2010 3rd International Conference on Computer and Electrical Engineering (ICCEE 2010)
IPCSIT vol. 53 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V53.No.1.33

An optimal Scheme Based on Local Query for Computer Graphics

Ce Fan+ and Xiaorong Wu

School of Informatics, Guangdong University of Foreign Studies, Guangzhou, China

Abstract. The construction of data structure plays a key role for the speedup processing of data in CAD/CAM systems. This paper proposes the schemata whose amount of storage is linearly proportional to the number of entity and implements the retrieval to data in local range. The interrogation of special relations and properties of three dimensional objects are often involved in applying for CAD/CAM, especially in solid modeling. The interrogation and query to data are discussed in this paper in terms of the primitives that we call topological query. The access schemata for three dimensional topological data structure founded on faces, edges and vertexes in graph. Boundary-based half-symmetry, loop and symmetry data structure schemata given in this paper are depended on the primitives of nine kinds of topology queries for creating new entity, and its analysis of storage and time complexity is shown. The time complexity based on the nine kinds of access primitives is confined in local or constant time, so that the optimum schemata for local queries can be implemented in CAD/CAM better.

Keywords: Component; Computer Graphics;Topological Query ;entity modelling; CAD/CAM

1. Introduction

Many methods and algorithms for CAD/CAM gave being discussed and developed [1-5]. In solid modeling, new entities can be modified or generated by Euler or Euler-like operations. For data retrieval, the following query may be necessary: "given an entity X_i, find all entities Y_j connected or adjacent to it". Queries in the foregoing scenarios are fairly common and generally present, but they depend highly on the interrelation among space objects in time. In this correspondence, we show that this type of queries can be expressed in terms of nine access primitives, and propose optimization schemata in correspond with different storage cost according to boundary-based topology structure

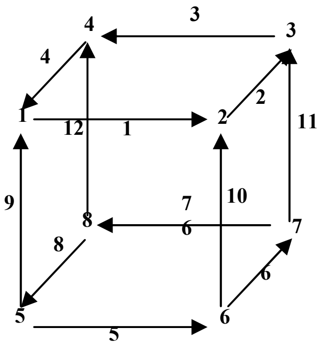
2. Relations, Storage and Time

Suppose the topological entities are F(Face), E(Edge) and V(Vertex). To facilitate discussion, we use the following symbols:

$\mathbf{X}_i$ —an entity of X type
{X} —the entity set of X type
X —the entity number of X type
YX_i —the entity number of Y type connected to X type
$X_i(Y)$ —the Y set for X

where X,Y=F/E/V, and the followings are the same except explanation.

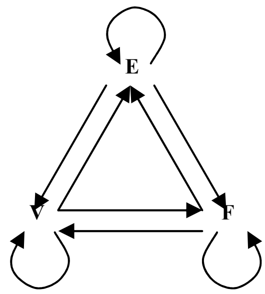
+ Corresponding author.
E-mail address: fance@mail.gdufs.edu.cn.



(a)

Local relation	Adjacent number	Adjacent set
VV_1	3	{2,4,5}
EV_1	3	{1,4,9}
FV_1	3	{1,3,6}
⋮		
FF_4	4	(1,2,3,5)

(b)



(c)

Figure 1． a cube example (a) a topology expression of (a), (b) some adjacent relations and (c) nine relations

A boundary data structure can be thought of as a set of adjacent relationships among topological entities. Let a relation be denoted by {X}→{Y},unless ambiguity arises, the followings are denoted by X→Y for simplicity.

2.1 Definition

If there exists X→Y, the query of complete relation is

$$X_i \to \{W \subseteq Y \mid (\forall X_i)(X_i \in X) \wedge (|W| = YX_i)\}$$

The relation V→E, for example, shown in the Fig. 1, stores its three edges for each vertex in V. Thus for each Vi, its adjacency edges can be modified or accessed. The storage complexity of a relation X→Y can be computed by taking the Sum ∑YXi. For example, the total storage for E→V is ∑VEi.

