Street Network Generation with Adjustable Complexity Using k-Means Clustering

Quentin Goss*, Mustafa İlhan Akbaş*, Luis G. Jaimes*, R. Sanchez-Arias†
*Department of Computer Science
†Department of Data Science and Business Analytics
Florida Polytechnic University, Lakeland FL
Email:{quentingoss0323,makbas,ljaimes,rsanchezarias}@floridapoly.edu

***Abstract*—The transportation system is an important part of the daily lives of a major portion of the world's population. Therefore, innovative applications are designed to improve the experience of drivers and pedestrians in this system as technological advances allow. These applications analyze street networks to come up with suggestions such as fastest route or optimal ride-sharing route. Graph theory has been used frequently for these analyses, where each junction is represented by a vertex and each road segment is represented by an edge. However, applications also require grouping of these vertices so that they can run optimization methods in a larger scale.**

In this paper, we propose a method for creating street networks with adjustable complexity. By using k-Means clustering, our mechanism allows for the increase or decrease of the gradient of a street network. The implementation results demonstrate the proposed method's efficiency and flexibility for providing street network graphs with adjustable complexity.

***Index Terms*—Street Networks, Simulation, k-means Clustering, Connected and Autonomous Vehicles, Graph Theory**

I. Introduction

The transportation system provides invaluable services to users, which traditionally focus on moving people and goods from one point to another in the shortest time. With the introduction of technological tools and applications into this system in the recent years, the transportation ecosystem started to offer interesting and innovative applications. Examples of these include not only the applications we use everyday such as navigation tools with expected trip duration, but also applications with a futuristic focus such as crowd sourcing methodologies or smart routing applications for connected and autonomous vehicles (CAV). These new applications generally have a wider scope and aim to provide service for a whole city, state or even a country. Therefore, they require high processing power, speed and their implementation is mostly based on cloud-based technologies.

The street network graphs have been important resources and tools for the innovative applications. A graph G is a simple construct of vertices V and edges E. A street network can be represented as a graph where the roads segments are the edges of G and the intersections, dead ends and connections between road segments are the vertices of G. When a street network is converted into a graph, the rich literature on graph theory and related tools become available to study this network. Hence, such a graph is a useful tool for not only intelligent transportation system research, but also for street network analysis [1] and cyber-physical system applications [2], [3].

978-1-7281-0137-8/19/$31.00

The value in defining street networks as graphs was recognized early. The use of graph theory in the study of street networks dates back to Euler's solution of the urban problem of the Königsburg bridges puzzle [4]. Graph theory is also proven to be useful for building complex street and city networks, and in the analysis of street network data [5].

Even though the street network graphs have proven to be resourceful in many fields, there are various challenges in studying them. Due to the processing limitations, some of the street network applications are oversimplified or their sample sets are very small. On the other hand, a graph of a complex street network such as the output of the street network retrieval tool OSMnx [6] contains various vertices and edges. A major portion of these may not be necessary for some of the street network applications. As the length of roads and distance between any two points in a street network forms the important data, they may be represented just as a spacial graph in which V has two positional values (x, y) and E has a length value l. Additionally, same application may require different levels of complexity on the same graph data depending on the scope of the study. Hence, the literature and applications in this domain suffer from the lack of a street network graph generation method with dynamic complexity.

In this paper, we introduce an approach to create street networks with adjustable complexity using k-means clustering. Our mechanism allows for the increase or decrease of the gradient of a street network by adjusting the number of clusters k. Consider a paper map for any geographical location with roads. A highway map showing the major routes throughout a state has a lower gradient of detail than a map of a downtown and all of the minor, interconnected streets. Much like a paper map, the gradient of detail in a street network may also need be reduced depending on the requirements of an application. This will improve the performance during processes with high requirements such as traffic simulations.

The remainder of this paper is organized as follows. Related work is given in Section II. Our graph generation approach and a detailed description of the processes are given in Section III.

We present the graph complexity adjustment with implementation examples in Section IV and conclude in Section V.

