The Journal of GIS Association of Korea, Vol. 2, No. 2, pp.147-163, October 1994

Transportation Network Data Generation from the Topological Geographic Database

Keechoo Choi*

GIS 위상구조자료로부터 교통망자료의 추출에 관한 연구

최 기 주

ABSTRACT : This paper presents three methods of generating the transportation network data out of the topological geographic database in the hope that the conversion of the geographic database file containing the topology to the conventional node-link type transportation network file may facilitate the integration between transportation planning models and GIS by alleviating the inherent problems of both computing environments.

One way of the proposed conversion method is to use the conversion software that allows the bi-directional conversion between the UTPS (Urban Transportation Planning System) type transportation planning model and GIS. The other two methods of data structure conversion approach directly transform the GIS's user-level topology into the transportation network data topology, and have been introduced with codes programmed with FORTRAN and AML (Arc Macro Language) of ARC/INFO. If used successfully, any approach would not only improve the efficiency of transportation planning process and the associated decision-making activities in it, but enhance the productivity of transportation planning agencies.

요 약

본고는 위상관계를 지니는 vector GIS 데이터구조로 부터 교통계획이나 도로계획시 필요한 node-link 중심의 교통망 자료를 획득하기 위한 절차 및 활용프로그램을 제시하는데 목적이 있다. 이를

* 아주대학교 교통공학과 (Department of Transportation Engineering, Ajou University, 5 Wonchon Dong, Paldal Ku, Suwon, 442-749, Tel. (0331) 219-2538)

위해 ARC/INFO GIS와 UTPS(Urban Transportation Planning System) 계의 TRANPLAN이 구현을 위한 대표적 소프트웨어로서 선정되었고, 상호간 데이터의 교환을 위한 세가지의 위상변환(Topology Conversion) 방법이 제시되었다. 첫번째 방법은 소프트웨어 개발자(ARC/INFO의 ESRI 및 TRANPLAN의 Urban Analysis Group)들이 공동개발하여 별도의 소프트웨어 package로 구축한 프로그램에 대해서 변환절차의 개요 및 프로그램의 단계적 확대방안에 관해서 소개하였고, 둘째로, ARC/INFO에서 사용가능한 node attribute table(NAT)을 별도의 topology로 구축하여 arc attribute table(AAT) 및 NAT에 있어서의 node 체계를 상호 관련시켜서 node-link 교통망 자료로 변환하는 알고리즘을 AML(Arc Macro Language)로서 제시했으며, 끝으로 FORTRAN 언어를 사용한 AAT에서의 node 변환 알고리즘을 소개하였다. 이러한 GIS 데이터의 위상변환의 필요성은 GIS를 교통부문에(특히 교통수요 예측) 직접 이용함은 물론 더 나아가서는 양자 사이의 효율적인 데이터의 교환이 그목적이라고 할 수 있다. 비록, 본고에서는 이들 세가지 방법 상호간의 구체적인 장단점에 관해서는 토의되지 못했지만 세방법중 어떠한 것이든 교통수요 예측모형을 위한 데이터의 변환과정에 역할을 할 수 있으며, 이들 시스템간의 효율적 통합은 교통계획과정에서의 생산성 향상에 기여할 것으로 보인다.

Introduction

GIS is a computer−assisted and integrated environment for geographic data creation, storage, retrieval, management, manipulation, analysis, and display. It relies on the integration of three distinct aspects of computer technology; data base management, routines for displaying and plotting graphic representation of the data, and algorithms and techniques that facilitate spatial analysis (Antenucci et al., 1991).

As the computing environment is getting better and better due to the price reduction of the computer hardware and software, GIS is more readily available to end users even on desktop computing environment. In addition, the growing concerns on GIS usage and standardization, especially in Korea, had drawn attention of the people in this community.

The applications of GIS into transportation planning and management, however, were largely those of FM (facility management) type works rather than those of analysis oriented works. Recognizing that importing most of transportation analysis functions (demand modeling, routing, and allocation) into GIS is impossible, it is believed that an interfacing algorithm is needed to integrate those analysis functions and the generic features of GIS.

