Short Term Scholar Weekly Progress Report 6

Ufuk Bombar

August 18, 2019

The goal of this week is to learn more on graphs and other possible applications and start implementation.

On Monday, I started to read an article, covering the methods of optimizations for graph traversal algorithms[1]. The article divides the graph traversal into two sub-problems. First is the *static routing* and secondly, *time dependent routing.* The *static routing* covers all the optimizations for graphs that have static edges. For example, a local internet network can be represented as a static network. Because the weight of the edges does not depend on time. On the other hand, a road network mostly classified as a time-dependent network since the weight of the edges depends on the traffic and as a result time. I acknowledged that there are many methods for pre-processing. For example, in static routing, the graph is optimized by running a method before any search queue. This method groups the nodes into chunks. So that search queues can be processed starting from high level to low level. This method in general named *node contraction.* Figure 2 shows this method visually. However, a bigger problem rests on the *time dependent routing.* Since the weight of the edges constantly changing the optimizations are harder to achieve and they are dependent to the shape of the graph. After covering the paper, I decided to make a traffic simulator application. I cogitated on basic features and implementation requirements. Rest of my time I work on setting up the environment for development.

On Tuesday, I work on the format of *analysis report.* Most of my time spent on learning to format on latex. After I finish the formatting, I started to add functional and non-functional requirements to the analysis report. In the middle of the report, I asked for feedback to Mr. Altınakar. We decided to move forward with another application that requires more interaction with raster files. He also gave me some articles to read. They were about extracting information from elevation data[2, 3, 4] and calculation of road lines for minimal earthwork[5]. I mostly interested in earthwork. I imagined a program that finds the road line for minimum earthwork. But before that, I started to do background research about *watersheds.*

On Wednesday, I read the last section of chapter 11 and finally finished it. By that, I learned possible computations I can do on raster files. I decided to write a small program for creating an elevation map from my vector file. First, I created an empty raster file, then I created a vector file that contains polygons of different elevation attributes. After that, I projected the vector to a raster file. But to make a plot the resolution was too high. Therefore, I re-sampled it to a lower resolution by using bilinear filtering. This method allows to re-sample an image while smoothing the sharp edges[6]. Then, I achieved an artificial elevation map made by a vector file. Lastly, I used *matplotlib* to create a 3d graph of the elevation map. The 3d plot shown in figure 1.

On Tuesday, I continued my readings. I learned *steam order. Steam order* is a numbering system of the steam branches that are used in different scenarios and algorithms. However, the numbering of branches varies. A most common one, *Strahler steam order* only allows the rank of the branch to be incremented if the sub-branches have the same ranks. Figure 3a shows the numbering process visually. Another method is the *Shreve steam order* shown in figure 3b which calculates the rank of a branch by adding other connected branches. It is important to note that the rank does not necessarily mean size. These ranks can be used for different purposes.

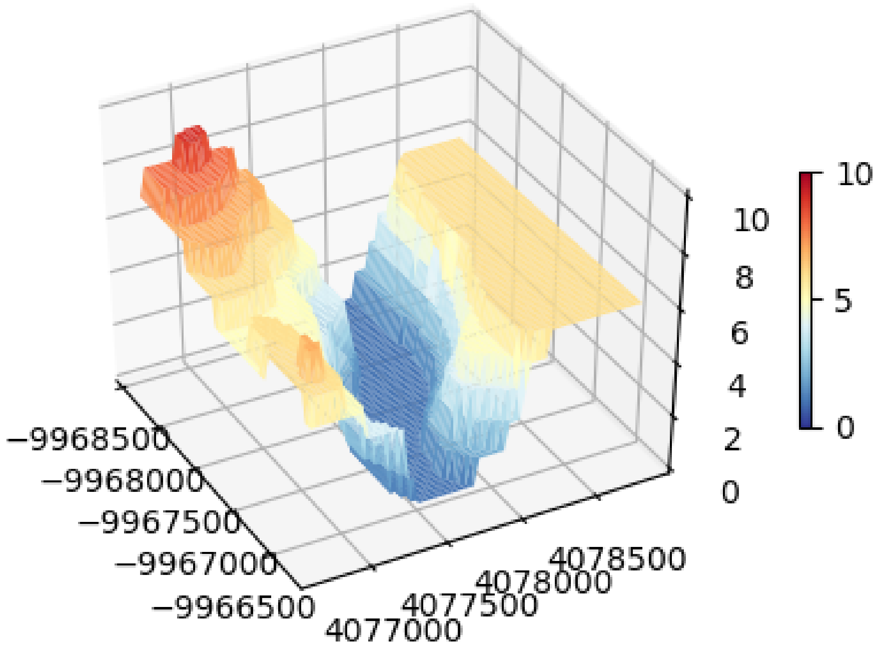


Figure 1: Plot of elevation map.