II. Related Work

Since the Euler's work on Königsburg seven bridges [4], the graph theory and eventually network science have been important tools for network analysis. Therefore, there are various approaches using graph theory, network science and street network to graph conversion specifically for transportation systems. The simplification of the graphs is also important in this field because of the scalability and applicability requirements.

As an urban planning example, Masucci et al. [5] uses graph theory for an analysis of London street network. The analysis compares London street network to idealized graph models and the results lay out the characteristics of the street network. An interesting outcome of the study shows that the physical and mental effort are needed equally to navigate within the city. Walkability is another critical property for urban planning. There are even walking index parameters created for this purpose [7]. Foti [8] proposes a scalable methodology that takes into account the destinations, demographics, historical travel behavior to create variables that can be used to define walking, automobile, and transit selection in an urban model.

Urban Network Analysis (UNA) toolbox for ArcGIS [1] is also used in urban planning and computes network centrality measures on street networks such as gravity index, betweennness or closeness. For urban planning applications, UNA includes buildings in addition to the vertices and edges in its calculations. Frizzelle et al. [9] also make use of geographic information systems (GIS) for street network analysis. Their approach focuses on the quality, accuracy and scalability of existing road data since the main purpose of the approach is to decide on the usability of road datasets for health research and applications.

The OSMnx [6] is one of the most comprehensive open source tools to download and analyze street networks. OSMnx acquires the map data from OpenStreetMap, constructs graphs using the data and provides simplification and visualization options. It also has built-in network measurement features, which is common in street network analysis. TrajGraph [10] is another visual analysis tool that uses graph theory for urban planning analysis. TrajGraph focuses on mobility patterns and uses real taxi trajectory data for its measurements such as Pagerank and betweenness. The results of the analyses are used to suggest a level for the importance of city streets.

The idea of grouping or clustering of road segments or junctions has been an important part of some transportation system or urban planning approaches. The framework by Kiliç and Gök [11] is a web based public transportation route planning tool, which groups addresses within the geographical region around a university. The tool determines the most efficient route from the starting point, an address within one of the clusters, to its destination using these grouping of addresses.

There are also several approaches using k-means clustering in transportation system. Pattanaik et al. [12] propose a framework, which identifies groupings of traffic on road maps using k-means clustering to group nearby vehicles, which they visualize with convex hulls.

The existing street network to graph conversion approaches serve the purpose of various specific applications. The method proposed in this paper can be used to eliminate the scalability limitations for most of these approaches by generating networks with varying complexity.

III. Street network to graph conversion

The transportation system simulators are used frequently for street network analysis and generation of what-if scenarios. SUMO (**S**imulation of **U**rban **MO**bility) is chosen as the transportation system simulation platform for our approach. SUMO is an open source traffic simulation software package which provides tools for creating, editing, and running 2D (two-dimensional) traffic simulations, extensive traffic modeling and pattern creation [13]. In SUMO software package is a Python interface known as TraCI (**Tra**ffic **C**ontrol **I**nterface) which provides micromanagement of the street network simulation at an atomic level [14]. A sample of a simple street network simulation produced by using SUMO and TraCI is shown in Figure 1, which features a grid-style layout of roads, triangular vehicles of multiple colors (types), intersections, and circular points of interest. SUMO street network data is stored in the data format XML (**Ex**tensible **M**arkup **L**anguage) [15].

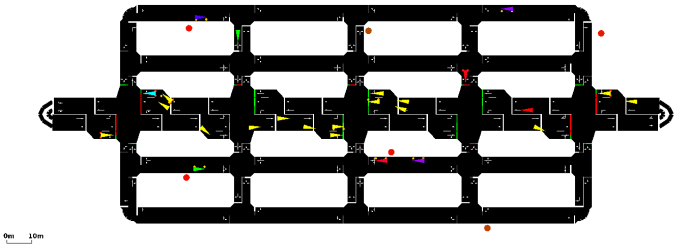


Fig. 1: A grid style road network with simulated traffic.