This paper presents three methods of generating the transportation network data out of the topological geographic database in the hope that the conversion of the geographic database file to the conventional transportation network data file would facilitate the linkage between the two by alleviating the inherent problems of both (transportation) planning tools.

Spatial Data Model and Review of Vector Data Format of GIS

In using computers for handling spatial data such as transportation planning and land use planning with computer, we must reduce our data to the computer's level of comprehension. In order to do this, we have to specify at least three things for the computer. There are:

location — *Where* each feature is in geographic space.

attribute — *What* each geographical feature is. and,

topology — *What spatial relationship* each geographical feature has to other geographical features.

Geographical data can be represented in a number of ways: as simple lists, as ordered sequential files, or as indexed files. When organized into a database, there are three different data models; they are hierarchical, network, and relational data structures.

Hierarchical structures require a large overhead to be maintained in the form of data redundancy, index file, and pointers. In the network structure, travel within the database is facilitated, but its size is enlarged by the pointers that have to be maintained and updated every time a change is made to the database. A relational database structure does not store pointers and has no hierarchy. Instead, Instead, data are stored in simple records maintained as two-dimensional tables. But considerable time is needed to search for the right records in a large database. Hence, very skillful design work is needed if these data structures are to perform adequately within a GIS database (See Burrough, 1986; Date 1981).

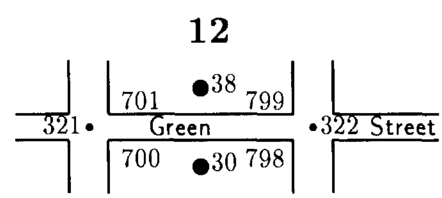
These days, however, vector GIS gains more attention in transportation application due to its flexibility of adapting its data structure. That is, entire geometrical relationships between map features (topology) of points and lines are built up. In addition, in most vector GIS, there are no limits on resolution capability and location for a point.

The choice of a particular spatial data structure is one of the important early decisions in designing a geographic information system. Since topology is more explicitly represented in vector data structure, several forms of vector data structures will be reviewed in this section. Among Star and Estes' (1990)classification of vector models, (They are; (1) Whole polygon structure, (2) DIME (Dual Independent Map Encoding) structure, (3) Arc—node structure, (4) Relational structure, and (5) DLG (Digital Line Graphs) structure), DIME, arc—node, and DLG structures will be reviewed since they are already available in transportation planning community of United States and may shed light on some hints in building transportation GIS database for Korean environment.

DIME Structure

The DIME (Dual Independent Map Encoding) structure, developed for use by the U.S. Bureau of the Census, was designed to incorporate topological information about urban areas for use in demographic analyses (Cooke, 1987). The basic element of the DIME file structure is a line segment defined by two end points or nodes. Figure 1 illustrates the structure of the DIME file in a simplified way, showing segment codes for line segment and nodes' metric location as easting and northing.

An advantage of using the DIME structure for certain applications is its ability to match addresses of spatial objects in multiple files, since the addresses are explicitly stored in the DIME files. A major disadvantage of the DIME structure lies in the difficulty of manipulating complex lines, as in functions that require a search along streets. This is because streets are generally broken into discrete street segments by the cross streets, so it is a significant computational effort to follow the segments in sequence when required.



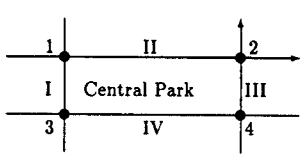
Street Name: Green
Street Type: Street
Left Addresses: 701-799
Right Addresses: 700-798
Left Block: 38, Left Tract: 12
Right Block: 31, Right Tract: 12
Low Node: 321
X-Y Coordinate: 155 000 - 232 000
High Node: 322
X-Y Coordinate: 156 000 - 234 000

Fig. 1 Contents of A DIME File (Totschek, Almendinger, and Needham, 1969)

Arc—node Structure

In an arc—node data structure, objects in the database are structured hierarchically. In this system, points are the elemental basic components. Arcs are the individual line segments that are defined by a series of x−y coordinate pairs. Nodes are at the ends of arcs and form the points of intersection between arcs. Polygons are areas that are completely bounded by a set of arcs. Thus, nodes are shared by both arcs and contiguous polygons (Peucker and Chrisman, 1975). Several commercial geographic information systems use forms of this arc−node data structure.

