SoftwareX 12 (2020) 100553

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Geo-MST: A geographical minimum spanning tree plugin for QGIS

Murat Çalışkan [a,b], Berk Anbaroğlu [b,*]

[a] Department of Information Technologies, Ministry of Agriculture and Forestry, Ankara, Turkey
[b] Department of Geomatics Engineering, Hacettepe University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 21 April 2020
Received in revised form 14 June 2020
Accepted 15 June 2020

Keywords:
Kruskal's algorithm
Network analysis
GIS
Open-science

ABSTRACT

Graphs describing the relation between nodes and edges are common in geographic information science. One of the algorithms that operate on graphs is 'Minimum Spanning Tree (MST)', which is a tree that connects all the nodes of a graph with minimum cost. There is no built-in functionality in QGIS, an open-source Geographical Information System (GIS) software, which can determine MST. This paper proposes a QGIS plugin that determines MST on geographical data using Kruskal's algorithm. The updated version of the plugin (v2.0) offers three substantial improvements with respect to its former version (v1.0). First, the updated version is much faster in execution. The execution time of the two versions was assessed by determining MST on a randomly generated dataset consisting of 5000 polygons and New York City's census blocks consisting of 38799 polygons. The updated version determined MSTs much faster, reaching up to 30-fold improvements. Second, the updated version can handle raster data. In this way, researchers might consider continuous geographical characteristics while estimating the costs of edges in addition to the discrete measure distance. Third, a barrier (obstacle) might be provided to ensure that the MST is fit for purpose as political boundaries or other restrictive socio-economic issues can be considered.

Code metadata

Current code version	*v2.0*
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX_2020_177
Legal Code License	*GPLv2*
Code versioning system used	*git*
Software code languages, tools, and services used	*Python*
Compilation requirements, operating environments & dependencies	*The software runs on QGIS 3.x* (https://www.qgis.org)
If available Link to developer documentation/manual	https://github.com/banbar/Minimum_Spanning_Tree_QGIS
Support email for questions	caliskan.murat.20@gmail.com, banbar@hacettepe.edu.tr

1. Introduction

A Geographical Information System (GIS) software is used to store, analyse and visualise geographic data. Many different research areas, ranging from field research [1] to site selection, that require multiple criteria decision-making [2] benefit from GIS software. QGIS is an open-source GIS software developed by using C++ and Qt library that allows plugin development to automate geographical analysis [3]. There are many plugins for QGIS, ranging from groundwater vulnerability assessment [4] to landscape ecological analysis [5].

Graphs are data structures that represent the relation between nodes and edges. Many different research areas that are related to complex networks including, but not limited to, biological networks and computer networks rely on graphs to model data. Location of nodes and edges play a key role in analysing graphs involving a geographical component [6]. Such graphs have been

* Corresponding author.
E-mail addresses: caliskan.murat@tarimorman.gov.tr, muratcaliskan@hacettepe.edu.tr (M. Çalışkan), banbar@hacettepe.edu.tr (B. Anbaroğlu).

https://doi.org/10.1016/j.softx.2020.100553

Table 1
Comparison of different software that can determine MST.

Software	I/O file formats	Interface	License	Last update	Total downloads	Geographic data
Peklo[a]	xml	GUI	GPLv2	10/2015	1802	✗
MSTgold[b]	txt, fasta, mstg, dot	CLI	N/A	02/2017	398	✗
Grapho[c]	Interactive editing	GUI	N/A	10/2016	552	✗
PHYLOViZ[d]	txt, fasta, tab	GUI	GPLv3	03/2019	23 329	✗
'MST' plugin in QGIS[e]	shp	GUI	GPLv2	04/2020	710	✓

[a] https://sourceforge.net/projects/peklo.
[b] https://sourceforge.net/projects/mstgold.
[c] https://sourceforge.net/projects/grapho.
[d] http://www.phyloviz.net/.
[e] https://plugins.qgis.org/plugins/minimum_spanning_tree/.

investigated for different purposes ranging from path planning for driverless buses [7] to modelling landscape changes [8].