Complimenting SUMO, we used OpenStreetMap (OSM) as a source of geographical map and road data provided by OSM contributors [16]. The OSM data is available under the Open Data Commons Open Database Licence and SUMO allows utilization of this data as a source of realistic street network data.

In addition to SUMO and OSM, we developed modules in R and Python in this study. Python's version 3.7.1 is utilized for the conversion of SUMO street network data into a data structure that R may work with to create a directed graph. R is a free software environment for statistical computation and graphics [17]. The following R packages are utilized during the process of street network simplification; R's `base` package is used for data wrangling, plotting points to a graph, and for an implementation of the k-means algorithm method. R's `jsonlite` package [18] is used for reading and writing data in JSON (JavaScript Object Notation) format [19]. R's `statnet`[20] and `ergm` [21] packages are used to visualize directed graphs.

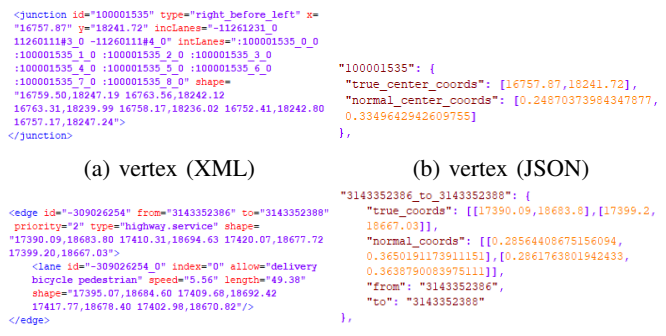
A. Processing the street network data

The street network data is filtered to extract the vertex and edge information. The data points that are filtered include identifiers (ID) of the vertices and edges as well as the two-dimensional (2D) positions of each vertex. Table I lists the filtered and processed format parameters in the street network data. The processed parameters are converted into an R readable format for further analysis. The `tru_center_coords` and `true_coords` parameters are spacial `x` and `y` position values. The `normal_center_coords` and `normal_coords` parameters are normalized spacial `x` and `y` position values between 0 and 1. The `from` and `to` parameters are the IDs of vertices belonging to the start and end point of an edge.

TABLE I: R JSON Data vs SUMO XML Data

R JSON Format	SUMO XML Format
Vertex data	
id	id
true_center_coords	$[x, y]$
normal_center_coords	$[\frac{\max(x^2)-x^2}{\max(x^2)}, \frac{\max(y^2)-y^2}{\max(y^2)}]$
Edge Data	
id	from_to
true_coords	$[[x_{to}, y_{to}], [x_{from}, y_{from}]]$
normal_coords	$[[\frac{\max(x_{to}^2)-x_{to}^2}{\max(x_{to}^2)}, \frac{\max(y_{to}^2)-y_{to}^2}{\max(y_{to}^2)}],$ $[\frac{\max(x_{from}^2)-x_{from}^2}{\max(x_{from}^2)}, \frac{\max(y_{from}^2)-y_{from}^2}{\max(y_{from}^2)}]]$
from	from
to	to

Figure 2 depicts the street network data before and after it is processed into the format we use for our implementation. Figures 2a and 2b depict the vertex data of the street network in XML and JSON format respectively and Figures 2c and 2d depict the edge data of the street network in XML and JSON format respectively.



(a) vertex (XML) (b) vertex (JSON)

(c) Edge (XML) (d) Edge (JSON)

Fig. 2: A comparison of SUMO XML formatted data versus filtered data in JSON format.

After we filter the street network data as described in Table I and shown as an example in Figure 2, the vertex data and the edge data are ready to be processed further. The filtered data is combined into a three-dimensional (3D) matrix for both vertex and edge data, the structures of which are given in Tables II and III respectively.

TABLE II: Vertex data matrix in R.

Vertex Data		
	true_center_coords	normal_center_coords
id_0	$true\_center\_coords_0$	$normal\_center\_coords_0$
id_1	$true\_center\_coords_1$	$normal\_center\_coords_1$
...	...	...
id_n	$true\_center\_coords_n$	$normal\_center\_coords_n$

TABLE III: Edge data matrix in R.

