing OWNER with DRVL under label 7. The other two graphs can be constructed from figure 6.1 by replacing OWNER with DRVL under label 6 in one case, and under both labels 6 and 7, in the other case.

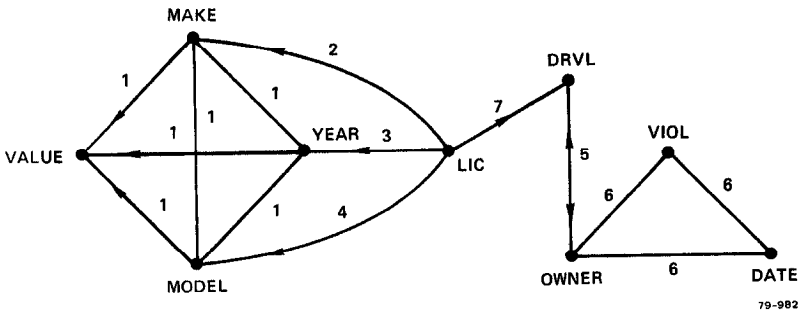
It has been our experience that CAZ-graphs provide a remarkably concise and expressive description which is easily understood by non-specialists.

7. SYNTHESIS OF E-R DIAGRAMS FROM CAZ-GRAPHS

Among the numerous proposals for graphical representations of database conceptual schemas, the E-R diagrams introduced by Chen [CHEN 76] have found particular favor among database designers; this may be attributed to the combination of simplicity and naturalness characterizing the model. Therefore it is of interest to indicate the relationship between CAZ-graphs and E-R diagrams, and to describe how the latter may be synthesized from the former.

An E-R diagram may be directly synthesized from a CAZ-graph which does not include FD's with scope Y, where R[Y] is decomposable, as follows:

1. Replace every vertex of the CAZ-graph by a rectangular node representing an entity in an E-R diagram.



Label	ACOVER	ZCOVER
1	{VALUE, MAKE, MODEL, YEAR}	{MAKE, YEAR, MODEL} -> VALUE
2	{LIC, MAKE}	LIC -> MAKE
3	{LIC, YEAR}	LIC -> YEAR
4	{LIC, MODEL}	LIC -> MODEL
5	{OWNER, DRVL}	OWNER -> DRVL
"	"	DRVL -> OWNER
6	{OWNER, VIOL, DATE}	none
7	{LIC, DRVL}	LIC -> DRVL

Fig. 6.2. Relation schema and CAZ-graph for a second decomposition of DMV

2. Replace the edges of every clique labelled "j" of the CAZ-graph by a diamond-shaped node labelled "j" representing a relationship in the E-R diagram; connect by arcs this relationship node to all the entity nodes corresponding to the vertices of the clique; the arc leading to an entity A is tagged "1" if, in the CAZ-graph, vertex corresponding to A had an edge labelled "j" with an arrow pointing towards A; it is tagged "many" otherwise, this being indicated by the letters "M", "N" or "P".

Thus, figure 7.1 shows the E-R diagram directly synthesized from the CAZ-graph presented in figure 6.1. The transformation from CAZ-graphs to E-R diagrams and vice-versa is thus seen to be very intuitive and easily understood. It may be seen that each "relationship" node of the E-R diagram represents an atomic relation involving the entities connected to that node. Furthermore the tag "1" labelling an arc of the E-R diagram denotes that the entity at the end of that arc is functionally dependent upon the other entities of the atomic relationship. Thus, for example, the tag "1" upon the arc leading to VALUE in figure 7.1 denotes the FD, {MAKE, YEAR, MODEL} -> VALUE.