One of the common algorithms that operate on graphs is determining the 'Minimum Spanning Tree (MST)', which is defined as the tree connecting all the nodes of a graph with the lowest cost [9,10]. Being one of the first network optimisation problems, efficient calculation and visualisation of MST still take attention of researchers. The common use of MST in a GIS environment (Geo-MST) motivated researchers to develop software for its identification. Specifically, some of the researchers developed their solutions in Matlab [11] or utilised Python to develop extensions on top of proprietary GIS software such as ArcGIS [12]. However, these solutions cannot be widely used as their source code is not openly available.

The aim of this paper is to describe the version 2.0 (v2.0) of the Geo-MST plugin that is readily available on QGIS and analyse its performance with its preceding v1.0 [13]. The advantages of the proposed plugin include:

- handling continuous geographic information in the form of raster data,
- handling discrete geographic information in the form of point, line or polygon data,
- increased performance compared to v1.0 in terms of execution time (23x–30x), and
- considering barriers while determining MST.

2. Problems and background

Determining and visualising MST is a common task in many research areas ranging from the optimal dynamic reconfiguration of power distribution networks [11] to investigating chemical structures and their associated properties [14]. Consequently, various readily available software can determine and/or visualise MST on an input graph. These solutions are compared in Table 1.

Apart from the Grapho, which was developed for education, other software have probably been used outside academia for both practical and research purposes. Most of the investigated software possess a Graphical User Interface (GUI) since users might find it difficult to rely on a Command Line Interface (CLI) to process their data. Peklo and PHYLOViZ were developed not only for determining MST but also for many other tasks including flow analysis or hierarchical clustering. PHYLOViZ, the most downloaded software as of 8 April 2020, was developed to advance genomics research [15]. Similarly, MSTgold was developed to advance computational biology, and it was optimised to determine hundreds of MSTs [16]. Amongst the readily available solutions, only the 'Minimum Spanning Tree' plugin of QGIS can handle geographic data.

There are two main types of geographic data: vector and raster. Vector data represent discrete geographic information such as trees in a forest, city borders, lakes, roads, points-of-interest etc. There are three main types of geometries to represent

Table 2
Comparison of the two most recent versions of the MST plugin.

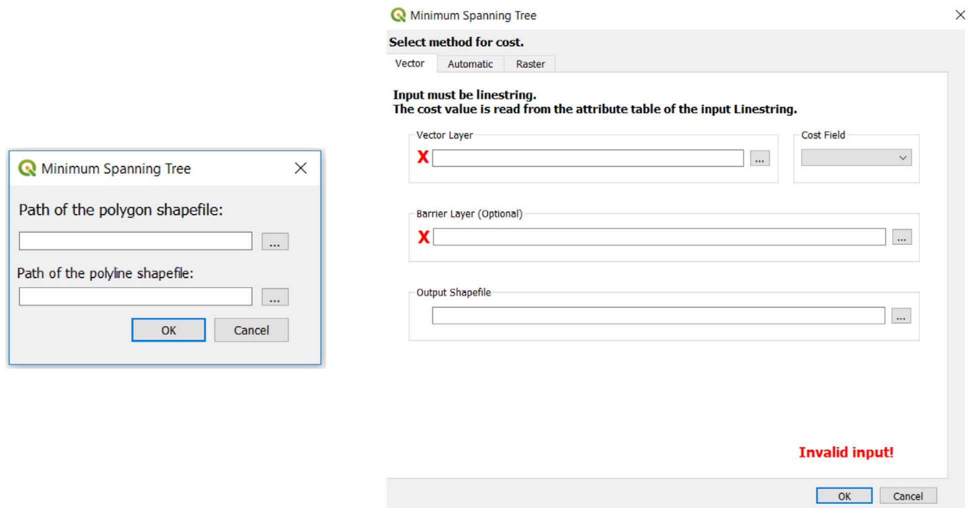
	v1.0	v2.0
Geometry		
Point	✗	✓ (DT)
Line	✓	✓
Polygon	✓ (adjacency)	✓ (DT on centroids)
Raster	✗	✓
Barrier	✗	✓
Library	PyQGIS	GDAL/OGR
GUI	✓	✓✓
Release date	16 October 2019	14 April 2020
Total downloads (24 May 2019)	295	143