Edge Data				
	true_coords	normal_coords	from	to
id_0	$true\_coords_0$	$normal\_coords_0$	$from_0$	to_0
id_1	$true\_coords_1$	$normal\_coords_1$	$from_1$	to_1
...	...	...	...	...
id_n	$true\_coords_n$	$normal\_coords_n$	$from_n$	to_n

B. Creating an Adjacency Matrix

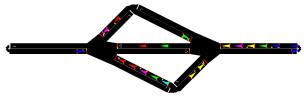
The vertex and edge data is ready to be translated into a directed graph following Algorithm 1. Algorithm 1 takes vertex and edge data as input and returns an adjacency matrix which we use to represent a directed graph. To create an adjacency matrix for a weighted directed graph, simply line 14 of Algorithm 1 can be changed to: A `[indexFrom,indexTo]` $\leftarrow$ A `[indexFrom,indexTo]` + 1.

Algorithm 1 Creating and adjacency matrix from vertex and edge data.

1: **procedure** CREATEADJACENCYMATRIX(Vertices data V and Edges data E)
2: $n_V \leftarrow$ length (V)
3: `vertIDS` $\leftarrow$ an array of all ids of each vertex in V
4:
5: */* Initialize the adjacency matrix */*
6: $A \leftarrow$ a 2D array of 0s of size n_V by n_V
7:
8: */* Fill the empty adjacency matrix */*
9: **for** each edge e in E **do**
10: `IDfrom` $\leftarrow$ start vertex of e
11: `IDto` $\leftarrow$ end vertex of e
12: `indexFrom` $\leftarrow$ index `(IDfrom,vertIDs)`
13: `indexTo` $\leftarrow$ index `(IDto,vertIDs)`
14: A `[indexFrom][indexTo]` $\leftarrow 1$
15: **return** A

As an example, we describe the process of creating a directed graph from vertex and edge data using R. To create a directed graph in R, an adjacency matrix of the street network is created and supplied into the `statnet` library method `gplot`. The process of creating a directed graph from a SUMO street network is presented in Figure 3.

The SUMO street network (Figure 3a) is processed into an edge matrix (Figure 3b), in which one row of the edge matrix is a unidirectional edge between the junction id stored in the `from` attribute to the junction id stored in the `to` attribute. With the `from` and `to` attributes of the edge matrix, an adjacency matrix is constructed (Figure 3c). Finally the adjacency matrix is input is used with the `gplot` method in the R `statnet` library to visualize the directed graph (Figure 3d).



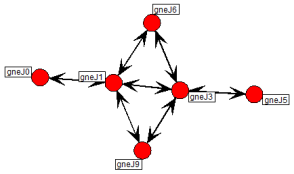
(a) SUMO street network

	true_coords	normal_coords	from	to
gneJ1_to_gneJ0	Numeric,4	Numeric,4	"gneJ1"	"gneJ0"
gneJ3_to_gneJ1	Numeric,4	Numeric,4	"gneJ3"	"gneJ1"
gneJ5_to_gneJ3	Numeric,4	Numeric,4	"gneJ5"	"gneJ3"
gneJ6_to_gneJ1	Numeric,4	Numeric,4	"gneJ6"	"gneJ1"
gneJ3_to_gneJ6	Numeric,4	Numeric,4	"gneJ3"	"gneJ6"
gneJ9_to_gneJ1	Numeric,4	Numeric,4	"gneJ9"	"gneJ1"
gneJ3_to_gneJ9	Numeric,4	Numeric,4	"gneJ3"	"gneJ9"
gneJ0_to_gneJ1	Numeric,4	Numeric,4	"gneJ0"	"gneJ1"
gneJ1_to_gneJ3	Numeric,4	Numeric,4	"gneJ1"	"gneJ3"
gneJ3_to_gneJ5	Numeric,4	Numeric,4	"gneJ3"	"gneJ5"
gneJ1_to_gneJ6	Numeric,4	Numeric,4	"gneJ1"	"gneJ6"
gneJ6_to_gneJ3	Numeric,4	Numeric,4	"gneJ6"	"gneJ3"
gneJ1_to_gneJ9	Numeric,4	Numeric,4	"gneJ1"	"gneJ9"
gneJ9_to_gneJ3	Numeric,4	Numeric,4	"gneJ9"	"gneJ3"

