An Efficient GIS Algorithm for Detecting Topological Polygonal Chains

Qiu Yonghong[1,2], Zeng Yongnian[2], Sun Guoqing[3]

[1]Central South University, China

[2]Hunan Normal University, China

[3] Hunan Provincial Communications Planning Survey and Design Institute, China

rsqyh@163.com

Keywords: Left-turning loop,Redundant edges,Polygonal chains,Nesting relationship, Backtracking algorithm.

Abstract. An efficient algorithm named Redundant Edges Backtracking Algorithm (REBA) is proposed for detecting topological polygonal chains and its advantages in time performance is analyzed and validated. Using the nesting relationship between redundant edges and polygonal chains in left-turning loops, the proposed algorithm, which solved the problem existing in available algorithms of failing to identified redundant edgeswhenextracting polygonal chains, needs only one left-turning calculation for each directed edge in the whole process of extracting polygonal chains and identifying redundant edges. Algorithm analysis and experiment results show that the time complexity of REBA is $\mathrm{O}(\mathcal{E}^2/\mathcal{N})$and close to $\mathrm{O}(\mathcal{E})$ in practical applications.

Introduction

Polygonal chain is a data structure in geographic information system (GIS) topological data model, which is used for representing the boundary of a polygonal region. Topological data model includes three basic elements of node, arc and regionusually [1,2], wherein node is located on the endpoint of arc or connection point of many arc, and it is used for describing adjacency relation among arcs; arc is a line for connecting nodes on both ends. regions(areas) are the polygons formed by the arcswhich divid the whole mapping scope and there canbetwo or more arcsina regionalboundary. Sequence formed by the arcs in accordance with connection order is called polygonal chain. Polygonal chain detection has the purpose that all polygonal chains are detected by utilizing the arc elements in topological data set. Polygonal chain detection is the key step in GIS topological data generation. It is foundations for polygon graph generation [3-5], cartographic generalization [6], buffer zone generation, polygon overlay analysis[7-9], spatial data quality inspection and other application. Improvement of the algorithm performance has very important significance to the efficiency improvement of related applications in GIS.

Algorithms for realizing polygonal chain detection can be divided into two categories roughly, namely planar graph polygon detection algorithm [10-14] and directed edge left-turning search algorithm[15,16]. The former planar graphpolygon detection algorithm is a directed graph minimum circle detection method essentially. Time and space complexities are $\mathrm{O}(\mathcal{E}^2)$($\mathcal{E}$ indicates the number of arcs), and duplicated edge and island polygons can not be handled correctly [13]. Directed edge left-turning search algorithm has lower time and space complexities without problems of handling multiple edge and island polygons [17]. It is the commonly-used polygonal chain detection algorithm.

GIS topological model requires that polygonal chain do not have dangling arcs, bridges and other redundant edges in order to guarantee data consistency and integrity. However, redundant edgesare inevitable in the topolgicaldata sets. Therefore, redundant edge must be excluded from the detection result by polygonal chain detection algorithm. Redundant edge can not be recognized and handled during acquisition of polygonal chain by existing polygonal chain detection algorithm based on directed edge left-turning search. Redundant edge should be handled firstly. Then polygonal chain can be searched then [17,18]. Left-turning calculation should be implemented on the non-redundant arc along the same direction twice at least, therefore the time cost in polygonal chain detection can be increased.

In the paper, the nesting relationship between redundant arc and polygonal chain on the left-turning search loop is utilized. A polygonal chain rapid detection algorithm - redundant edge backtracking algorithm (REBA) is proposed. The algorithm need only one left-turning calculationfor each directed edgeto completethe processingof redundant edge and obtaining of polygonal chain. Therefore, the polygonal chain detection efficiency can be prominently improved.

Related concepts

The following basic concepts are given in the section in order to facilitate algorithm description.