The direct synthesis described above yields an E-R diagram whose entities correspond to the attributes of the CAZ-graph. Yet, while there is no precise definition of "entity", it is generally accepted as a class of compound objects having a number of properties (described by attributes) and whose members may be distinguished unambiguously. Thus, rather than describing VALUE, MAKE, MODEL or YEAR as entities, it is more in accordance with current usage to define an entity (or entity class) such as "VEHICLE-TYPE" (VT) with attributes VALUE, MAKE, MODEL and YEAR. The corresponding E-R diagram is shown in Fig. 7.2. Here the "relationship" between the entities VT and LIC is shown as a single diamond labelled "3" with left arc tagged "1" and right arc tagged "M" replacing the previous three diamonds marked "2", "3" and "4". While this representation

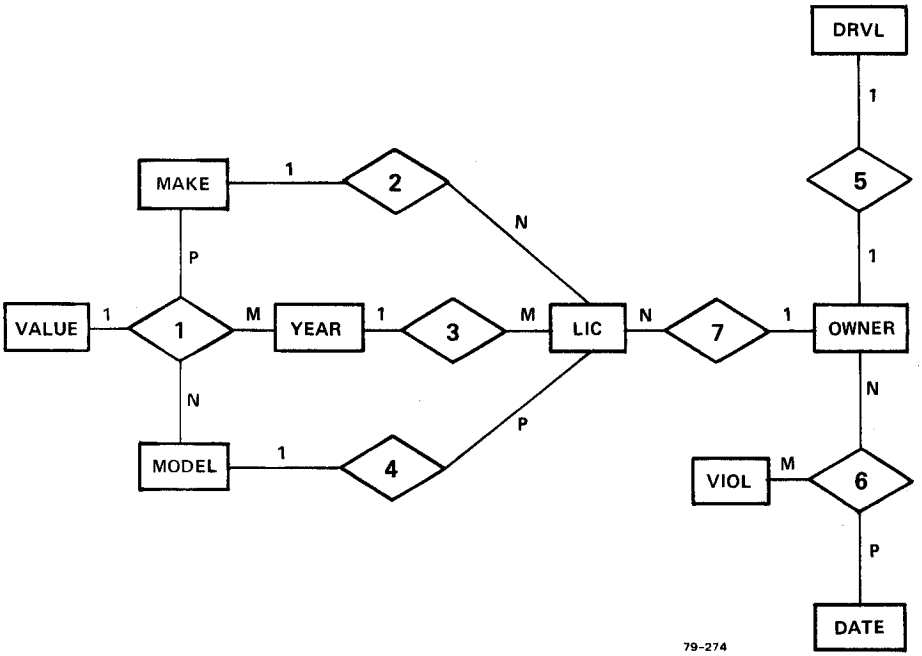


Fig. 7.1. The E-R diagram representation of the CAZ-graph of Fig. 6.1.

corresponds to the more customary usage of entities, it wipes out the description of the entity VT in terms of its four attributes. On the other hand the modifications introduced by [BROW 79] through the ERA model permits the information concerning the attributes of entities and their FD's.

Other possible E-R diagrams corresponding to the CAZ-graph of figure 6.1 are shown in figure 7.3. Thus figure 7.3(a) utilizes an entity CAR with (invisible) attributes MAKE, VALUE, YEAR, MODEL and LIC. Figures 7.3(b) and (c) utilize also the entity CIT (for citation) which is characterized by the attributes VIOL, DATE and OWNER (or DRVL). On the other hand figure 7.4 shows an E-R diagram which corresponds to a redundant decomposition of DMV. Such E-R diagrams may often arise in practice as representations of actual databases where the redundancy may be accidental or voluntary. These diagrams which allow more than one path from one entity to another are being studied both in the context of the ERA model and the relational model.

The question as to what constitutes an entity seems to have a subjective answer; the various E-R diagrams depicted above can all be justified under various circumstances. Certain authors [FLOR 78] have specified that entities should have a unique key attribute and at least one other attribute. Thus for example the entity, VT, would require a unique vehicle type designator; the entity, CAR, has LIC as key, while CIT would require a citation number. Atomic relations comprising attributes with no unique key would then be considered as relationships.