(b) Edge Matrix

	gneJ0	gneJ1	gneJ3	gneJ5	gneJ6	gneJ9
gneJ0	0	1	0	0	0	0
gneJ1	1	0	1	0	1	1
gneJ3	0	1	0	1	1	1
gneJ5	0	0	1	0	0	0
gneJ6	0	1	1	0	0	0
gneJ9	0	1	1	0	0	0

(c) Adjacency Matrix



(d) Directed Graph

Fig. 3: The process of creating a directed graph from a SUMO street network.

C. Directed Graphs

Using Algorithm 1, we create seven adjacency matrices which are processed into the directed graphs shown in Figure 4. Figure 4d is derived from a map of Florida Polytechnic University in Lakeland, Florida and Figures 4f and 4g are more complex street networks of one city and two city networks respectively. Figures 4a, 4b, 4c, and 4e are custom made street networks.

The corresponding statistics for the directed graphs are shown in Table IV. Three of the directed graphs are realistic in size and shape of a potential street network. A graph marked as *realistic* is derived from real-world street network data.

TABLE IV: Directed Graph Statistics

Graph	Vertices	Edges	Realistic?
Fig. (4a)	6	7	
Fig. (4b)	7	12	
Fig. (4c)	37	104	
Fig. (4d)	278	498	Yes
Fig. (4e)	412	1186	
Fig. (4f)	2656	6872	Yes
Fig. (4g)	2973	7037	Yes

IV. Adjusting Graph Complexity

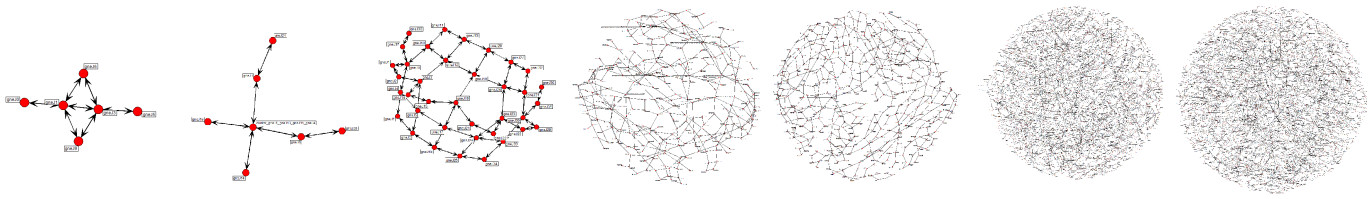
In this section, we will discuss the methods of processing the vertexes of our street networks by performing k-means clustering on the `x` and `y` position values and using the results of the k-means clustering to rebuild a street network graph which is of less complexity. The k-means clustering is chosen as it produces as output a centroid with `x` and `y` position values that can then be plotted as a any other vertex in the street network may. As such, any clustering algorithm in which the output is a centroid with `x` and `y` position values may be substituted for the k-means steps in this section.

1) Assigning Each Vertex to a Cluster: In order to produce a centroid as output from the k-means clustering algorithm, it is necessary to assign vertices to a cluster and provide `x` and `y` position values of which will be used when determining the centroid's `x` and `y` position values. In Algorithm 2 we perform k-means clustering on the vertices data V based on Euclidean distance of the `x` and `y` position values of each vertex in V. The amount of clusters k must be given to the algorithm and should be within $1 \geq k \leq n_V$ where n_V is the amount of vertices in V. For k-means clustering, we use the `kmeans` procedure from R and the Hartigan-Wong algorithm.As will be shown in the results section of the paper, the simplicity of a street network is increased as the value of k is increased. We initialize an empty list C of length n_V, which contains an integer for each vertex in V. The list will correspond to the cluster which the vertex is placed in by the k-means clustering algorithm. For initialization, the cluster is placed into cluster -1. The vertices data V, clustering algorithm F and amount of clusters k are given as inputs into a k-means clustering method. Within this method, the vertices of V are grouped into numbered clusters.

