Polygon Processing on OpenStreetMap XML Data

Fangli Ying, Peter Mooney, Padraig Corcoran and Adam C. Winstanley

[1]Department of Computer Science, National University of Ireland Maynooth, Co. Kildare. Ireland
Tel. +353 1 62801011
fying@cs.nuim.ie, peter.mooney@nuim.ie, padraigc@cs.nuim.ie adamw@cs.nuim.ie

KEYWORDS: OpenStreetMap, Polygon extraction and analysis

1. Introduction

One of the many ways of accessing the raw geographic data collected and distributed by OpenStreetMap is by using the OpenStreetMap (OSM) XML data format. The OSM XML can be explored using standard XML visualisation and search tools. This paper describes the development of a software tool for the examination of polygons represented in OpenStreetMap XML. Given an OSM XML file corresponding to a specific geographical area the software performs the following two functions during polygon examination:

1. *Examination of the connectedness of water features*. The OSM XML is checked to ensure that water features are correctly connected and are consistent with the physical reality. For example a river or stream flowing into a lake. Potential spatial connectivity problems are highlighted and can be used for quality control purposes for the OpenStreetMap data.
2. *Automated identification and selection of suitable sets of polygons as input for testing generalisation algorithms*. This software automatically identifies suitable sets of polygons for testing generalisation algorithms by calculating the overall complexity for each polygon in a specified geographical region. If this overall complexity is high then all of the features are extracted from the OSM XML and output in ASCII format to the generalisation algorithms.

The software is written in Java. Figure 1 provides a schematic of the individual components in the software tool. An OSM XML file is presented as input. Firstly connection issues in the spatial data are identified and a report is generated on these issues. An OSM community member can then use this information to update the OSM database to address these issues. Secondly the tool computes a number of polygon characteristics for each of the polygons in the OSM XML file. If the set of polygons are identified as sufficiently complex they are reformatted into ASCII file format where they are passed to the generalisation module. This is a separate software tool that runs a generalisation algorithm on this spatial region and is not described in this paper. A local database copy of OSM is then updated automatically with the generalised representations of the polygons inserted. The polygons in the OSM XML are captured in a number of ways: from GPS data collections, tracing over aerial imagery, and bulk upload of spatial data to OpenStreetMap.

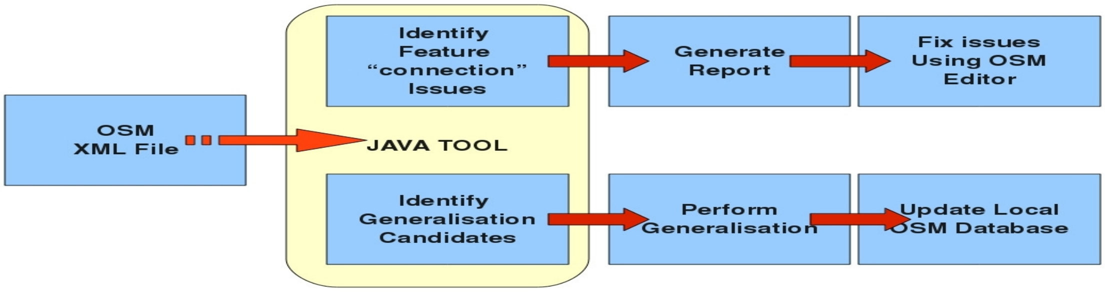

OSM XML File
JAVA TOOL
Identify Feature "connection" Issues
Generate Report
Fix issues Using OSM Editor
Identify Generalisation Candidates
Perform Generalisation
Update Local OSM Database

Figure 1. Schematic Diagram showing components of the software tool for polygon processing

2. Spatial Representation in OpenStreetMap XML

While the OSM XML schema is easy to understand, it is very difficult to identify geographic features which are connected to each other without visualising the XML as a tree-based structure or rendering the data as a set of base map tiles or an overlay on another map. The problem is particularly difficult when the OSM XML represents (1) a very large geographical area, and/or (2) contains a very large number of lines and polygons. OSM XML contains points, lines, and polygons. Every spatial attribute (or *tag*) corresponding to each point, line, or polygon feature is included in the XML. It was a specific requirement of this software to operate directly on the OSM XML by downloading OSM XML *on-the-fly* from the Internet and performing the processing in real-time. Consequently, there is no need for setup of spatial databases or desktop GIS. Data in OpenStreetMap is contributed by members of the OSM community. In the OSM database there are many features, sometimes simple polygons, which are often greatly over-represented when considering them for applications such as Location-based Services (LBS). This means that the GPS traces uploaded to OSM for these features were sampled at a very high rate. Regular shaped polygons (squares, rectangles, triangles) representing buildings, car parks etc often contain many hundreds of points. More complex polygon shapes, such as those representing natural features such as lakes are often represented by many thousands of points. Given the characteristics of the polygon this is often necessary. However in some cases the number of points used to represent the polygon could be reduced. Relationships between points and polygon/line features are represented simply in OSM XML. Each point in OSM has a unique ID (OSM_ID). Each polygon/line in OSM also has a unique ID (OSM_ID). In the OSM XML each polygon/line is represented as a <way> with a unique OSM_ID. Each <way> is a ordered collection of <node> features with unique OSM_ID.