2.2 Topological query

Since there nine topological relations formed from F, E and V, there exist nine data structure access primitives. We will refer to them as topological queries Ti, for i=1,…,9(see TABLE I.). Hence, the time complexity of data structure schemata can be cost by these nine access primitives. T1——T9:

Table 1 Table Type Styles

Nine access primitives	**Meaning**
T_1(V→V)	Given a vertex, find all vertexes connected to it
T_2(V→E)	Given a vertex, find all edges connected to it
T_3(V→F)	Given a vertex, find all faces around it
T_4(E→V)	Given an edge, find the two vertexes connected to it
T_5(E→E)	Given an edge, find the four edges connected to it
T_6(E→F)	Given an edge, find the two faces intersecting at it
T_7(F→V)	Given a face,, find all vertexes around it
T_8(F→E)	Given a face, find all edges bounding it
T_9(F→F)	Given a face, find all faces around it

We show the example of Fig. 1 based on these nine primitives. Queries T1——T9 , correspond to the time complexity measures for direct, indirect, and inverse relations V→V,…,F→F.

3. Bounds for Storage and Time Complexity

This section introduces the techniques for counting storage and for evaluating the time required for answering T1——T9 , queries and show the lower and the upper bound for both storage and time for all data structures. For measuring the number of storage locations needed, we will use total number of edges E as the unit.

It is clear that C_2^9 and C_9^9 are bound for storage and time among eight classes of data structures C_m^9 (m=2,…,9) formed from F,E and V. If there exists a direct relation Ti of X→Y for any data structure with C_m^9, Ti is the query of constant time K. If there exist relations E→V and E→F, since an edge is shared by two faces and two vertexes, their storage complexity is the sane 2E.

3.1 Lemma

The storage complexity of any relation X→Y except E→E is 2E.

Proof: In the closed solid modeling, since there exists YXi = ZXi = XXi(Xxi = EEi), the time complexity can be evaluated by YXi for any relation X→Z. If each entity Xi is adjacent to YXi edges, and Vi, Vj∈Xi(Y)∩Xj(Y)=Yi for any Yi , each edge Yi is stored exactly twice. For E edges, we get ∑YXi = 2E.

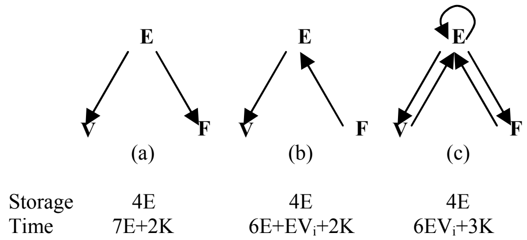


Figure 2. The C_2^9 class and winged-edge schema

		C_2^9	C_3^9	C_4^9	C_5^9
time	Best	6E +EV$_i$ +2K	4EV^2 +2EV$_i$ +3K	5EV$_i$ +4K	4EV$_i$ +5K
	worst	7E+2K	6E+3K	5E+4K	4E+5K
storage	Best	4E	6E	8E	10E
	worst	4E	8E	10E	12E
		C_6^9	C_7^9	C_8^9	C_9^9
time	Best	3EV$_i$ +6K	2EV$_i$ +7K	EV$_i$ +8K	9K
	worst	3E+6K	2E+7K	E+8K	9K
storage	Best	12E	14E	16E	20E
	worst	14E	16E	18E	20E

Figure 3. The storage and time for C_m^9

Fig. 3 shows the storage for any C_m^9 and the upper and the lower bound for both storage and time——(20E, 4E), (7E+2K, 9K).

3.2 Wing- edge structure

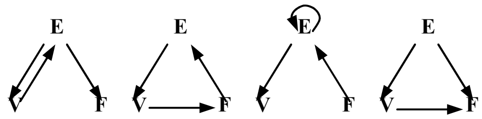
Baumgart[4] discovered the use of a fractional relation $F \xrightarrow{1} E$ and $V \xrightarrow{1} E$ in designing of the wing-edge data structure. $F \xrightarrow{1} E$ gives rise to a storage cost of F. Similarly, $V \xrightarrow{1} E$ costs V in storage. The total storage is therefore 8E+(V+F)=9E.

For time complexity, there are clearly three constant time queries. For the other six queries, local clocking around a face or around a vertex is required, costing EFi or EVi in time respectively. Since EVi is of the same order as EFi , we chose EVi as the unit. Thus, the total time complexity of the winged-edge is 6EVi+3K.

4. Transmission, Loop, and Symmetry Data Structure and their Optimality

Fig. 2(a) shows the lower and upper bound for storage and time for the class of C_2^9. But if we transform E→F into F→E then the other schema of C_2^9 class is formed. Storage cost does not change, but since there is the side benefit of a indirect relation F→V which can be derived from F→E and E→V, the time complexity decreases.