Algorithm 2 K-Means clustering on vertex data.

1: **procedure** GroupAndFindClustersK-Means(Vertices data V, Given number of clusters k, Clustering algorithm F)
2: $n_V \leftarrow$ length (V)
3: `xyCoords` $\leftarrow$ a 2d array of the (x,y) coordinates of each vertex in V. Size n_V by 2.
4:
5: */* Initialize empty clusters index array */*
6: $C \leftarrow$ an empty array of -1's of size n_V
7:
8: */* Perform k-means on the (x,y) positions of V */*
9: $C \leftarrow$ k-means `(xyCoords,`F`,`k`)`
10: **return** C

2) Finding Centroids:: The centroids of the clustered vertices represent all of the vertices within that cluster as a single vertex. Unless k is equal to the amount of original vertices, the centroids `x` and `y` position values will be nearby all of the vertices that share the same cluster. In Algorithm 3 we determine the (x, y) coordinates for the centroid of each cluster by cross-referencing our vertex data and the cluster index data from the output of Algorithm 2. The centroid (x, y) coordinate values are the arithmetic means of all the (x, y) coordinate values of vertices within the cluster that the centroid belongs to. The centroids are used as the vertices of the simplified street network.



(a) (b) (c) (d) (e) (f) (g)

Fig. 4: Seven directed graphs for the street network data, where complexity increases from (a) to (g).

Algorithm 3 Find centroids given vertex cluster indexes and vertex data.

```
1:  procedure FINDCENTROIDS(Vertices data V, Vertex cluster indexes C)
2:      n ← max (C)
3:      /* Initialize centroids array. */
4:      R ← a 2d array of 0.0's of size n by 2
5:      Rsizes ← an array of 0's of size n
6:
7:      /* Find the arithmetic mean for (x,y) in V */
8:      for each vertex v in V do
9:          x ← x coordinate of v
10:         y ← y coordinate of v
11:         i_V ← index (v, V)
12:         i_C ← C [i_V]
13:         newSumX ← R [i_C][0] + x
14:         newSumY ← R [i_C][1] + y
15:         R [i_C][0] ← newSumX
16:         R [i_C][1] ← newSumY
17:         newSize ← Rsizes [i_C] + 1
18:         Rsizes [i_C] ← newSize
19:     for each centroid r in R do
20:         i_R ← index (r, R)
21:         r̄ ← R[i_R] / Rsizes[i_R]
22:         R[i_R] ← r̄
23:     return R
```

Algorithm 4 Determine edges between clusters.

```
1:  procedure FINDNEIGHBORS(Vertex data V, Edge data E, Centroids R, Clusters C)
2:      n_R ← length (R)
3:      vertIDs ← an array of all ids of each vertex in V
4:
5:      /* Initialize the Adjacency Matrix */
6:      A ← An 2d matrix of 0's of size n_R by n_R
7:
8:      /* Fill the Adjacency Matrix */
9:      for each edge e in E do
10:         IDfrom ← start vertex of e
11:         indexFrom ← index (IDfrom,vertIDs)
12:         c_0 ← C[indexFrom]
13:         IDto ← end vertex of e
14:         indexTo ← index (IDto,vertIDs)
15:         c_1 ← C[indexFrom]
16:         newWeight ← A[c_0][c_1] + 1
17:         A[c_0][c_1] ← newWeight
18:     return A
```