Arc−node structures, as shown in Figure 2, permit us to encode the geometry of the data with no redundancy. In an arc−node database, attribute data of arcs or nodes are explicitly linked to geometry. For example, traffic control device descriptions are stored with the relevant nodes, and roadway length and pavement condition are stored with the appropriate arcs.



Node No.	X	Y	...	Traffic Control
1	126.5	578.2		Signal
2	218.6	581.9	...	Sign
3	224.2	470.4		None
4	129.1	471.9		Sign

Arc No.	F-Node	T-Node	Length	...	# of Lanes
I	3	1	106.3		4
II	1	2	92.2	...	4
III	2	4	116.6		2
IV	3	4	95.1		2

Polygon	Arcs	Perimeter	Area	...	Zoning
Central Park	I,II,III,IV	405.2	10203	...	34

Fig. 2 Arc-Node Data Structure
(Adapted from Star and Estes, 1990)

Digital Line Graph Structure

The USGS (United States Geological Survey) has developed a digital cartographic data set called Digital Line Graph (DLG) data. The data are topologically−structured and supplied in DLG version 3 format and it supports both spatial and non−spatial attribute data. The DLG−3 structure has three major components−−nodes, lines, and areas which are analogous to the 0−cell, 1−cell, and 2−cell. The line is the basic element of the DLG−3 structure and any user of the DLG−3 data must start by processing these line records. As shown in Figure 3, the line records contain one

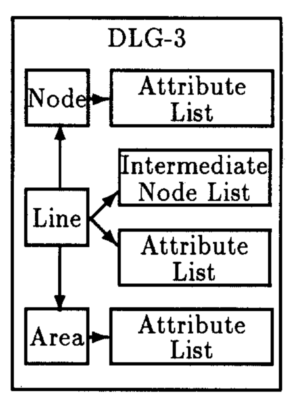


Fig. 3 The Stucture of Digital Line Graph
(Marx, 1990)

—way pointers to the nodes at each end of the line and to the areas on each side of the line. These pointers provide the minimal information needed for a topological structure (Marx, 1986).

The DLG—3 format is an open structure in the sense that users can easily add data to the structure with minimal disturbance to the other elements. However, to use this data for purposes other than mapping, the user should do lots of processing to create additional pointers that allow one to get back to a line from its nodes. Furthermore, there are no provisions for connecting areas having the same attributes. Therefore, the DLG—3 file structure is generally considered as a data exchange format not as an application format.

Topologies: Comparative Views of ARC/INFO and UTPS—type Transportation Models

Topology of Vector GIS

The vector GIS keeps topology in managing their geographical features. Topology is a branch of mathematics dealing with two types of object—points (also called nodes), lines (also called edges); and one type of basic relation between them. The term topology in mathematics has a much more precise meaning. The topological properties dealt with in mathematics are those which are preserved even when an oject is stretched or distorted. Therefore, they are different from geometrical properties.[1)]

In the topological approach, a map may be regarded as a combination of fundamental building blocks called cells. The term n—cell is a building block of dimension n and entities involved in a map are 0—cell (a point; vertex, node), 1—cell (an arc; line segment), and 2—cell (a region; block, parcel, tract, area)(Hollingshead, 1992). Table 1 shows the minimum elements required for a definition of a map, and the incidence relations between the n—cells required to capture the topology of the map.

The spatial relationship between map features are represented graphically on a

Table 1. Basic Map Definition Elements (Hollingshead, 1992)

Spatial Entities	Relationship	Metric
0-cell		Coordinates
	} 0-1 Incidence	
1-cell		Shape
	} 1-2 Incidence	
2-cell		Shape

1) Topology and geometry ane two important properties of spatial objects. Informally, topology involves *adjacency* and *connectivity,* while geometry involves *position.* The two types of spatial properties anre, in general, independent. That is one con not be deived from the other (Bernstein and Eberlein, 1992).

map and the map reader should interpret them to understand. Questions that characterizes the spatial relationship are divided into two categories. First, the topological questions deal with the relationships of the elements, and the metrical questions deal with shape and location (Hollingshead, 1992).