Definition 1 (redundant arc): the arcs not participating in composing regionboundary curve is called redundant arcs, such as dangling arcs, bridges, etc.

Definition 2 (directed edge): the arcswhich connection direction are defined called directed edges. Each arc havetwo directions: positive and negativedirections. Therefore, the given arc e should be correspondingly provided with two directed edges, and they are recorded as $\vec{e}^{+}$ and$\vec{e}^{-}$Wherein $\vec{e}^{+}$ refers to positive directed edge of e, the direction, starting and ending nodes are the same as e. The direction, starting and ending nodes of $\vec{e}^{-}$ are opposite with that of e, and it is called negative directed edge of e.

Definition 3 (left-turning directed edge): the first directed edge with $n_e$as starting point on the clockwise direction of directed edge $\vec{e}$ around the terminal node$n_e$is called left-turning directed edge of$\vec{e}$, and it is recorded as $left(\vec{e})$. When the number of incident arc of n_e is 1,left-turning directed edge of $\vec{e}$ is calledreverse directed edge. It is obvious that the left-turning directed edge of the given directed edge is unique. Namely, $left(\vec{e}) = left(\vec{e}')$only when$\vec{e} = \vec{e}'$.

Definition 4 (closed chain): the node-directed edge sequence $\mathcal{L} = (n_1, \vec{e}_1, n_2, \vec{e}_2, \cdots, n_k, \vec{e}_k, n_{k+1})$, wherein n_i、 $n_{i+1}(i = 1,2,\cdots,k)$ are respectively starting and ending nodes of$\vec{e}_i$ id called closed chain or loop when$n_1 = n_{k+1}$and, $\mathcal{L}$is called chain when $\vec{e}_i \neq \vec{e}_j (i \neq j)$. The directed edge contains information of starting and ending nodes, therefore $\mathcal{L}$ is also recorded as$(\vec{e}_1, \vec{e}_2, \cdots, \vec{e}_k)$.

Definition 5 (polygonal chain): If directed edges in the closed chain$\mathcal{R} = (\vec{e}_1, \vec{e}_2, \cdots, \vec{e}_k)$are connected in turn to form azone boundary,then $\boldsymbol{\mathcal{R}}$ is a polygonal chain. In the same polygonal chain, the left side and the right side of each directed edge are respectively close to different regions. Meanwhile, the left side and the right side of each redundant arc are close to the same region, therefore polygonal chain does not contain redundant arc.

Definition 6 (left-turning loop): closed chain is$\mathcal{P}_{\vec{e}_1} = (\vec{e}_1, \vec{e}_2, \cdots, \vec{e}_k)$, the node-directed edge sequence is$(n_1, \vec{e}_1, n_2, \vec{e}_2, \cdots, n_k, \vec{e}_k, n_1)$, if $left(\vec{e}_i) = \vec{e}_{i+1} (i = 1,2,\cdots,k-1)$ is established, $\mathcal{P}_{\vec{e}_1}$is called left-turning loop of$\vec{e}_1$.

Polygonal chain detection algorithm

Basic concept. The algorithm in the paper is realized by utilizing the nesting relationship between redundant arc and polygonal chain in the left-turning loop. The nesting relationship shows that the two directed edges of the same redundant arcdivide the own left-turning loop into two loops: nesting loop and non-nesting loop. If the left-turning loop can also contain other redundant arc, the directed edges of the same redundant arc must be in the nesting loop or non-nesting loop. If the left-turning loop has polygonal chain, the polygonal chain must be in the same loop.