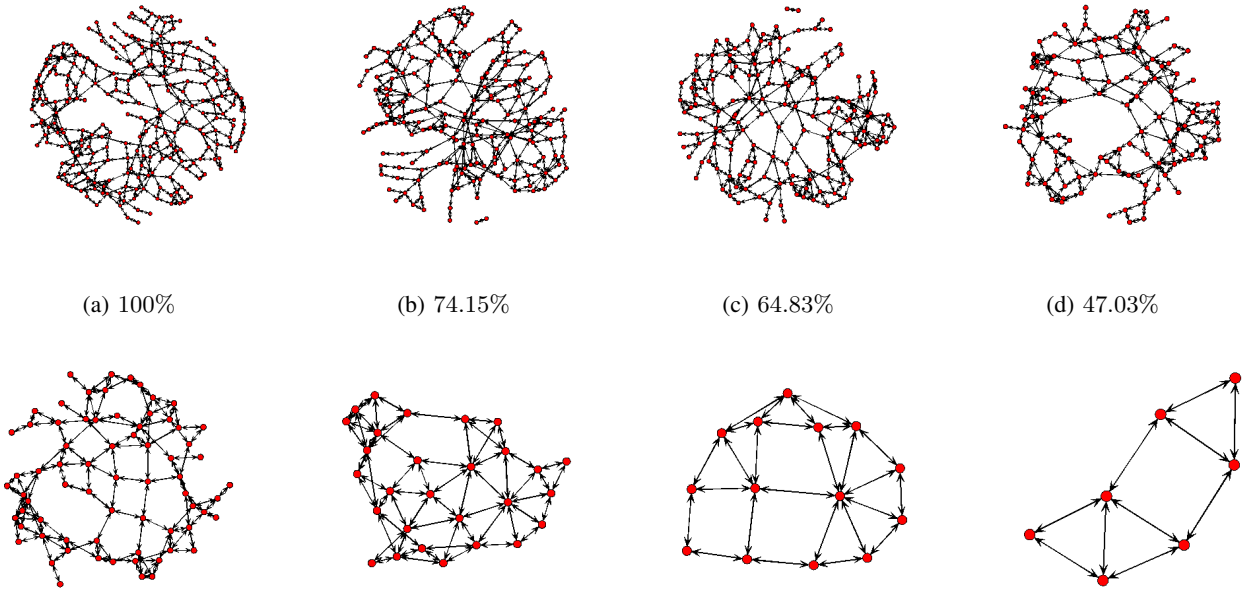
3) Connecting Clusters: Just as roads connect the junctions of a street network, the vertices of the simplified street network must also retain their connections to one another. In Algorithm 4, we construct a weighted adjacency matrix for the k-means clustered data which will be used to create the edges of the simplified street network. This adjacency matrix is of size n^k by n^k where k is the total amount of centroids or otherwise the k of the k-means clustering algorithm. If an edge in the unprocessed edge data has a starting vertex and an ending vertex that belongs to two different clusters, then we say that there is a link between those two clusters. For example, given that A and B are two clusters and e_0 and e_1 are vertices that belong to clusters A and B respectively, we say that there exists a one-directional edge from A to B.

Gradient of Complexity Reduction: Figure 5 depicts the results of varied values of k of a graph input into Algorithm 2, then 3 and 4. Shown are the gradients of complexity of the directed graph from Figure 4d which is the street network of Florida Polytechnic University, in Lakeland Florida. Figure 5a depicts the simplified directed graph in which k is equal to the total amount of vertices and 5h is the same street network where k is 2.96% of the total amount of vertices.

V. Conclusion

Street network analysis is critical in various fields, from urban planning to ride-sharing applications. The conversion of street networks to graphs has been an important part for these analyses. However, there are complexity and scalability issues in the existing approaches. To address these issues, we propose a street network graph generation method with adjustable complexity. By using k-Means clustering, our mechanism allows adjustment of the gradient of street networks.

We plan to implement this approach on a large transportation system to dynamically simplify a portion of the street network using our techniques. We are going to utilize many occurrences of k-means clustering to street network for adjusting the gradient of simplification as necessary in a way that varies across the street network. Another direction of future research could be on the development of methods for determining the optimal value of k for a given application.



(a) 100% (b) 74.15% (c) 64.83% (d) 47.03%

(e) 29.66% (f) 11.86% (g) 5.93% (h) 2.97%

Fig. 5: The directed graphs of a street network in incremental levels of vertex reduction.