vector data: point, line and polygon. Geometric measures like distance between two points or area of a polygon can be calculated on vector data. On the other hand, raster data are images composed of pixels and represent continuous geographic information. Raster data can be collected from earth observation satellites, unmanned aerial vehicles, planes etc. The main advantage of relying on raster data is that it provides a more reliable cost estimate rather than simply relying on discrete measures like distance. Specifically, different continuous information such as Digital Elevation Model (DEM), slope, land cover etc. can be considered to provide a better cost estimation model [17].

The v1.0 of the QGIS plugin 'Minimum Spanning Tree' (Fig. 1a) was developed as a part of the undergraduate course entitled 'GIS Programming' delivered at the Geomatics Engineering Department of Hacettepe University to the 2018–19 cohort [13]. The v1.0 of the MST plugin had several shortcomings. First, it could only operate on lines and polygons, whereas geographic data may also include point data or raster data. Second, the execution time to determine the MST increased substantially when the input size got larger. This might be due to the reliance on PyQGIS functionality when conducting the spatial operations such as finding the centroids of polygons. In v1.0, an edge is generated between the centroids of two polygons, whereas the proposed version relies on the Delaunay Triangulation (DT) [18]. Third, political or socio-economic boundaries that are in the form of a barrier (obstacle) have not been considered. In this way, the MST might pass through a restricted region. The proposed v2.0 was designed to overcome these limitations. The overall comparison of these two versions is summarised in Table 2.

3. Software framework

The proposed v2.0 determines the Geo-MST of input data and overcomes the limitations of its preceding version. The plugin is readily available, and can be installed to QGIS with the following steps:



(a) (b)

Fig. 1. Graphical User Interface of the 'Minimum Spanning Tree' plugin in QGIS: v1.0 (a) and v2.0 (b).

1. Navigate to 'Plugins' and select 'Manage and Install Plugins'.
2. Navigate to the 'Settings' tab, enable 'Show also experimental plugins'.
3. Navigate to the 'All' tab. Search for 'Minimum Spanning Tree', and install the plugin.
4. The plugin's symbol, , will appear on the toolbar.

Supported input vector layer formats include shapefile (SHP), which is a common format in GIS software, GeoJSON and Geography Markup Language (GML). The Tagged Image File Format (TIFF) is supported for raster layers. The output layer is the MST, which is saved as a line shapefile. Once the plugin is executed, the GUI illustrated in Fig. 1b welcomes the user. The GUI possess three tabs, which correspond to different ways to determine the costs of edges based on the input geographic data. Regardless of this choice, Kruskal's algorithm is used to determine the MST [19]. First, edges are sorted in ascending order of their cost. Second, an edge is included in the MST if its inclusion does not form a cycle. Otherwise, the next edge is considered for inclusion. The second step is repeated until all nodes of the graph are connected.

3.1. Software architecture

The developed plugin is written in Python and utilised PyQt library to develop the GUI. In addition, core geographic analysis libraries 'Geographic Data Abstraction Library' (GDAL)[1] and 'OpenGIS simple features Reference implementation' (OGR) are used to increase the performance of the geographic analysis. These libraries are released by Open Source Geospatial Foundation (OSGeo) [20] under an X/MIT style Open Source license.