Figure 1 shows that directed edge $\vec{e}_3^{+}, \vec{e}_3^{-}$ of the redundant arc e_3 divide the own left-turning loop $\mathcal{P}_{\vec{e}_1^{+}} = (\vec{e}_1^{+}, \vec{e}_2^{+}, \vec{e}_8^{+}, \vec{e}_{10}^{+}, \vec{e}_8^{-}, \vec{e}_7^{+}, \vec{e}_9^{+}, \vec{e}_{11}^{+}, \vec{e}_9^{-})$ into two loops, namely $(\vec{e}_{10}^{+})$ and$(\vec{e}_1^{+}, \vec{e}_2^{+}, \vec{e}_7^{+}, \vec{e}_9^{+}, \vec{e}_{11}^{+}, \vec{e}_9^{-})$, where the former one is nesting loop of e_3 in$\mathcal{P}_{\vec{e}_1^{+}}$. The later is a non-nesting loop. Obviously, another redundant arc $e_9$directed edge $\vec{e}_9^{+}, \vec{e}_9^{-}$ in $\mathcal{P}_{\vec{e}_1^{+}}$ is located in the non-nesting loop, polygonal chain $(\vec{e}_{10}^{+})$ is located in the nesting loop, while $(\vec{e}_1^{+}, \vec{e}_2^{+}, \vec{e}_7^{+})$ and $(\vec{e}_{11}^{+})$ are in the non-nesting loop.

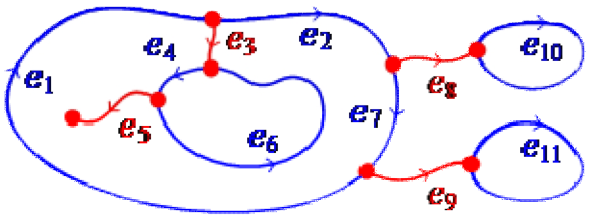

Figure 1. Topological set containing redundant arc

The nesting relationship between redundant directed edge and polygonal chain in the loop is focused. Corresponding directed edges and sub-sequence among them are deleted in turn according to the order of the second visit for each redundant arc. Each deleted loop and the finally remaining loop (if they are not empty) are polygonal chains. Therefore, the polygonal chains $(\vec{e}_{10}^{+})$, $(\vec{e}_{11}^{+})$, $(\vec{e}_1^{+}, \vec{e}_2^{+}, \vec{e}_7^{+})$can be obtained in turn through left-turning loop.

Algorithm description

The redundant edge backtracking polygonal chain detection algorithm(REBA) designed according to concept in 3.1 is shown in algorithm 2, wherein $\vec{e}_1$ refers to starting directed edge; $\mathcal{S}$refers to search chain, which is realized by linear list. It is used for recording the directed edge experienced in the search process. $\vec{e}$,$e$and i are iteration variables. $left(\vec{e})$refers to taking the left-turning directed edge of left-turning directed edge$\vec{e}_1$. $id(e)$refers to the storage position of directed edge in $\mathcal{S}$with arc of e. When $id(e) = 0$, it is obvious that $\mathcal{S}$does not contain directed edges with arcof e. $d(\cdot)$refers the storage position of all targeted directed edges.

Algorithm2. redundant edge backtrackingalgorithm polygonal chain detection (REBA)

```
01: Input starting directed edge $\vec{e}_1$,initialize searchchain $\mathcal{S}$, $\vec{e} \leftarrow \vec{e}_1$
02: do
03:    $\vec{e}$is marked as ‘being visited, $e \leftarrow e(\vec{e}), i \leftarrow id(e)$
04:    if($i = 0$)then
05:       $\vec{e}$ is added at the end of $\mathcal{S}$,$id(e) \leftarrow \mathcal{S}$ in the$\vec{e}$ is subscript
06:    else
07:       One polygonal chain is formed by directed edges in $\mathcal{S}$ with subscript>$i$.
08:       $id(\cdot) \leftarrow 0$aiming at art section in $\mathcal{S}$, with subscript≥$i$.
09:       Delete subscript≥$i$directed edges in $\mathcal{S}$
10:    end if
11:                                   $\vec{e} \leftarrow left(\vec{e})$
12: while ($\vec{e} \neq \vec{e}_1$)
13: One polygonal chain is formed by directed edges in $\mathcal{S}$
14: $\mathcal{S}id(\cdot) \leftarrow 0$ aiming at all art sections in $\mathcal{S}$
```