Topological questions

1. What 0−cell, 1−cells, and 2−cells are involved?
2. For a 1−cell, which 2−cells are adjacent (left−right polygon topology;adjacency or contiguity)?
3. For a 0−cell, which 1−cells are terminated (arc−node topology;connectivity)?
4. For a 2−cell, which 1−cells are bounding (polygon−arc topology; areadefinition)?
5. For a 1−cell, which 0−cells are end points (arc−node topology)?

Metrical questions

1. What is the location of a 0−cell?
2. What is the shape of a 1−cell?

The above five topological questions are directly related with notions of *contiguity, connectivity, and area definition.* Contiguity will, especially, play an important role in generating transportation network data from GIS's topological database.

Topological Difference between GIS and Transportation Network Data

Transportation planning involves a great deal of information on features that are geographically distributed over a study area. Locational data collected for network data development is a spatial component comprised of a set of nodes and links. Based on the similarity of spatial data used in transportation system modeling and geographic information systems, GIS could be used to manage data and information needed for transportation network development. In addition, GIS allows many elements of the spatially distributed transportation database to be linked to the graphical display.

However, as noted earlier, the database function of GIS has been mostly used for descriptive−type problems. Even though GIS can perform spatial analysis and network operations that are not available in CAD−type software, not all modeling activities of transportation planning process can be achieved. Therefore, it seems more reasonable to combine the transportation planning models and GIS than to try to obtain solutions only within GIS.

Table2 shows the fundamental differences between transportation planning models and GIS in the handling of network data.

Table 2. Topological Difference of GIS and Transportation Data

Transportation Model	GIS
Abstract context Single topology (link-node) Link-node structures Sort-indexed	Geographic context Many topologies (point, arc, polygon) Chain structures Spatially-indexed

Gaps between the two are expected since each system was developed for its own specific purpose. In combining the two systems, the topological gap of the data structures between the two systems should be resolved in order for acombined system to work to overcome the problems of transportation planning models mentioned before.

Transportation Network Data Generation through the Topology Conversion

Regardless of the difference in data structure between transportation planning software and GIS, there are three methods that we can integrate both software if the software concerned are UTPS-type TRANPLAN and vector-type ARC/INFO GIS. The first method was developed by joint efforts of both software developers: Urban Analysis Group and Environmental Systems Research Institute. The other two methods was developed by author using the end-user level commands and FORTRAN language.

Method I: Internal Conversion

As discussed before, even though the network structure of both software is different, these can not override the similarities between them. This led to the fact that one can be drawn from the other[2)]

This attempt was initiated with the belief that coupling ARC/INFO with TRANPLAN will give transportation planners the best of both world. The TRANPLAN to ARC/INFO interface is to be developed in three phase.

1. The first phase consists of a set of programs that convert TRANPLAN files to ARC/INFO database format, and vice verse as shown in Figure 4. In this mechanism, since generation of arcs and arc attribute information does not require the same integrity checking as the generation of a TRANPLAN network, TRANPLAN networks can be direct-

2) The author gave some impetus to both developers in developing the conversion software between ARC/INFO and TRANPLAN, meeting both team leaders: Mr. Ed Granzow of Urban Analysis Group (UAG) who is in charge of TRANPLAN development and Update and Mr. Dale Honeycutt of Environmental Systems Research Institute (ESRI) who is group leader of the ARC/INFO's NETWORK product.

ly reassembled in ARC/INFO coverage files. Arc definition and arc attribute data are imported to TRANPLAN via a modified **$BUILD HIGHWAY NETWORK** function, called **$CONVERT ARC/INFO COVERAGE,** which allows full TRANPLAN verification of network attribute data being input and validation of the input network structure and connectivity. This phase one programs are currently available.[3)]

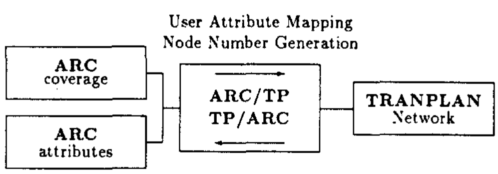


Fig. 4 Conversion between ARC/INFO and TRANPLAN

2. The second phase is to develop the common internal format for both package. The Urban Analysis Group announced new database capabilities which allow network topology and attributes to be stored in selected external databases. This will allow TRANPLAN users to manipulate ARC/INFO coverages directly and transparently, without the overhead and additional effort of file conversion.
3. The final stage is to use ARC/INFO's dynamic segmentation to model one−to−many and many−to−one relationships on networks, and to access such relationships directly from TRANPLAN.