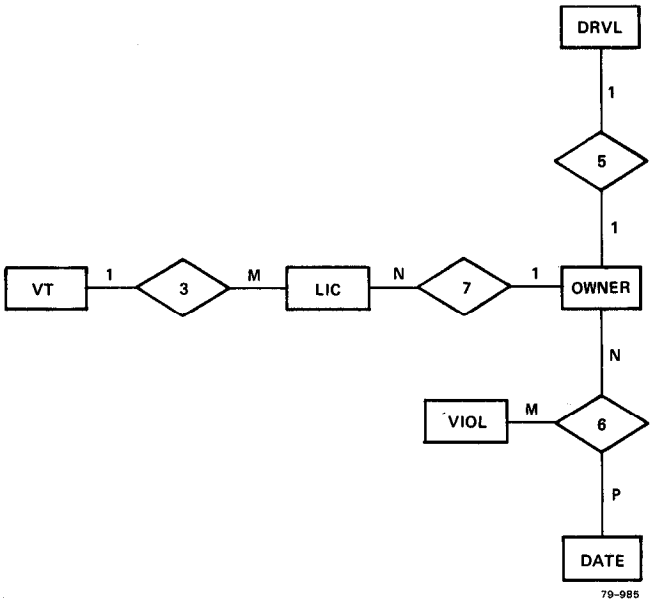


Fig. 7.2. The E-R Diagram representation of the CAZ-graph of fig. 6.1. utilizing the entity VT.

Another problem concerning the relationship between CAZ-graphs and E-R diagrams arises when the former includes elementary FD's of non-atomic scope. Consider for example the relation WS (DAY, TIME, GROUP) which represents the weekly schedule of occupancy of a conference room where various groups meet [ZANI 79b]. Assuming that only one group meets at any given day and time, and that a group must follow the same schedule for any day it uses the room, the situation is summarized by the following elementary dependencies:

D1: {DAY, TIME} -> GROUP
D2: GROUP ->> TIME
D3: GROUP ->> DAY

The decomposition algorithm here yields:

Label	ACOVER	ZCOVER
1	none	{DAY, TIME} -> GROUP
2	{GROUP, TIME}	none
3	{GROUP, DAY}	none

and the corresponding CAZ-graph is shown in figure 7.5. Here it is not clear as to which relationships should appear in the E-R diagram. Figure 7.6 presents three possible approaches:

(a) Representation of atomic relations and FD's, ignoring non-atomic relations such as {TIME, DAY} -> GROUP.

(b) Representation of non-atomic elementary FD's neglecting finer relationships contained therein.