Algorithm 2 is a iterative process. Ending condition is$\vec{e} = \vec{e}_1$, namely it can be returned to the starting edge. Algorithm 2 only can be used for detecting the polygonal chain in one left-turning loop. Any one directed edge should be selected from each left-turning loop as a starting edge for searching respectively in order to obtain all polygonal chains. Therefore, the visited $\vec{e}$ in the algorithm 2 should be marked as ‘being visited’ in order to avoid repeated search. Meanwhile, algorithm 3 can be adopted, and un-visited directed edges can be selected as starting edge for searching.

Algorithm 3. Starting directed edge selection

```
01: Input topological set $\mathcal{T}$
02: for each $e$ in $\mathcal{T}$
03:    if ($\vec{e}^{+}$ is not visited), then the polygonal chain is detected from$\vec{e}^{+}$
04:    if ($\vec{e}^{-}$ is not visited), then then polygonal chain is detected from $\vec{e}^{-}$
05: end for
```

Performance analysis

In algorithm 2, since operation on search chain $\mathcal{S}$mainly includes record insertion and deletion at the end, if $\mathcal{S}$is realized by array, one record time complexity O(1) is inserted at the end. Similarly, if one temporary variable is used for recording the storage position of each arc in the search chain$\mathcal{S}$, then, we should check whether $\mathcal{S}$ contains the time complexity of one directed edges of O(1) or not.

Therefore, the time complexity of the algorithm in the paper is mainly determined by$left(\overrightarrow{\ })$, If node associated arc is represented by data structure which is the same as the directed edge, the associated arc section list is stored and taken by one 1-dimensional array. They are sequence according to the inclined angle of the connecting end with the positive east direction on the clock-wise direction. The associated arc section list on the terminal node is set as ***I*** aiming at the directed edge$\overrightarrow{\ }$, the subscript of the left-turning directed edge in the ***I*** is$(i + 1)\%m$, wherein % is complementation. $m$refers to the size of$\boldsymbol{I}$, refers to subscript of $\overrightarrow{\ }$ in . Therefore, the time cost of $(\overrightarrow{\ })$ is mainly used for determining subscript of $\overrightarrow{\ }$ in . Since is not large generally, sequence search is generally adopted for determining . Therefore, the associated arc average comparison frequency for realizing $(\overrightarrow{\ })$ is$(+ 1)/2$. Topological data set $\mathcal{T}$ is given, average value $\overline{\ } = 2\ \ /$ (□,□ of □ is respectively arc section quantity and node quantity), and the average comparison frequency of associated arc section for realizing $(\overrightarrow{\ })$ is shown as follows:

$$\overline{\mathrm{C}} = \frac{1}{2}\cdot\left(\overline{\ } + 1\right) = \frac{1}{2}\cdot\left(\frac{2}{\ } + 1\right) = - + \frac{1}{2} \qquad (1)$$

It is function of □ and □. Therefore, the comparison calculation frequency of associated arc section for the algorithm in the paper is shown as follows:

$$(\ , \ ) = 2\ \left(- + \frac{1}{2}\right) = 2\ \ ^{2}/\ \ + \qquad (2)$$

Therefore, the progress time complexity of the algorithm in the paper is$\mathrm{O}(\ ^{2}/\ )$.

When topological data set □ only has one node, namely $= 1$, $(\ , \ ) = 2\ \ ^{2} +$; When all arc sections are not adjacent mutually, namely , $= 2$, $(\ , \ ) = 2$, time complexity of the algorithm in the paper had better be O( ). It is obvious that the time complexity of left-turning loop acquisition algorithm had better beO( ), the worst condition is$\mathrm{O}(\ ^{2})$.