This software will be purchased pretty soon the results of application will be announced thereafter.

Method II: Using NAT

Unlike the pc ARC/INFO system, workstation version ARC/INFO system does have one more important topology called node attribute table (NAT). It is a relationship between nodes in arc attributes tables and user assigned nodes (User−defined node numbers can be assigned easily using the forms command, which is menu interface written Arc Macro Language.). Once the node topology has been created using the command build [cover name] node}, a user can create transportation network topology file that can be seen in Figure 5 (h) that can be directly used for highway network file.

The procedure is as follows. First a

3) For more information on the availability of the phase one software, contact the Urban Analysis Group, 50 Oak Court, Suite 110, Danville, CA 94526−4048, Tel: 510−838−1363, Fax: 510−838−1372.

user digitize the region like (b) in Figure 5, and obtain (c) as a result. After splitting two arcs (arc B and D) by assigning node as shown in (g), the user reassign node numbers; zone nodes (centroids) as 1 and 2, and regular nodes (intersections) from 101 thru 104. Then our goal is to draw and ASCII type file of (h) by relating both NAT and AAT files. The AML procedure for doing this is coded below (For more information on this, refer to RELATE commands).

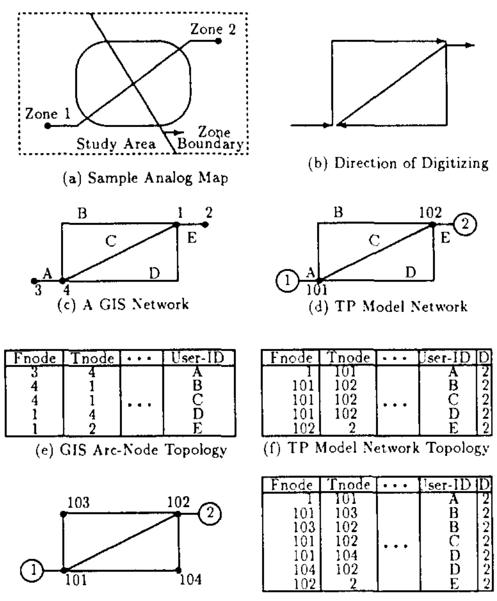


Fig. 5 Topological Difference between GISs and TP Models

```
/* AML (Arc Macro Language) file relating NAT & AAT &args cover
&if [null %cover%] &then &return &
warn USAGE: cmd <cover name>
/* joining .NAT with .AAT
/* can be run in ARC environment
/* create relation
relate add
rel1
%cover%.nat
info
fnode#
%cover%#
linear
ro
rel2
%cover%.nat
info
fnode#
%cover%#
linear
ro
~


/* end of AML
list %cover%.aat %cover%-id, rel1//%cover%-id, rel2//%cover%-id &return
```

The above AML is composed of two parts. One is to relate fnode# and tnode # in arc attribute table and user-assigned node number in NAT file, and then to list the new topology besides AAT file.

Method III: Using FORTRAN Language

As the last resort, the author developed the node conversion algorithm in FORTRAN. The strength of this approach is that this can be used even with pc ARC/INFO that is not equipped with NAT feature.

Let's assume that there is a study area that consists of two zones as shown in (a) of Figure 5. After the digitizing and building topology, we will have the topology (c) and (e) both in analog and abstract forms. However, usable transportation network topology (network data structure) is different from the GIS topology in that the zone centroid is numbered from 1 to 2 as shown in (d) and (f). If there are n zones, numbers from 1 to *n* will be reserved as zone nodes. The network coding scheme in (d) assumes that total number of zones are less than 100 and every intersection node is coded greater than 100. Figures in (g) and (h) represent the modified transportation network and the topology that must be differentiated from the topology in (d) because multiple links between a pair of nodes are not allowed in the UTPS−type transportation planning packages. In other words, a specific link should be composed of a unique pair of nodes.