3. Examination of the connectedness of water features in OpenStreetMap.

Given a feature class the software automatically identifies polygons (belonging to that particular feature class ie. waterbodies, forest and woodland, etc) in the area specified in the OSM XML file. The software identifies all other line and polygon feature intersections. Based on specified rules for spatial feature connectivity (river flows into lake, steam flows into river) it determines features not correctly connected within the OSM XML. Figure 2 shows an example from the OpenStreetMap database correct as of Feb 25th 2010. Lough Erne in County Fermanagh, Northern Ireland is shown. Lough Erne (the Upper and Lower loughs) are actually widened sections of the River Erne. The representation in OpenStreetMap is incorrect where the two loughs are represented as two distinct unconnected features. The physical reality is that Upper lough flows into the Lower lough. For water features our software computes the nearest nodes between pairs of polygons. If it is found that this distance is less than a pre-determined value we then infer that these features should be physically connected. For example a river or stream separated by less than 20 meters from a lake. Potential errors in representation are then reported to allow OpenStreetMap contributors to address the issues. For natural landscape features such as rivers and lakes this software could assist in improving the quality of the OpenStreetMap database by suggesting improvements in terms of feature connectivity.

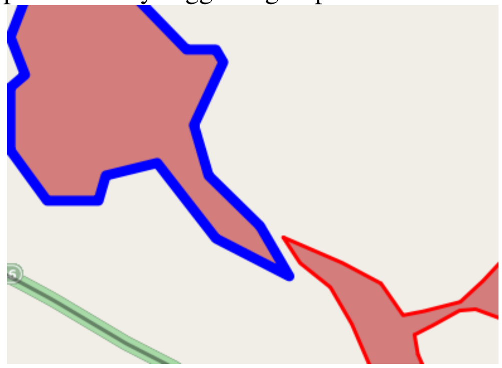


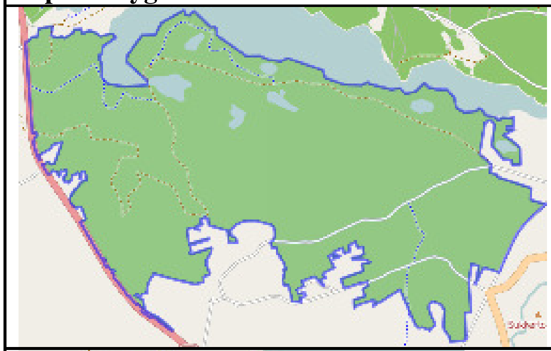
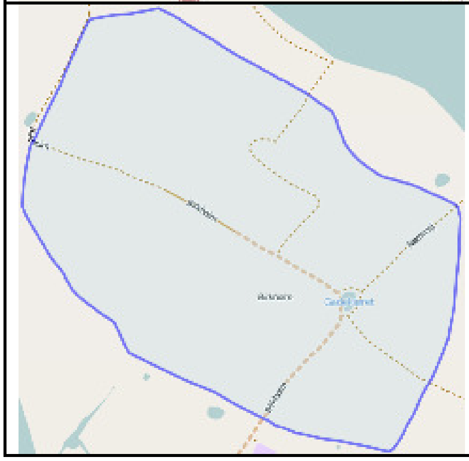
Figure 2. Visual Representation of apparently connected polygon features

4. Automated identification and selection of suitable sets polygons as input for testing generalisation algorithms.

In our software we have implemented a strategy to identify suitable sets of polygons for use as input to testing generalisation algorithms. The software attempts to calculate characteristics of each polygon which can be used to give an overall measurement of the complexity of polygon. Four principal characteristics are outlined as follows which together can help to automatically describe the complexity of a polygon without human visual evaluation of the complexity (Bryson and Mobolurin 2000). The software attempts to estimate the polygon complexity in the smallest number of steps which are outlined as follows:

1. **N:** The number of points representing the polygon
2. **Turning angle (k)** at each node. The overall measure of turning angle is the mean of all k values denoted as K. The greater the variation in turning angle (large K values) the more complex the polygon is likely to be (Latecki and Lakmper,1999)
3. **Cicularity:** Simple circularity measure - ratio of the square of the perimeter length to the area (normalised between 0.0 and 1.0)
4. **Area Ratio:** Convex Hull circularity – normalised ratio of difference between area of polygon and its convex hull. Larger ratios (close to 1.0) can indicate a complex polygon.

Other measurements are calculated including: distribution of normalised distances between adjacent nodes in the polygon, width of the polygon, and distribution of normalised distances between every vertex the centroid of the polygon. This purpose of the steps above is to allows us to select suitable test data as input. Most of the time only a subset of the steps will be required for the software to check if the polygon is complex or not (Brinkhoff *et al*, 1995). Table 1 below shows some examples of the output from our software for some singular polygons input.