The overview of the plugin and a simplified function diagram are illustrated in Fig. 2a and b respectively. The function diagram represents only some of the functions used in the plugin, where these functions are used for various purposes ranging from obtaining the spatial reference of input geographic data (i.e. GetSpatialRef) to reading raster data as a Python NumPy array object (i.e. ReadAsArray). The comprehensive list of OSGeo functions and their definitions are presented in the plugin's GitHub page, which can be found in the code metadata table.

Three tabs correspond to different ways to obtain the costs of edges as illustrated in Fig. 2a. The first tab, 'Vector', accepts a line layer, which is the natural way to represent a graph with a geographical component. The plugin then asks the user to select the cost attribute. Consequently, the first tab assumes that the costs of edges are available within the input.

The second tab, 'Automatic', does not assume the availability of a cost field, and calculates the costs based on the length of edges. If the input is in the form of line geometry, then the length of the lines are calculated to obtain the costs. However, an Edge Generation (EG) step must be accomplished if the input is in point or polygon geometries. DT is used to convert input points to a graph [18], where no edge cuts through another edge and all points are connected. If the input is in polygon geometry, first, the centroids of these polygons are identified, and then DT is applied to obtain the graph. The length of the edges in the constructed DT are assumed to be their cost.

The third tab, 'Raster' is the most complicated form of analysis as it combines all of the possible input geographic to estimate the cost of edges. Specifically, users can provide points or polygons to generate edges by using DT as described previously or input a line layer. Instead of only relying on the length of edges as in the second tab, raster data is used to estimate the costs of edges. Three different methods are proposed to estimate the edge costs. These methods are described in the next subsection.

The users may optionally provide barrier data under all of the three tabs. If an edge of the input graph intersects with a line present in the barrier layer, then that edge is removed from the analysis. Barriers, represented as lines, are useful in different scenarios since MST might be restricted to be within only a specified region due to political or socio-economic reasons. For example, when re-establishing a traffic lights' network, an operator might restrict the passage of MST underneath roads to avoid road closures. In this case, traffic lights would be represented as points and the road network (i.e. barrier) as lines.

3.2. Cost estimation using raster data

Raster tab conducts the most complicated analysis amongst others as it utilises information about the earth's surface, such as elevation or slope to estimate the costs of edges. Different methods can be used to estimate the costs of edges. The developed plugin proposes three simple cost estimation methods: sum, average and hybrid. An exemplar scenario to estimate the costs of edges on a grid of 10×8 pixels (cells) with six edges are illustrated in Fig. 3a. The values stated in each cell corresponds

[1] https://www.osgeo.org/projects/gdal/.

