Short Notes

Join Dependencies in Relational Databases and the Geometry of Spatial Grids

It is shown that a relation whose tuples describe a regular folded spatial grid contains a join dependency, that originates in the geometrical properties of the grid. There is a class of join dependencies corresponding to 2-dimensional grids, or polygonal join dependencies, and a class corresponding to 3-dimensional grids, or polyhedral join dependencies, and so on.

1. INTRODUCTION

The determination of dependencies within relations is important in database design, since undesirable dependencies within relations can lead to anomalies when the database is updated.[3,6] The best-known dependencies are functional dependencies,[2,3] multivalued dependencies,[4,7,10] join dependencies,[6,8] and a variety of generalised dependencies.[9]

In this paper we point out a hitherto unrecognised property of join dependencies, namely that they directly reflect the geometrical properties of a wide variety of spatial grids.

2. JOIN DEPENDENCIES

Join dependencies were first defined by Rissanen.[8] The simplest join dependency occurs in a 3-attribute relation. We use the relation $R(X, Y, Z)$ and the table below to specify this dependency:

X	Y	Z
a	b	—
—	c	d
a	—	c
a	b	c

In other words, if the first three tuples occur in the relation, so must the last. The dependency is a join dependency because a relation containing it can be non-loss-decomposed only into the three relations XY, YZ and XZ, that is, the relation R can be regenerated only from a join of these three relations, and not any two of them such as XY and YZ. Table 1 demonstrates this.

The relation XY^*YZ is the natural join [1, 3] of XY and YZ on the join attribute Y. We see that this does not recover the original relation XYZ.

3. THE TRIANGULAR JOIN DEPENDENCY

We propose that the simplest join dependency above be called a triangular dependency, for it reflects the properties of a folded triangular grid. To see this, consider first the unfolded grid in Fig. 1 (ignoring dashed lines). There are four 'triangles' (whose edges do not have to be straight), identified by the apexes; these are triangles (1, 5, 2), (1, 2, 3), (2, 3, 3), (1, 3, 6) where, as is usual in conventional geometry, the order of listing the apexes is not important. Thus, conventionally, the triangle (1, 5, 2) is the same as triangle (5, 2, 1).

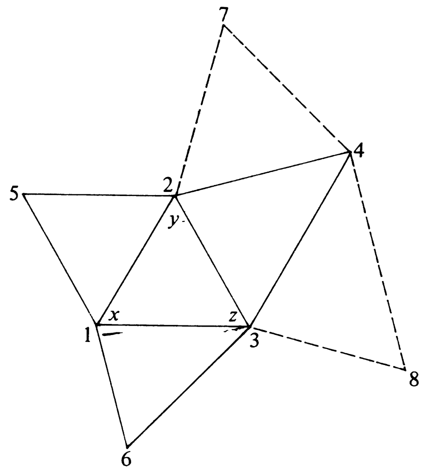

Figure 1.

Now it is self-evident that in the unfolded grid in Fig. 1, if we have triangles (1, 5, 2), (2, 3, 4) and (1, 3, 6), then because of the existence of the edges (1, 2), (2, 3) and (1, 3) the triangle (1, 2, 3) must exist as well. This is a basic property of the surface defined by a triangular grid.

Suppose that we wish to record each of the four triangles in Fig. 1 as a tuple of the relation $R(X, Y, Z)$. We clearly need a standard procedure for identifying X, Y and Z apexes for each triangle. A viable method is as follows. Simply fold each triangle on a triangle edge, with repeated folds if necessary, until the triangle being folded has been fitted on top of, or below, a triangle that has been previously selected as a standard for the grid. We refer to this standard triangle as the base triangle. For the grid in Fig. 1 we have chosen the triangle (1, 2, 3) as the base triangle, and have defined (1, 2, 3) to be a tuple of the relation $R(X, Y, Z)$.

The folds for each of the triangles (1, 2, 5), (4, 2, 3) and (1, 6, 3) are shown in Fig. 2, so that the relation $R(X, Y, Z)$ describing the four triangles is

X	Y	Z
1	2	5
4	2	3
1	6	3
1	2	3

Clearly, if we have a triangle in the fold with XY edge (1, 2), YZ edge (2, 3) and XZ edge (1, 3), because of the identical apexes, we must have triangle (1, 2, 3). Thus the geometry of the grid explains the tuple-generating aspect of the join dependency.

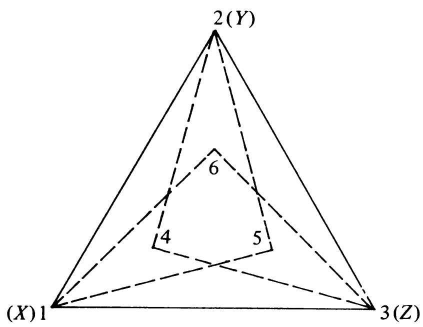

Figure 2.