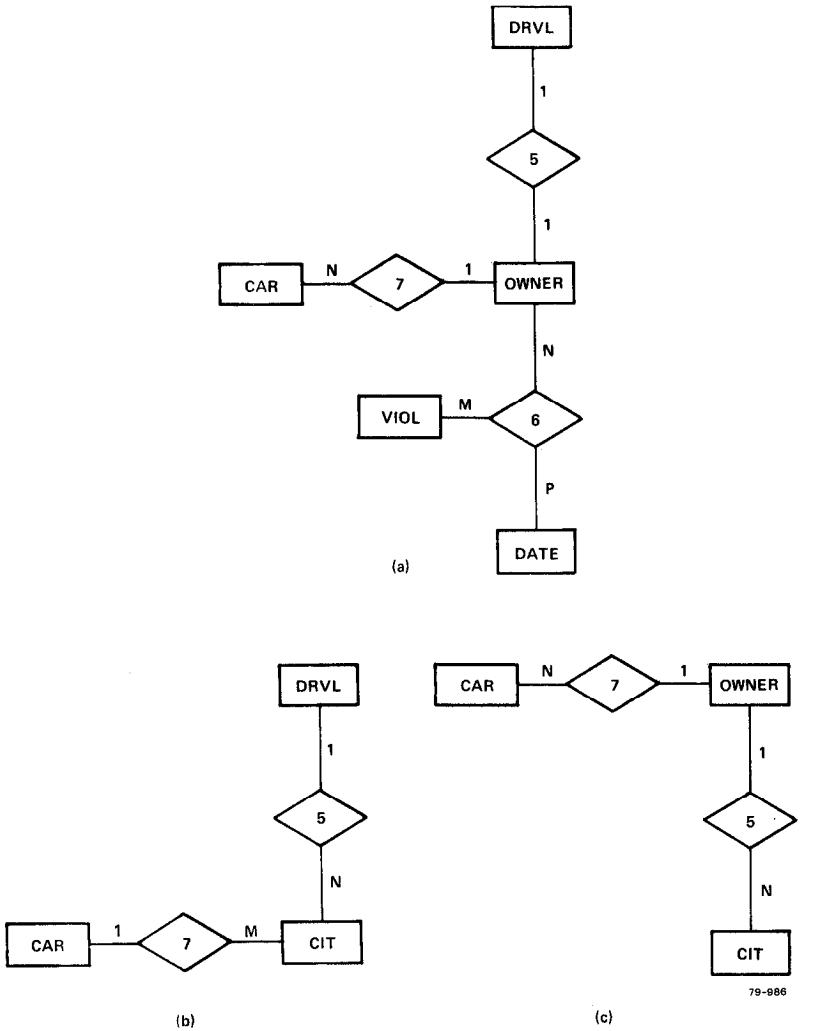


Fig. 7.3. Various possible E-R diagrams corresponding to the CAZ-graph of Fig. 6.1.

(c) Representation of atomic relations and of both atomic and non-atomic FD's.

Each of these solutions has certain advantages and disadvantages.

In summary, once the entities of interest have been defined, there exists a simple correspondence between E-R diagrams and CAZ-graphs. This shows that relationships

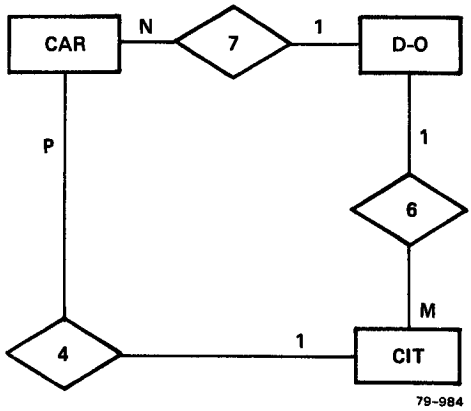


Fig. 7.4. An E-R diagram for the relation DMV which corresponds to a redundant decomposition.

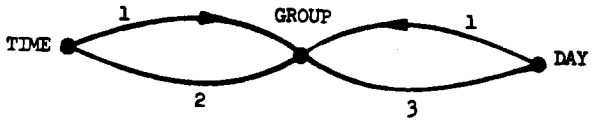


Fig 7.5. A CAZ-graph Representing Relation WS.

can formally be defined by atomic subrelations and elementary FD's combined under the admissibility condition. These can in turn be generated starting with the FD's and MD's of relations and using the decomposition Algorithm 2.1.

A number of issues regarding entities are left up to the designer. In particular, we know that decomposition algoritms, which are all based on the "universal relation" assumption, can not capture (a) the notion that instances of a certain entity-class can exsist which does not participate in any relationship, and (b) the notion that an entity may appear under different roles (such as in the part explosion proplem.) The designer will have to deal with these issues when selecting the entities, before the decomposition algorithm can be applied.

The problem of selecting entities ideserves further investigation. The intended role of the schema will probably be a main factor in the selection process. For instance, it has been shown that CAZ-graphs, when each attribute is treated as an entity, can be used (1) as a design aid to generate normal form schemas [ZANI 79a], and (2) as a conceptual schema to establish logical and functional mappings between external schemas and internal schemas through query reformulation [ZANI 76]. On the contrary for the enterprise schema, the designer may prefer few high-level entities, each including many attributes. Thus while the same graphical model can be used for schemas in different roles, different levels of abstraction are least fitted for each individual role.

8. CONCLUSION

We have presented a new algorithm for decomposing relations into primitive elements which include atomic relations and elementary FD's. This algorithm is based on the complete relatability conditions. It provides a first step for designing