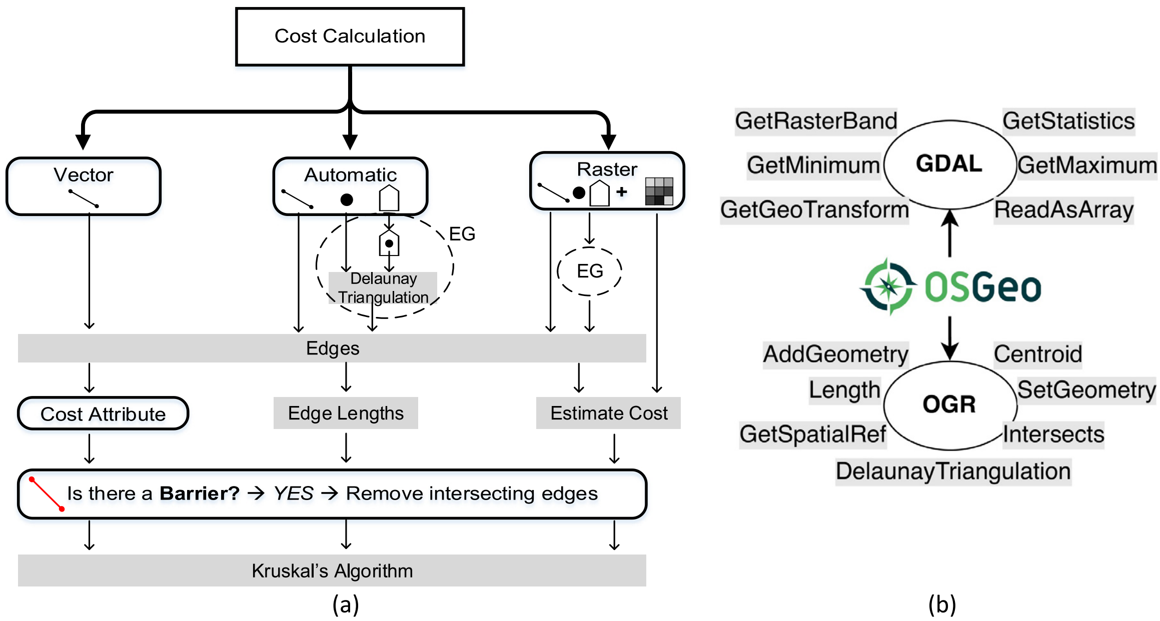
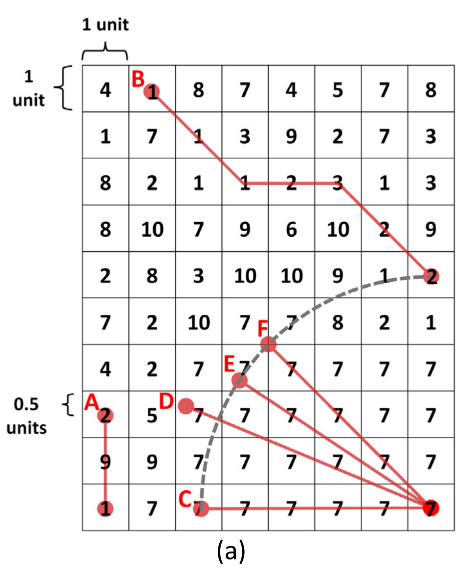


Fig. 2. Overview of the plugin (a) and the module diagram (b).



Edge	Length	Cost Estimation Methods: Sum	Average	Hybrid (Length * Average)
A	2	12	4	8
B	7.7	12	1.7	13.1
C	5	42	7	35
D	6	56	7	42
E	5	56	7	35
F	5	28	7	35

(b)

Fig. 3. Cost estimation in raster analysis.

to the cost of an edge passing through that pixel. The results of the proposed cost estimation methods are illustrated in Fig. 3b.

The first cost estimation method, adds up all the pixel values that an edge intersects. For example, edge A intersects with three pixels whose values add up to 12. This method may produce misleading results in some scenarios. For example, although the length of edges C and E are the same and both pass through the same homogeneous region, their costs are different just because of the pixel size.

The second cost estimation method, average, takes the average pixel value that an edge intersects. This method might solve the aforementioned issue, and provide the same cost for the edges C and E. However, this time edge length is not considered. For example, edges D and F possess the same cost but they have a different length. In other words, taking the average of pixel values that an edge intersects with may also produce unreliable cost estimates.

The third method is a hybrid approach that utilises the advantages of both previous approaches. Specifically, the cost of an edge is determined by multiplying the length of an edge with the average pixel value.

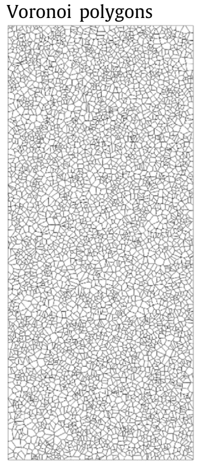
4. Illustrative examples

Two illustrative examples of the plugin were investigated to provide a better understanding of the effectiveness of the updated version. First, the performance of v1.0 and v2.0 was investigated on two polygon data sets. Second, the effectiveness of v2.0 was demonstrated on raster data.