VI. Acknowledgments

The street network data is partially derived from OSM data ©OSM contributors and available at www.openstreetmap.org/.

This material is based upon work primarily supported by the National Science Foundation (NSF) under NSF Award Number 1739409. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the author(s), and do not necessarily reflect those of the NSF.

References

[1] A. Sevtsuk and M. Mekonnen, "Urban network analysis," *Revue internationale de géomatique–n*, vol. 287, p. 305, 2012.

[2] G. Solmaz, M. İ. Akbaş, and D. Turgut, "A Mobility Model of Theme Park Visitors," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2406–2418, 2015.

[3] M. R. Brust, M. I. Akbaş, and D. Turgut, "VBCA: A Virtual Forces Clustering Algorithm for Autonomous Aerial Drone Systems," in *Annual IEEE Systems Conference (SysCon)*, 2016, pp. 1–6.

[4] L. Euler, "Solution problematis ad geometrian situs pertinentis," *Comm. Acad. Sci. Imp. Petropol*, vol. 8, p. 128, 1736.

[5] A. P. Masucci, D. Smith, A. Crooks, and M. Batty, "Random planar graphs and the london street network," *The European Physical Journal B*, vol. 71, no. 2, pp. 259–271, 2009.

[6] G. Boeing, "OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks," *Computers, Environment and Urban Systems*, vol. 65, pp. 126–139, 2017.

[7] K. Manaugh and A. El-Geneidy, "Validating walkability indices: How do different households respond to the walkability of their neighborhood?" *Transportation research part D: transport and environment*, vol. 16, no. 4, pp. 309–315, 2011.

[8] F. Foti, "A behavioral framework for measuring walkability and its impact on home values and residential location choices," Ph.D. dissertation, UC Berkeley, 2014.

[9] B. G. Frizzelle, K. R. Evenson, D. A. Rodriguez, and B. A. Laraia, "The importance of accurate road data for spatial applications in public health: customizing a road network," *International journal of health geographics*, vol. 8, no. 1, p. 24, 2009.

[10] X. Huang, Y. Zhao, C. Ma, J. Yang, X. Ye, and C. Zhang, "TrajGraph: A graph-based visual analytics approach to studying urban network centralities using taxi trajectory data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 160–169, 2016.

[11] F. Kiliç and M. Gök, "Totrop: A web-based public transportation routing tool," in *Proceedings of 7th International Symposium on Intelligent and Manufacturing Systems*, vol. IMS 2010. IMS, 09 2010, pp. 500–509.

[12] V. Pattanaik, M. Singh, P. Gupta, and S. Singh, "Smart real-time traffic congestion estimation and clustering technique for urban vehicular roads," in *IEEE Region 10 Conference (TENCON)*, Nov 2016.

[13] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO - Simulation of Urban MObility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, December 2012.

[14] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "Traci: An interface for coupling road traffic and network simulators," in *Proceedings of the Communications and Networking Simulation Symposium*. ACM, 2008, pp. 155–163.

[15] T. Bray, J. Paoli, and C. M. Sperberg-McQueen, "Extensible markup language," *World Wide Web J.*, vol. 2, no. 4, pp. 29–66, Nov. 1997.

[16] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[17] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2018.

[18] J. Ooms, "The jsonlite package: A practical and consistent mapping between json data and r objects," *arXiv:1403.2805*, 2014.

[19] Ecma International, "The json data interchange format," Standard ECMA-404, October 2013.

[20] M. S. Handcock, D. R. Hunter, C. T. Butts, S. M. Goodreau, P. N. Krivitsky, and M. Morris, *ergm: Fit, Simulate and Diagnose Exponential-Family Models for Networks*, The Statnet Project, 2018, R v 3.9.4.

[21] D. R. Hunter, M. S. Handcock, C. T. Butts, S. M. Goodreau, and M. Morris, "ergm: A package to fit, simulate and diagnose exponential-family models for networks," *Journal of Statistical Software*, vol. 24, no. 3, pp. 1–29, 2008.