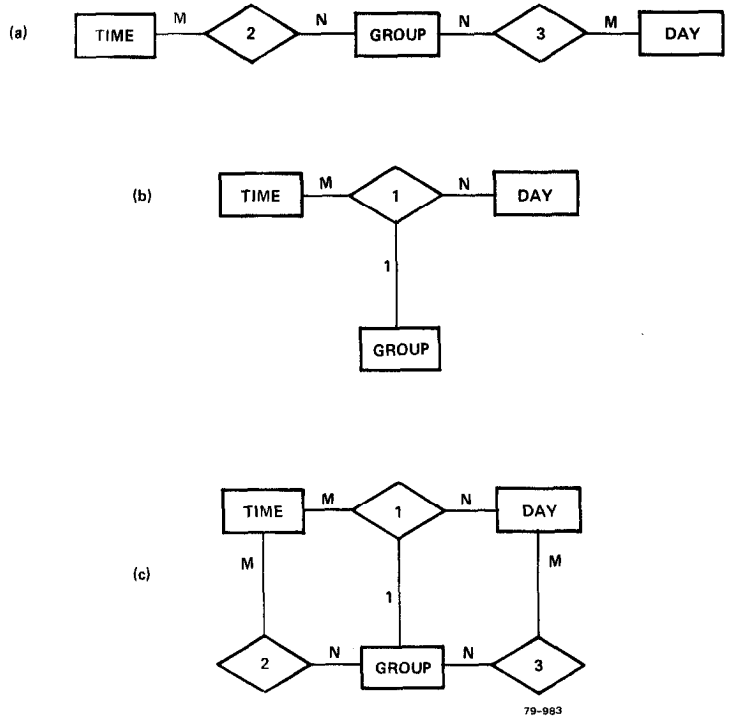


Fig. 7.6. Three possible E-R Diagrams Corresponding to the CAZ-graph for Relation WS.

conceptual schemas in two steps. In the secons step the designer will have to check for redundancy in the schema.

The algorithm has been illustrated in some detail by an example of moderate complexity exibiting several decompositions.

The results of the decomposition may be conveniently exhibited pictorialy in terms of CAZ-graphs which display the information both concisely and expressively.

The CAZ-graphs may be used to synthesize E-R diagrams although subjective uncertainty of what constitutes an entity leaves certain ambiguities. Other possible correspondences between CAZ-graphs and E-R diagrams remain to be investigated.

REFERENCES

[AHO 77] Aho, A. V., Beeri, C. and Ullman, J., "The Theory of Joins in Relational Databses," ACM Transactions on Database Systems, Vol. 4, No. 3, Sept. 1979.

[BEER 77] Beeri, C., R. Fagin and J. H. Howard, "A Complete Axiomatization for Functional and Multivalued Dependencies in Database Relations," ACM-SIGMOD Int. Conference on Management of Data, Toronto, Canada, Aug. 3-5, 1977.

[BROW 79] Brown, R. R. and Ramey, T. L. "Domains and Data Types in the ERA Information Model," Proc. of the Int. Conf. on Entity-Relatioship Approach to System Analysis and Design," UCLA, Los Angeles, Cal, Dec. 10-12, 1979.

[CHEN 76] Chen, P. P., "The Entity-Relationship Model-Toward a Unified View of Data", ACM Transactions on Database Systems, March 1976.

[CODD 76] Codd, E.F. "A Relational Model of Data for Large Shared Data Banks," Communication of the ACM, Vol. 13, No. 6, June 1970.

[DATE 77] Date, C. J. "An Introduction to Data Base Systems," 2nd ed., Addison-Wesley, New York, 1977.

[FAGI 77] Fagin, R. "Multivalued Dependencies and a New Normal Form for Relational Databases," ACM Trans. on Database Systems, Sept. 77.

[FLOR 78] Flory, A. and Kouloumidjan, J., "A Model and a Method for Logical Data Base Design," Proc. of the 4th Int. Conf. on Very Large Databases, Sept. 13-15, 1978, West Berlin, Germany.

[SILV 78] Silva, A. M., "Derivation of Relational Dependencies Through Case-Study Analysis," M.S. Thesis, Computer Science Department, UCLA, June 1978.

[SILV 79] Silva, A. M. and Melkanoff, M. A. "A Methodology for Helping Discover Dependencies in Relations", to appear in Proceedings of the Database Conference in Toulouse, Dec. 1979.

[ZANI 76] Zaniolo, C. "Analysis and Design of Relational Schemata for Database Systems," Ph.D. Thesis, University of California, Los Angeles, Computer Science Dept. report ENG-UCLA-7669, July 1976.

[ZANI 79a] Zaniolo, C. and Melkanoff, M. A., "On the Design of Relational Database Systems", to appear in ACM Transactions on Database Systems.

[ZANI 79b] Zaniolo, C. and Melkanoff, M. A., "A Formal Approach to the Definition and the Design of Conceptual Schemas for Database Systems", to appear in ACM Transactions on Database Systems.