4.1. Performance analysis

The effectiveness of the proposed v2.0 was compared with its preceding version on two different polygon layers on the same PC having a 16 GB RAM with 3.6 GHz CPU. The first layer

Table 3
Performance analysis of different versions of the MST plugin in QGIS.

		Voronoi polygons	Census blocks of New York City
			
Total area (km^2)		1 158 248	786
Number of polygons		5000	38 799
Run time (min)	v1.0	7.6	552
	v2.0	0.25	24
Length of MST (km)	v1.0	78 686	4725
	v2.0	78 682	4692

was randomly generated Voronoi polygons, and the other layer was the 2010 census blocks of New York City.[2] The comparative results are illustrated in Table 3.

The new version of the plugin was better than its preceding version on both of the compared criteria. Specifically, it determined the MST much faster (23x–30x) and the total length of the MST was shorter. The MST was generated much faster because the core library of QGIS (i.e. GDAL/OGR) was used instead of the PyQGIS Application Programming Interface (API) to perform spatial analysis such as finding the centroid of a polygon and calculating the distance between two points. The MST was also shorter in v2.0 since the two versions differ in how they generate the input graph. The proposed version utilised DT to construct the edges based on the centroids of polygons, whereas the former version assumed an edge between the centroids of adjacent polygons.

4.2. Raster analysis

The MST illustrated in Fig. 4 aims to connect the cities of Turkey while making sure that none of its edges passes through a water body (provided by land cover data obtained from CORINE[3]) or through a pixel whose height is higher than 2560 m. Such pixel values are assigned a NULL value. Finally, all the pixel values are scaled between 0 and 100 to provide a unified cost value for each pixel by weighting DEM, slope and land cover. The 'Hybrid' method described in Fig. 3 is utilised to estimate the costs of edges.

It can be seen that some centroids were isolated due to the restrictions that were imposed on the raster. Determining such isolated points are valuable as it facilitates researchers to investigate other edge generation methods in addition to DT. In this way, it would be possible to obtain fragmented edges, such as edge B in Fig. 3a, which are longer but lower in total cost.

[2] https://data.cityofnewyork.us/City-Government/2010-Census-Blocks/v2h8-6mxf.

[3] https://land.copernicus.eu/pan-european/corine-land-cover.

5. Impact

Researchers have utilised MST for various purposes in a GIS environment [21–25]. One of the common use-cases is clustering. By pruning longer edges in an MST, clusters inherent in the data can be identified [25]. For example, building clusters could be identified with this approach to do map generalisation [21]. In this way, buildings would be represented better when the scale of a map is reduced (i.e. map is zoomed out). Similarly, [22] relied on MST for regionalisation by successive removal of edges that link dissimilar regions. A recent study conducted by [23] aimed to understand building patterns in a large urban area by considering the spatial proximity and building characteristics while constructing the MST. The authors of [24] modelled road network by two components: superhighways and roads. This classification emerged when the centrality of nodes was analysed within an MST.

The proposed Geo-MST plugin in QGIS can readily be used in these various research areas. Researchers and analysts who need to identify MST on geographical data can readily install and start using the proposed plugin without any monetary cost. Consequently, the proposed plugin contributes to the advancement of open science.

6. Conclusions

From providing a dynamic reorganisation of power distribution networks to understanding how road network links are classified, MST has been used in various research domains that are inherently geospatial. The purpose of the proposed QGIS plugin is to provide a unified framework to determine and visualise MST in geospatial data. While determining MST is not new, to the best of our knowledge, the existing solutions either lack in performance or cannot handle different types of geospatial data. The updated version of the plugin offers different options to obtain the costs of edges, which correspond to the three tabs of the GUI. The first tab, 'Vector' assumes the availability of the costs and operates for line inputs. The second tab, 'Automatic' generates a graph automatically based on DT and assumes the cost of each edge is the distance between two points. Finally, the 'Raster' tab utilises both raster and vector data to estimate costs.