The topology conversion algorithm proposed in this approach first selects the zone nodes from all nodes in the coverage created in the digitization process based on special values assigned to User−ID for each zonal arc (which correspond to links that connect zone centroids and nodes in transportation planning model). Then it renumbers each zone node and intersection nodes. That is, whatever node is connected to a zone centroid (assumed to be chosen before the digitization process and 1 and 2 in Figure 5 (d), and 3 and 2 in (c), respectively) will be selected first based on the arc User−ID set bigger than a certain limit

(A and E value in Figure 5 (c) should be bigger than this number, in FORTRAN program it is assumed 3000), during the digitization process to differentiate a plain arc from a zonal arc (A zonal arc may be defined in such a way that the User−ID value of the arc exceeds 3000, since the DOS (Disk Operating System) version of TRANPLAN can handle up to 3000 zones. The algorithms and code are shown in the Appendix A.

Conclusion

Up to now, GIS has been adopted as a database integrator, display device for transportation planning model output. This paper reviewed other aspect of GIS's help for transportation planning: topology generator and the development of the conversion mechanisms. Any method can be employed to drawing out the transportation network data by converting the generic GIS topology through the conversion algorithm available.

There will be a greater potential for integrating GIS and the conventional transportation planning models if GIS con not only organize all relevant data and information for overall planning activities but manage data more efficiently. However, substantial time and resources will be required to explore this potential fully. Thus, in order to develop a system which can be used in a field-level works(such as planning agencies), substantial time and resources will be required to fully explore this potential.

References

John C. Antenucci, Kay Brown, Pater L. Croswell, Michael J.Kevany, and Hugh Archer, 1991, *Geographic Information Systems: A Guide to the Technology.* Van Nostrand Reinhold, New York, NY.

David Bernstein and Xu Jun Eberlein, 1992, A relational GIS. In D. David Moyer, editor, *Proceedings of the 1992, Geographic Information Systems for Transportation Symposium,* pages 79-96, American Association of State Highway and Transportation Officals, Portland, OR, March 1-4.

P.A. Burrough, 1986, *Principles of Geographic Information Systems for Land Resource Assessment.* Clarendon Press, Oxford, UK.

Donald F. Cooke, 1987, Map storage on CD-ROM. *Byte Magazine,* 12(8):129-138.

C.J.Date, 1981, *An Introduction to Database Systems.* Addison-Wesley Publishing, Reading, MA.

Mark A. Hollingshead. So what, who cares about topology? In *Proceedings of the 1992 Annual Conference of the Urban and Regional Information Systems Association,* Pages 1-12, Urban and Regional Information Systems Association, Washington, DC, July 1992.

Robert W. Marx, 1986, The TIGER system: automating the geographic structure of the united states census. *Government Publication Reviews,* 13:181-201.

T.K. Peucker and N. Chrisman, 1965, Cartographic data structures. *The American Cartographer,* 2(1):55-69.

Jeffrey Star and John Estes, 1990, *Geographic Information Systems: An Introduction.* Prentice Hall, Englowood Cliff, NJ.

B. Totschek, V.Almendinger, and K.Needham, 1969, A geographic base file for urban data systems. Published by Systems Development Co.