From a given Fi ,EFi edges are returned in constant time. For each of EFi edges, exactly two vertexes are returned by E→V in constant time, as for EFi edges, the overall time for answering T7 is EFi or EVi . which forms a transmit relation schema. It changes global E query into local query EVi, and also decreases from 7E+2K to 6E+EVi2K.



(a) (b) (c) (d)

Storage	4E	4E	6E/8E	6E
Time	$3E+3EV_i$ +3K	$4EV^2_i+2EV_i$ +2K	$5E+EV_i$ +3K	6E+3K

Figure 4. The C_3^9 class (a) half-symmetry (b)loop (c) and (d) others

Suppose we add a new relation V→E to the C_2^9 data structure as shown in Fig. 2(a) in terms of the transferability. The additional storage cost is 2E. There are clearly three constant time queries, but because of increasing V→E, it gives rise to the side benefit of three indirect queries V→E, V→V and E→E. Thus, overall time complexity highly decreases to 3E+3EVi+3K.

If we apply the transferability of relations on Fig. 2(b), adding a new relation V→F, the Fig. 4(b) schema can be brought about. Clearly, all relations can be derived locally, except the three direct relations. It is the addition of the V→F that brings about the five side benefit from indirect query which changes from global E to local EVi .

We add a new relation V→F to the C_2^9 data structure as shown in Fig. 2(b), which storage is still 2E, but the time complexity changes greatly. Hence, the topological data structure independent of global E is raised, the worst time is the local O (EV2). Ti , for example, goes through two variable set Vi(F) and Fi (E), so 2FVi*EFi is required. Because of FVi=EVi, its time is EV2. Thus, the overall time complexity for Fig. 4(b) is the 4EV2+2EVi+3K.

Though the loop relation schema rises the twice order complexity EV2, because of the Y adjacent objects of X entity YXi≤6, the Fig. 4(b) is preferable to (a) and its complexity is proportional to E when E is greater.

4.1 Theorem 1

In the C_3^9 class, the loop relation schema gets the optimum time.

Proof: The first case: if there exists the relation X→X(see Fig. 4(c)) in the C_3^9 class(X=F/E/V), it only contributes a K constant time and does not raise indirect relations, so no side benefit on time improvement. The best case is the addition of X→X on Fig. 2(a), its time complexity is more than that of Fig. 4((a) and (b).

The Second case: if there not exists relation X→X in the C_3^9 class, it must be the Fig. 4(d)-like schema. Because of corresponding to the addition of a K constant time in C_3^9 class, its time complexity is greater than that of Fig. 4(a) and (b).

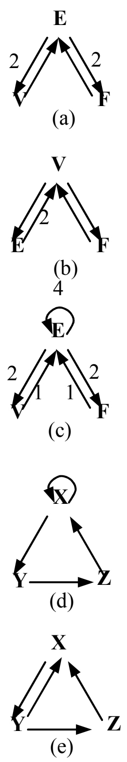
We have shown a loop relation schema of the C_3^9 classes, but it has still the twice order complexity of the local query. Suppose transferability is applied again on the Fig. 4(b), there would be the Fig. 6(e)-like three schemata. For the addition of a relation X→Y, since there exists Y→X, X→X and Y→Y can be answered in linear time EVi, but Z→Z is still loop path(a EV2 complexity).

Look at the Fig. 4(a), if a relation F→E whose storage is still 2E is added again, three EVi time queries, which can be derived from X→Y/X→Z and Y→X/Z→X for any relation X→X, would be bought about (see Fig. 4(a)schema based on the edge-symmetry).

There are symmetric schemata edge-symmetry, vertex-symmetry and face-symmetry which is similar Fig. 5(b). For vertex-symmetry/face-symmetry, there exists still the twice order complexity from local queries(face/vertex), so that it can be proved that the edge-symmetry is preferable to the others, and also not faster optimal schemata than the Fig. 5(a) in C_4^9.

4.2 Theorem 2

In the C_4^9 class, the edge-symmetry schema gets the optimum time.