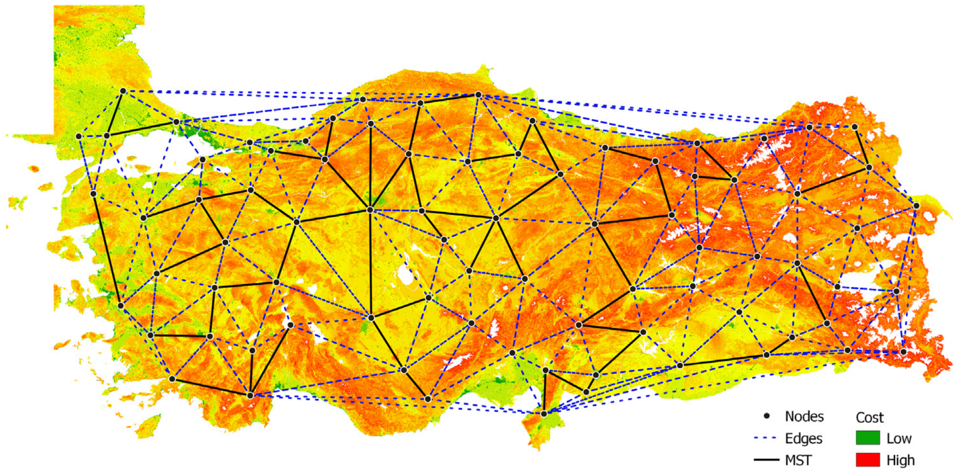


Fig. 4. Determining MST by relying on raster data.

The authors of this research will continue updating the plugin by incorporating additional functionality. Specifically, different ways to construct edges on point and polygon inputs will be investigated. In addition, the proposed plugin will be used to investigate how road links can be classified within urban road networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to thank Abdurrahman Serhan for his contributions to the v1.0 of the plugin. We also would like to thank the editor and the anonymous reviewer whose comments improved the quality of this manuscript.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.softx.2020.100553.

References

[1] Ballsun-Stanton B, Ross SA, Sobotkova A, Crook P. FAIMS Mobile: Flexible, open-source software for field research. SoftwareX 2018;7:47–52. http://dx.doi.org/10.1016/j.softx.2017.12.006.

[2] Jankowski P. Integrating geographical information systems and multiple criteria decision-making methods. Int J Geogr Inf Sci 1995;9(3):251–73. http://dx.doi.org/10.1080/02693799508902036.

[3] Lawhead J. Learning geospatial analysis with python. 2nd ed.. Birmingham: Packt Publishing; 2015.

[4] Duarte L, Espinha Marques J, Teodoro AC. An open source GIS-based application for the assessment of groundwater vulnerability to pollution. Environments 2019;6(7):86. http://dx.doi.org/10.3390/environments6070086.

[5] Jung M. LecoS — A python plugin for automated landscape ecology analysis. Ecol Inform 2016;31:18–21. http://dx.doi.org/10.1016/j.ecoinf.2015.11.006.

[6] Kuhn W. Core concepts of spatial information for transdisciplinary research. Int J Geogr Inf Sci 2012;26(12):2267–76. http://dx.doi.org/10.1080/13658816.2012.722637.

[7] Yu L, Kong D, Shao X, Yan X. A path planning and navigation control system design for driverless electric bus. IEEE Access 2018;6:53960–75. http://dx.doi.org/10.1109/ACCESS.2018.2868339.

[8] Degenne P, Lo Seen D. Ocelet: Simulating processes of landscape changes using interaction graphs. SoftwareX 2016;5:89–95. http://dx.doi.org/10.1016/j.softx.2016.05.002.

[9] Graham RL, Hell P. On the history of the minimum spanning tree problem. IEEE Ann Hist Comput 1985;7(1):43–57. http://dx.doi.org/10.1109/MAHC.1985.10011.