There is also a geometrical explanation for the inability of a natural join of two of the projections, such as XY and YZ, to recover the original relation. Each tuple of a projection XY describes an XY edge of a triangle physically in the grid. Similarly, each YZ tuple describes a YZ edge. Looking at the grid in Fig. 1, it is clear that some edges occur in only one triangle, while others occur in two triangles. When we perform a natural join of XY tuple (1, 2) with YZ tuple (2, 3), geometrically we are connecting edge (1, 2) with edge (2, 3) at apex 2, to give two, and not three, edges of the grid triangle (1, 2, 3). Nevertheless, these two edges are sufficient to identify the existing triangle (1, 2, 3). However, the natural join XY^*YZ will also give rise to a tuple (4, 2, 5), because of connecting XY edge (4, 2) to YZ edge (2, 5). But these two edges are from two different triangles (4, 2, 3) and (1, 2, 5) that have a common Y-apex after folding. The join of these two edges would therefore form a spurious triangle (4, 2, 5), since there is no XZ edge (4, 5) in the grid in this case (although there could be). And because the edge (4, 5) does not occur in the XZ projection, the final join of XZ with (XY^*YZ) will eliminate the spurious triangle (4, 2, 5).

It is thus clear that all recognised properties

Table 1

X	Y	Z
1	2	5
4	2	3
1	6	3
1	2	3

XYX or R

X	Y
1	2
4	2
1	6

XY

Y	Z
2	5
2	3
6	3

YZ

X	Z
1	5
4	3
1	3

XZ

X	Y	Z
1	2	5
1	2	3
4	2	5
4	2	3
1	6	3

XY^*YZ

X	Y	Z
1	2	5
1	2	3
4	2	3
1	6	3

$(XY^*YZ)^*XZ$

of the join dependency in a ternary relation have their explanation in the geometry of a folded (2-dimensional) triangular grid. That the result we have described is quite general with respect to folded triangular grids becomes clear from a study of the additional triangles (2, 4, 7) and (3, 4, 8) in Fig. 1 (dashed lines).

Let us first fold these triangles over the base triangle. Triangle (2, 4, 7) is first folded along edge (2, 4), and then along edge (2, 7), showing us that the XYZ tuple (4, 2, 7) will describe that triangle. The folding of triangle (3, 4, 8) similarly results in XYZ tuple (4, 8, 3). The relation XYZ for the six triangles in Fig. 1 is therefore:

X	Y	Z
1	2	5
4	2	3
1	6	3
1	2	3
4	2	7
4	8	3

Looking at Fig. 1, we would expect that since triangles (1, 2, 3), (2, 7, 4) and (3, 4, 8) are in the grid, then triangle (2, 4, 3) must be in it as well, so that there should be a generation of XYZ tuple (4, 2, 3) from three other tuples – in conformity with the rules for specifying the join dependency. There is. According to the rule from section 2, since we have tuples (4, 2, –), (–, 2, 3) and (4, –, 3), we must have tuple (4, 2, 3).

Extending stepwise in this manner, it is clear that the grid can be extended indefinitely with preservation of the join-dependency rule. The grid does not have to form a plane in order for the join dependency to hold. An edge of the grid can be an edge of three or more triangles, that is, the grid can have been constructed from the intersection of two or more grids. We define two grids to intersect when they have at least one edge in common. In all these cases the relation for the folded grid will contain a join dependency.

Note the following restriction, however. For a planar grid, an apex is saturated, so that no additional triangles can be inserted, when it is the apex of 6 triangles. Thus, in Fig. 1, triangle (5, 2, 7) is invalid. Instead we would expect to have triangles (5, x, 2) and (x, 2, 7).

4. POLYGONAL JOIN DEPENDENCIES

If a grid is composed of tetragons, or 4-sided meshes, the relation described by folding the tetragonal grid contains another join dependency, which we propose to call a tetragonal dependency.

Consider the grid of 5 tetragons, as in Fig. 3, and consider the tetragon in the middle to be the basic tetragon, on which the others are folded. Now, referring to Fig. 3, the dependency arises because the existence of tetragons (5, 6, 2, 1), (2, 7, 8, 3), (3, 9, 10, 4) and (12, 1, 4, 11) means that tetragon (1, 2, 3, 4) must also exist. When we fold the five tetragons on base tetragon (1, 2, 3, 4), each tetragon can be described by a tuple in the relation:

X	Y	Z	W
10	9	3	4
1	12	11	4
1	2	6	5
7	2	3	8
1	2	3	4

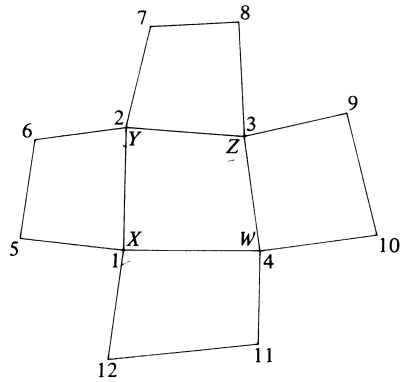

Figure 3.

which therefore must contain the join dependency:

X	Y	Z	W
a	b	–	–
–	b	c	–
–	–	c	d
a	–	–	d
a	b	c	d

If we project on XY, YZ, ZW and XW, we will be able to reform $XYZW$ only if all four projection relations are joined. As in the case of the triangular dependency, a join of less than the total number of projections gives rise to spurious tuples, for the same geometrical reasons.

In a similar manner, a grid made up of 5-sided figures will give rise to a relation with a pentagonal dependency, and so on for hexagonal, septagonal grids, etc.

5. POLYHEDRAL JOIN DEPENDENCIES

Relations described as polygonal folded grids define the class of polygonal join dependencies. In addition, relations describing polyhedral folded grids define the class of polyhedral dependencies.

As with the polygonal grids, a polyhedral grid must be folded before each volume of the grid can be given a tuple in the relation for the grid. Although 'folding' here is mathematically similar to the case of a polygon, geometrically, it consists of turning the object inside out, as is illustrated in Fig. 4 for the case of a cubical grid. The relation to describe the grid has attributes $XYZWABCD$ with the tuple (1, 2, 3, 4, 5, 6, 7, 8) describing the basic cube of the grid (Fig. 4). When the neighbouring cube (4, 3, 12, 13, 8, 7, 10, 11) is folded it will be described by the tuple (13, 12, 3, 4, 11, 10, 7, 8). If we have six such cubes, one extending from each side of the basic cube, then the basic cube must exist. This is the basis for the hexahedral join dependency, which is

X	Y	Z	W	A	B	C	D
a	b	c	d	–	–	–	–
–	–	–	–	e	f	g	h
a	b	–	–	e	f	–	–
–	–	d	c	–	–	g	h
–	b	c	–	–	f	g	–
a	–	d	–	e	–	–	h
a	b	c	d	e	f	g	h

in the relation describing a folded grid of hexahedra. In similar manner we can have

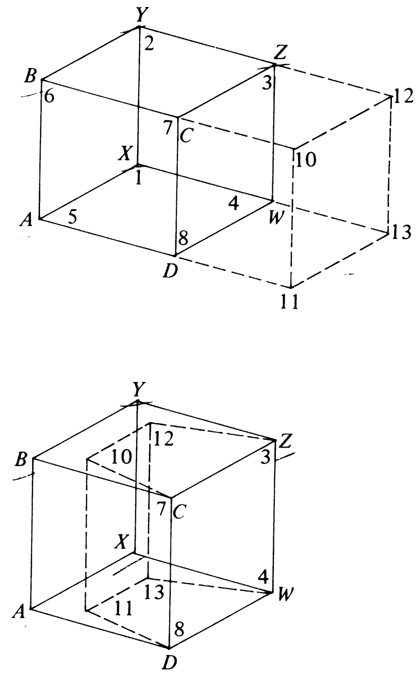

Figure 4.

tetrahedral and pentahedral join dependencies, and so on for all the polyhedrons. This gives us a whole class of 2-dimensional or polyhedral join dependencies.

6. *N*-DIMENSIONAL JOIN DEPENDENCIES

There will clearly also be a join dependency in a relation that describes a folded grid of n-dimensional objects, for example, 4-dimensional cubes. Thus it is possible to classify join dependencies as 4-dimensional, 5-dimensional and so on. For each dimensional class there will be an essentially infinite number of join dependencies.

7. CONCLUDING REMARKS

The dependencies for the 2- and 3-dimensional cases could conceivably have application in databases describing engineering structures. However, it seems highly unlikely that the higher-order join dependencies will have any application in the foreseeable future. Nevertheless, the elucidation of the individual higher-order n-dimensional join dependencies represents an interesting mathematical challenge, which will undoubtedly be taken up in years to come.

J. BRADLEY, University of Calgary,
2500 University Drive NW, Calgary, Alberta,
Canada T2N IN4
Received August 1985

REFERENCES

1. A. V. Aho, C. Beeri and J. D. Ullman, The theory of joins in relational data bases. *ACM Trans. on Database Syst.* **4**, (3), 297–314 (1979).
2. W. W. Armstrong, Dependency structures of data base relationships. *Proc. 1974 IFIP Congress*, pp. 580–583. North-Holland, Amsterdam (1974).
3. E. F. Codd, Further normalization of the data base relational model. In *Data Base Systems*, pp. 33–64. Prentice-Hall, Englewood Cliffs, N.J. (1972).