	Storage	Time
(a)	8E	$5EV_i+4K$
(b)	8E	EV^2+4EV_i+4K
(c)	9E	$6EV_i+3K$
(d)	8E/10E	$3EV^2+2EV_i+4K$
(e)	8E	EV^2+4EV_i+4K

Figure 5. The C_4^9 class (a) edge-symmetry (b) vertex-symmetry(c) winged-edge (d) and (e) others

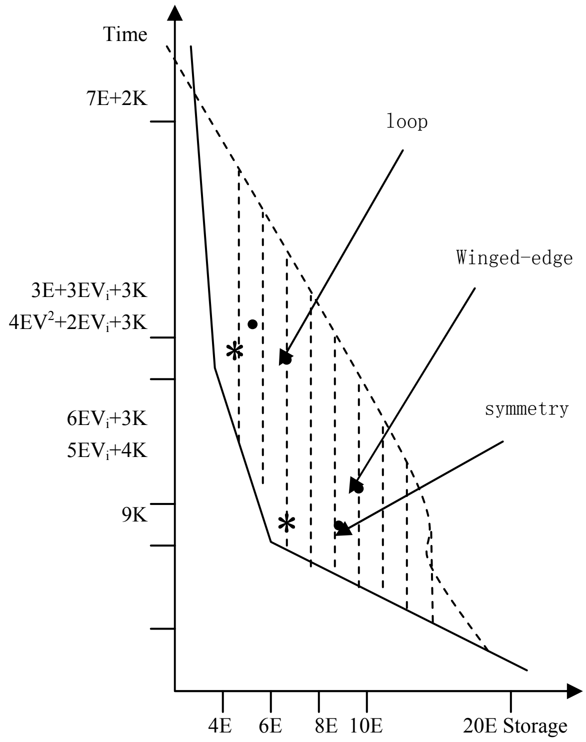


Figure 6. Complexity curve on the storage and time for C_4^9

Proof: The first case: if there exists the relation X→X in the C_4^9 class, it only contributes a K constant time and does not raises indirect relations. The best case is the addition of X→X in C_3^9 class which improves the complexity from global E to K. It is more than that of edge-symmetry(see Fig. 5(a) and (d)).

The second case: If it is not a symmetric schema in C_4^9 class, and not exists the relation X→X, then it must be Fig. 5(e) schema which exists twice order complexity (EV2), so that its time is greater than that of the edge-symmetry schema.

It can be seen from Fig. 6 that the left bound curve of the shadow area is the curve to form the optimum schemata with different storage, the right(dash line) is the worst expression on time, and the shadow area is where possible various schemata appear.

5. Conclusion

The optimum schemata for local query have are proposed above, and proved their uniqueness and effective method for constructing boundary data structure in this paper. The idea C_5^9, C_6^9, C_7^9, C_8^9 and C_9^9 schemata will be formed on the C_4^9 symmetric schema, but adding 2E storage only gets the linear change(see the Fig. 6) from EVi to K. It does not raise more optimum schemata (refer to the theorem 1 and theorem 2). The edge-symmetry schema contributes good traversal for searing various objects in 3D entity modeling, and its storage and time cost is preferable to the others schemata.

6. Acknowledgment

This work is supported by the Natural Science Foundation of Guangdong University of Foreign Studies under Grant No. GW2006-TB-012.

7. References

[1] Wang Hongshen, Zhang Shusheng, "3D CAD Surface Model Retrieval Algorithm Based on Distance and Curvature Distributions," Journal of Computer-Aided Design & Computer Graphics, Vol.22, pp.762-770, May 2010

[2] ZHANG Jun, QIN Xiao-lin, HU Cai-ping, “Completeness Research on Topological Relations Between 3D Spatial Objects of Lines and Bodies,” Journal of Image and Graphics, Vol.14, pp.543-549,March 2009

[3] Wang Fei, Zhang Shusheng,Bai Xiaoliang, “3D Model Retrieval based on both the Topology and Shape Features,” Journal of Computer-Aided Design & Computer Graphics, Vol.20, pp.99-103,Jan. 2008.

[4] Tony C. Woo,“a Combinational Analysis of Boundary Data Structure Schemata,” IEEE CG&A, Vol.5,pp.19-26, March 1985.

[5] T. C. Woo, “A Constant Expected Time, Linear Storage Data Structure for Representing Three-dimensional Object,” IEEE Transactions on System, Man and Cybernetics, Vol.14, pp.356-362, March 1984

[6] A.A.G. Requicha, “Representation for Radial Solids: Theory, Methods and System,” ACM Computation Survey, Vol.12, pp.1221-1229, Ari. 1980

[7] P.R.Willson, “Euler Formulas and Geometric Modelling ,” IEEE CG&A, Vol.5, pp.25-36, Agu. 1985

[8] David Cowan, “Survey of the CAD Field, ” CAD international Directory, 1985