Appendix A: FORTRAN Code for Method III

```
      character middle*32, fi*12, fo*12,mid*32,foo*12
      integer a,b,id,a1,b1,id1,aa(3000),node(5000)

      write(*,*)'                                                     '
      write(*,*)'                                                     '
      write(*,*)'                                                     '
      write(*,*)'          ARC/INFO Arc Attributes File               '
      write(*,*)'              (Node Arc System)                      '
      write(*,*)'                                                     '
      write(*,*)'              CONVERSION PROGRAM                     '
      write(*,*)'                      TO                             '
      write(*,*)'                                                     '
      write(*,*)'         Transportation Planning Systems             '
      write(*,*)'               <<< TRANPLAN >>>                      '
      write(*,*)'          (Zone, Node and Link System)               '
      write(*,*)'                                                     '
      write(*,*)'                      BY                             '
      write(*,*)'                 KEECHOO CHOI                        '
      write(*,*)'       DEPARTMENT OF TRANSPORTATION ENG.             '
      write(*,*)'                AJOU UNIVERSITY                      '
      write(*,*)'                                                     '
      write(*,*)'                                                     '

      write(*,'(a\)')' Enter Input File (ARC/INFO Topology):'
      read(*,'(a12)')fi
C     write(*,'(a\)')' Enter Output File:'
C     read(*,'(a12)')fo
      write(*,'(a\)')' Enter Digitized Map Scale ..........:'
      read(*,*)imap
      fo='int'
      foo='t$000'
      open(1,file=fi,status='old')
      open(2,file=fi,status='old')
      open(6,file=foo,status='new')

C**** Select node & add 3000 to node***********************************
C**** Read from ARC/INFO .fil file and create temporary file T$000 ****

20    read(1,1,end=99)a,b,middle,dist,id,irec
      dist = dist * (2.54 / 1.609 ) * (imap/100000)
1     format(2i11,t22,a22,t45,e13.5,t69,i11,t58,i11)
111   format(i5,i6,18x,f7.3,3x,i9,i9)
      if(id.lt.1000)then
           write(6,111)a+3000,b+3000,dist,id,irec
        else
           rewind(2)
10         read(2,1,end=20)a1,b1,mid,dist,id1
           if(id1.gt.1000) goto 10
             if(a.eq.a1.and.b.eq.b1) goto 10
             if(a.eq.a1.or.a.eq.b1)goto 91
             if(b.eq.a1.or.b.eq.b1)goto 90
           goto 10
        endif

        goto 20

90    write(*,'(a)')' Node Conversion In Progress !!'
      write(*,'(a)')' A*NODE Hello!'
      write(6,111)a,b+3000,dist,id,irec
      goto 20
```

```
91    write(*,'(a)')'  Node Conversion In Progress !!'
      write(*,'(a)')'  B*NODE Hello!'
      write(6,111)a+3000,b,dist,id,irec
      goto 20
99    close(6)
c**** Select centriod and resort it **********************************
c**** Read from temporary file T$000 & create output file ???.int ****

      open(6,file=foo,status='old')
      open(7,file=fo,status='new')
      kk=1
66    read(6,111,end=999)a,b,dist,id,irec
      if(a.gt.3000.and.b.gt.3000) goto 77
       if(a.lt.3000) then
         if(kk.eq.1) goto 41
           do 55 i=1,kk*1
             if(a.eq.aa(i)) then
              a=i
              goto 61
             endif
 55        continue
         aa(kk)=a
         a=kk
         kk=kk+1
 61      if(a.le.b)then
           write(7,111)a,b,dist,id,irec
         else
           write(7,111)b,a,dist,id,irec
         endif
       else
         if(kk.eq.1) goto 42
          do 56 i=1,kk*1
           if(b.eq.aa(i)) then
            b=i
            goto 62
           endif
 56      continue
         aa(kk)=b
         b=kk
         kk=kk+1
 62     if(a.le.b)then
          write(7,111)a,b,dist,id,irec
        else
          write(7,111)b,a,dist,id,irec
        endif
        endif

        goto 66
 77     if(a.le.b)then
          write(7,111)a,b,dist,id,irec
        else
          write(7,111)b,a,dist,id,irec
        endif

        goto 66

 41     aa(kk)=a
        a=kk
        kk=kk+1
        if(a.le.b)then
          write(7,111)a,b,dist,id,irec
        else
          write(7,111)b,a,dist,id,irec
        endif
        goto 66
```

```
42      aa(kk)=b
        b=kk
        kk=kk+1
        if(a.le.b)then
          write(7,111)a,b,dist,id,irec
        else
          write(7,111)b,a,dist,id,irec
        endif
        goto 66

999     close(7)

c***** Esbalishing the Relative Code System between ********************
c***** ARC/INFO Code System and TRANSPLAN Code System ******************

        rewind(6)
c        open(7,file=fo,status='old')
        k=1
        foo='node.int'
201     format(i5,i6)
        open(8,file=foo,status='new')

        iii=1
        read(6,201)A,B
c        read(7,201)ac,bc

        if(a.gt.3000) then
          write(8,201) a,a*3000
        else
          write(8,201) iii,a
          iii=iii+1
        endif
        if(b.gt.3000) then
          write(8,201) b,b*3000
        else
          write(8,201) iii,b
          iii=iii+1
        endif

        node(k)=a
        k=k+1
        node(k)=b
551     read(6,201,end=991)a,b
c        read(7,201,end=991)ac,bc
          do 202 i=1,k
            if(node(i).eq.a) goto 203
202       continue
          k=k+1
          node(k)=a
          if(a.gt.3000) then
             write(8,201) a,a*3000
          else
             write(8,201) iii,a
             iii=iii+1
          endif
203       do 204 i=1,k
           if(node(i).eq.b) goto 551
204       continue
          k=k+1
          node(k)=b
          if(b.gt.3000) then
            write(8,201) b,b*3000
          else
            write(8,201) iii,b
            iii=iii+1
```

```
      endif
      goto 551

991   write(*,'(a)')' Phase  I: Node Conversion successfully completea
      write(*,'(a)')' Phase II: Now! Connecting Link Attribute ....'
      stop
      end
```