Input Polygon	**Class**	**Rationale**
	**Complex**	**Generalise = Yes:** 559 nodes, Circularity = 0.285, Area Ratio = 0.255. K = 0.0018. Very low K value means that this shape could be generalised as some vertices could be removed without dramatically altering the structure of the polygon.
	**Simple**	**Generalise = No:** 40 nodes, Circularity = 0.246, Area Ratio = 0.097. K = 0.0256. This polygon is very circular. There is a small number of nodes. The K value is very high meaning that all nodes have very high significance.

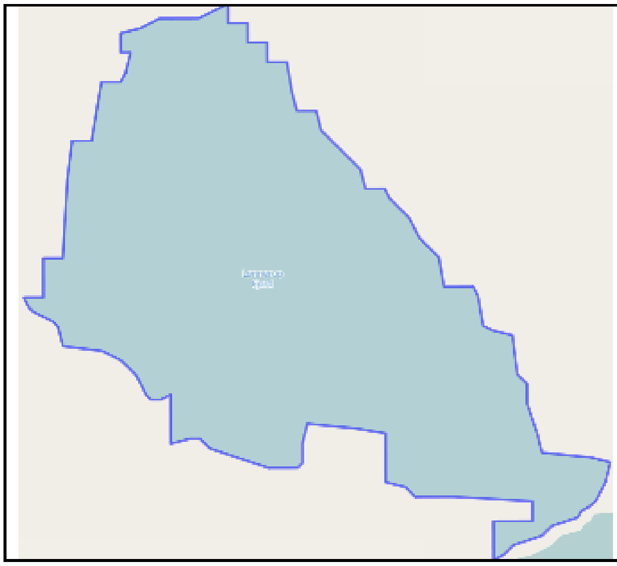
	**Simple**	**Generalise = Yes:** 177 Nodes, Circularity = 0.311, Area Ratio = 0.251. K = 0.0057. The K value for the polygon is very low meaning that many insignificant nodes exist in the polygon. These could be removed by generalisation.

Table 1Example of complexity analysis results for three sample polygons.

5 Conclusions

The software tool described has been developed to (1) identify spatial data quality issues for connected polygon features in OpenStreetMap and (2) select suitable sets of polygons from OpenStreetMap for use in the testing of generalisation algorithms. The software has been developed to work on small geographical regions – approximately 10Km2. It is guided by the user who provides specifies the region that may have problems in relation to the connectivity of polygon features. This software will assist the authors in identifying these connectivity issues in the OpenStreetMap database. Problems encountered can be fixed using one of the many OpenStreetMap data editors available. Automated identification of polygons for testing generalisation algorithms is a very interesting topic and the algorithmic approach described is at an early stage. Human visual perception can identify complex polygons very quickly when presented with a cartographic representation of the polygons (Brinkhoff *et al*, 1995). Our software tool aims to identify suitable generalisation candidates quickly from the OSM XML provided as input by calculating a suitable subset of the polygon characteristics outlined above. Several characteristics of the polygons must be calculated in order to automatically assess the complexity. For applications such as Location-based Services (LBS) the spatial data in the OpenStreetMap database is one where generalisation will always be needed as new polygon features are added and existing polygon features are edited. We will continue to investigate other methods for shape complexity identification particularly from the fields of computer vision and shape recognition. This paper does not provide details on validation of the results from the generalisation algorithms employed. This work is described in other papers by the authors.

Acknowledgements

Fangli Ying is funded by Irish Universities Association and China Scholarship Council. Peter Mooney is funded by the STRIVE Research Programme administered by the Environmental Protection Agency of Ireland. All of the authors are part of the Location-based Services (LBS) Strand under Science Foundation Ireland's STRAT-AG funding at the National Center for Geocomputation (NCG) at NUI Maynooth. Adam Winstanley is co-PI of the LBS Strand.

References

T. Brinkhoff, H.-P. Kriegel, R. Shneider, A. Braun, *Measuring the complexity of spatial objects*, Proceedings of the 3rd ACM International Workshop on Advances in Geographic

Information Systems, Baltimore, MD, 1995, pp. 109–117

N. (Kweku-Muata) Bryson, A. Mobolurin, *Towards modeling the query processing relevant shape complexity of 2D polygonal spatial objects*, Information and Software Technology, Volume 42, Issue 5, 1 April 2000, Pages 357-365

L. J. Latecki and R. Lakmper, *Convexity rule for shape decomposition based on discrete contour evolution*, Computer Vision and Image Understanding, vol. 73, no. 3, pp. 441 – 454, 1999.

Biography

Fangli Ying is a first year PhD student at the Department of Computer Science. He holds a degree in Computer Science. He commenced his PhD in October 2009. Peter Mooney is a postdoctoral research fellow at the Department of Computer Science and at the Environmental Protection Agency of Ireland. Padraig Corcoran is a lecturer at the Department of Computer Science with research interests in machine learning, pattern matching, and computer vision. Adam Winstanley is head of the Computer Science Department and is co-PI for the Location-based Services strand of the Science Foundation Ireland STRAT-AG project at the National Center for Geocomputation at NUIM