The storage space for the algorithm in the paper is mainly used for storing all experienced directed edges in the search chain □, the size depends on directed edge quantity in left-turning loop, which does not exceed 2□ (□ refers to the total number of arcs). Therefore, the space complexity is O( ).

Experiment results

The experiment is divided into 2 groups, which are respectively used for testing the relationship of algorithm time consumption with arc section and node quantity. The testing environment is Intel Core i7-4770@3.40GHz four-core CPU, 12GB memory, Windows 8.1 64bit operation system and Visual C++ 2013 compiler.

Figure 2 shows the relationship experiment result between time cost and arc quantity □ of algorithm in the paper. The experiment data include nine 9 data sets, node quantity □ is $1.0\times10^{6.}$ The associated arc section quantity of each node in the data set is the same. Obviously, when the number of nodes is fixed, the time cost □ of algorithm in the paper is secondary function (R^2=1) of arc quantity□. The fitting result shows that the quadratictermcoefficient is very small compared with monomial coefficient (the ratio thereof is up to 10^{6}). Therefore, it can be believed that the time cost □ of algorithm in the paper is arc section quantity□,which is close to the linear relationship.

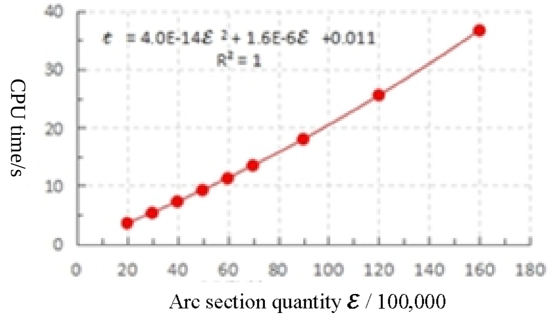

Figure 2. Relationship between algorithm time consumption and arc section quantity

Figure 3 refers to the relationship test result between time cost and node quantity □ of the algorithm in the paper. 6 arc section quantity □=6.4×10^6 data set is adopted for the test data. Similarly, the associated arc section quantity of each node in the same data set is the same. Since □ is fixed, we adopt arc section node quantity ratio □/□ as independentvariable in order to reflect the time efficiency of algorithm in the paper more clearly. Obviously, the algorithm time cost □ is the linear function of arc-node quantity ratio(/),namely when □ is fixed, the time cost and node quantity of the algorithm in the paper is reversely proportional.

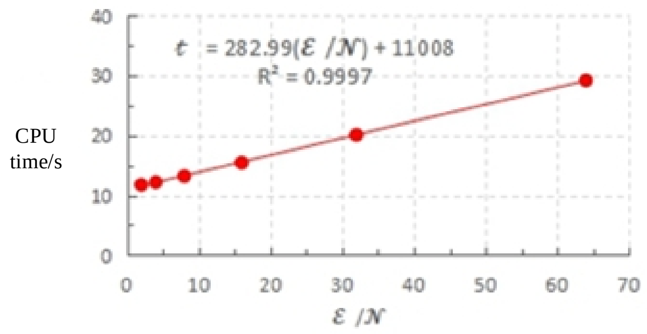

Fig.3 The relationship between time costs and the number of nodes

In summary, it is not necessary to implement additional redundant edge recognition and treatment before polygonal chain detection according to algorithm in the paper. Left-turning calculation should be implemented on all directed edges once.

Conclusion

A rapid algorithm-redundant edge backtracking algorithm (REBA) based on directed edge left-turning is proposed in the paper aiming at polygonal chain detection which requires redundant edge processing in the GIS topological treatment. Time and space performance of utilizing the algorithm for polygonal chain detection is analyzed. The performances are validated through experiments.