[10] Tarjan R. Data structures and network algorithms. Soc Ind Appl Math 1983.

[11] Mosbah M, Arif S, Mohammedi RD, Hellal A. Optimum dynamic distribution network reconfiguration using minimum spanning tree algorithm. In 2017 5th International Conference on Electrical Engineering - Boumerdes (ICEE-B): 1–6. http://dx.doi.org/10.1109/ICEE-B.2017.8192170.

[12] Dutta S, Patra D, Shankar H, Verma PA. Development of gis tool for the solution of minimum spanning tree problem using prim's algorithm. In ISPRS - International archives of the photogrammetry, remote sensing and spatial information sciences, XL-8: 1105–1114. http://dx.doi.org/10.5194/isprsarchives-XL-8-1105-2014.

[13] Anbaroğlu B, Serhan A. Geomatik Mühendisliğinde Aktif Öğrenme için bir Model Önerisi: QGIS Eklentisi Geliştirme. In: TMMOB 6. Coğrafi bilgi sistemleri Kongresi, Ankara. 2019, Accessed: May 21, 2020. [Online]. Available: https://cbskongresi.org/wp-content/uploads/2020/01/TO15-2.pdf.

[14] Probst D, Reymond JL. Visualization of very large high-dimensional data sets as minimum spanning trees. J Cheminform 2020;12(1):12. http://dx.doi.org/10.1186/s13321-020-0416-x.

[15] Nascimento M, Sousa A, Ramirez M, Francisco AP, Carriço JA, Vaz C. PHYLOViZ 2.0: providing scalable data integration and visualization for multiple phylogenetic inference methods. Bioinformatics 2017;33(1):128–9. http://dx.doi.org/10.1093/bioinformatics/btw582.

[16] Salipante SJ, Hall BG. Inadequacies of minimum spanning trees in molecular epidemiology. J Clin Microbiol 2011;49(10):3568–75. http://dx.doi.org/10.1128/JCM.00919-11.

[17] Longley PA, Goodchild MF, Maguire DJ, Rhind DW. Geographic information science and systems. 4th ed.. Wiley; 2015.

[18] Klein R. Voronoi diagrams and Delaunay Triangulations. In: Kao M-Y, editor. Encyclopedia of algorithms. New York, NY: Springer; 2016, p. 2340–4.

[19] Kruskal JB. On the shortest spanning subtree of a graph and the traveling salesman problem. Proc Amer Math Soc 1956;7(1):48–50. http://dx.doi.org/10.2307/2033241.

[20] Jolma A, et al. Open-source tools for environmental modeling. In: Kresse W, Danko DM, editors. Springer handbook of geographic information. Berlin, Heidelberg: Springer; 2012, p. 597–619.

[21] Wei Z, Guo Q, Wang L, Yan F. On the spatial distribution of buildings for map generalization. Cartogr Geogr Inf Sci 2018;45(6):539–55. http://dx.doi.org/10.1080/15230406.2018.1433068.

[22] Assunção RM, Neves MC, Câmara G, Freitas CDC. Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees. Int J Geogr Inf Sci 2006;20(7):797–811. http://dx.doi.org/10.1080/13658810600665111.

[23] Wu B, et al. An Extended Minimum Spanning Tree method for characterizing local urban patterns. Int J Geogr Inf Sci 2018;32(3):450–75. http://dx.doi.org/10.1080/13658816.2017.1384830.

[24] Wu Z, Braunstein LA, Havlin S, Stanley HE. Transport in weighted networks: Partition into superhighways and roads. Phys Rev Lett 2006;96(14):148702. http://dx.doi.org/10.1103/PhysRevLett.96.148702.

[25] Gower JC, Ross GJS. Minimum spanning trees and single Linkage Cluster analysis. J R Stat Soc Ser C Appl Stat 1969;18(1):54–64. http://dx.doi.org/10.2307/2346439.