Appendix B: Flow Chart for Method III

Variables and notations used in this program are as follows:

a	=	ARC/INFO generated origin node of a link
b	=	ARC/INFO generated destination node of a link
a^* and b^*	=	ARC/INFO generated origin and destination nodes of a link read second time for comparison
id	=	User-ID of each link used to differentiate zonal link
θ	=	the upper limit of number of maximum zones supported by DOS-based TRANPLAN
A	=	connection point between Phase I and Phase II
s(k)	=	array for storing and sorting zone node
k	=	zone number locator
i	=	do-loop counter
eof or EOF	=	End-of-file.

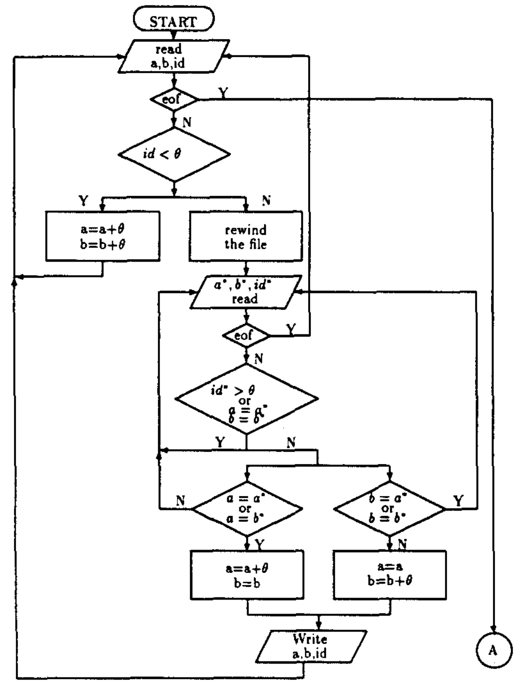


Fig.6 Detail Flow of the Topology Conversion Algorithm: Part I

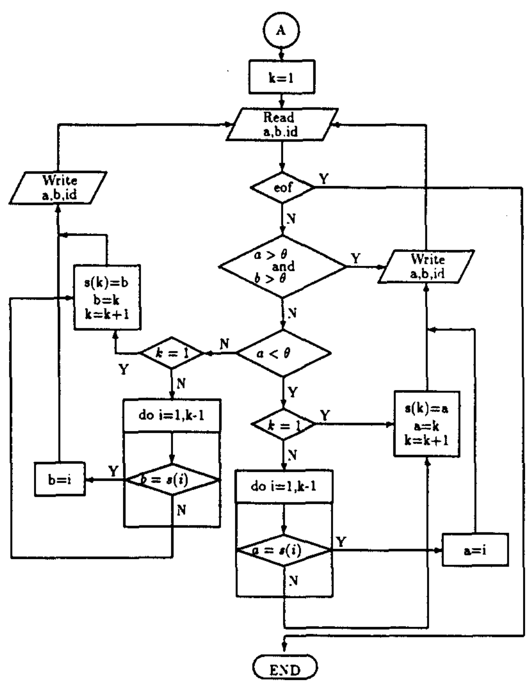


Fig.7 Detail Flow of the Topology Conversion Algorithm: Part II