The experimental results show that the time cost of utilizing new algorithm in the paper for polygonal chain detection is the arc in the topology data set, which is related to the number of nodes. When the number of nodes is fixed, total time cost of algorithm is the secondary function of arc section quantity. Whenthe number ofarcs is fixed, the time cost shows linear relationship with the arc-node quantity ratio in the topology data set, the number of nodes is larger, the time cost is lower. When the arc-node quantity ratio is constant, total time cost of algorithm is gradually increased linearly with the increase ofthe total number ofarcs. Since arc-node quantity ratio is generally closer to a smaller constant in the actual topological set, new algorithm time complexity generally has linear relationship with the total number of arcs in the topological set, thereby it has very prominent efficiency advantage in applications.

Acknowledgments

The paper is one of staged achievement of ‘Civil-military Integration Application Demonstration in Highway Traffic Field’ (GFZX0404080102).

References

[1]Peucker T K, Chrisman N. Cartographic Data Structures. The American Cartographer,1975,2(1):55-69.

[2]Theobald D M. Topology Revisited: Representing Spatial Relations. International Journal of Geographical Information Science, 2001,15(8):689-705.

[3]Siejka M, Ślusarski M, et al. Correction of Topological Errors in Geospatial Databases. International Journal of Physical Sciences, 2013,8(12):489-507.

[4]Pueyo O, Patow G. Structuring Urban Data. The Visual Computer, 2013,29(4):1-14.

[5] RONG Yuecheng. A Dynamic Algorithm of Vectorization for Large Classified Remote Sensing Image. ActaGeodaetica et Cartographica Sinica,2012.41(6):898-903.

[6] ZHANG Lan, LI Jiatian, XU Heng, et al. Polygon Growing Algorithm for Network Schematic Maps. ActaGeodaetica et Cartographica Sinica,2015,44(3):346-352.

[7]LIU Shuhua, ZHOU Yunyan, CAO Liqiang. An Inner Hole Deployed Polygon Combination Algorithm Based on Vector Wander Method. Microcomputer Applications, 2011, 16(06):1-7.

[8]XIE Zhong, WEI Dongqi, WU Liang, et al. Graph Model of Polygon Clipping Using Simple Vector Data. ActaGeodaetica et CartographicaSinica, 2009,38(04):369-374.

[9]FAN Junfu, KONG Weihua, MA Ting, et al. RaPC: A Rasterization-based Polygon Clipping Algorithm and Its Error Analysis. ActaGeodaetica et CartographicaSinica, 2015,44(3):338-345.

[10]Amato N.M., Goodrich M.T., Ramos E.A.. Computing Faces in Segment and Simplex Arrangements, Proc. 27th Annual ACM Sympos. Theory Comput.. ACM, 1995:672-682.

[11] Asano T, Guibas L J, Tokuyama T. Walking in an Arrangement Topologically. International Journal of Computational Geometry & Applications, 1994,4(02):123-151.

[12]Ferreira A, Fonseca M J, Jorge J A. Polygon Detection from a Set of Lines, Proc. 27th EncontroPortugues de ComputacaoGrafica. Porto, Portugal, 2003:159-162.

[13]Johnson D. Finding All the Elementary Circuits of a Directed Graph. SIAM Journal on Computing, 1975,4(1): 77-84.

[14]McKenney M. Region Extraction and Verification for Spatial and Spatio-temporal Databases, Scientific and Statistical Database Management. Springer Berlin Heidelberg, 2009: 598-607.

[15]Chen Chun, Zhang Shuwen, et al. The Basis for Generation of Topologic Information of Polygons in GIS. ActaGeodaetica et CartographicaSinica, 1996.25(4):267-271.

[16]QIU Yonghong, ZENG Yongnian, ZOU Bin. Improved Algorithm for Searching GIS Topological Polygonal Chains[J]. Computer Engineering and Applications, 2012, 48(34): 23-27.

[17] ZHANGXiaocan, HUANG Zhicai, et al. A Fast Algorithm for Distinguishing the Ascription of Islands and Label Points in GIS[J]. Chinese Journal of Computers, 2005,28(3):343-349.