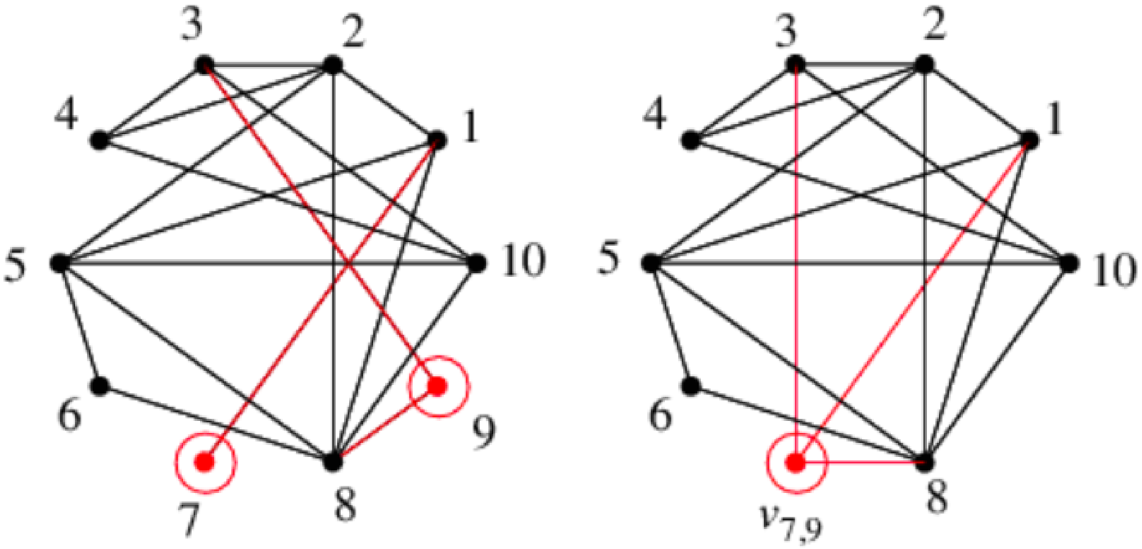
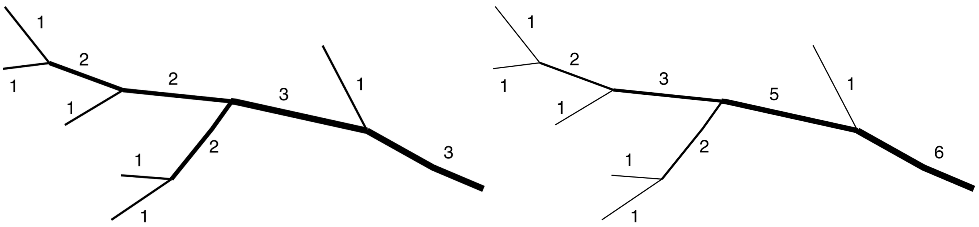


Figure 2: Node contraction [7].



(a) Strahler steam order.

(b) Shreve steam order.

Figure 3: Comparison of Shrahler and Shreve steam order.

References

[1] Daniel Delling et al. “Engineering route planning algorithms”. In: *Algorithmics of large and complex networks*. Springer, 2009, pp. 117–139.

[2] Susan K Jenson and Julia O Domingue. “Extracting topographic structure from digital elevation data for geographic information system analysis”. In: *Photogrammetric engineering and remote sensing* 54.11 (1988), pp. 1593–1600.

[3] Anton Kornilov and Ilia Safonov. “An overview of watershed algorithm implementations in open source libraries”. In: *Journal of Imaging* 4.10 (2018), p. 123.

[4] Chaojun Liang and D Scott MaCkay. “A general model of watershed extraction and representation using globally optimal flow paths and up-slope contributing areas”. In: *International Journal of Geographical Information Science* 14.4 (2000), pp. 337–358.

[5] *NDDOT Construction Conference 2011East Conference*. NDDOT. 2011.

[6] Travis E. Oliphant. *Guide to NumPy: 2nd Edition*. Continium Press, 2015. ISBN: 978-1517300074.

[7] Eric Weisstein. *Vertex Contraction*. 2019. URL: `http://mathworld.wolfram.com/VertexContraction.html`. (accessed: 18.8.2019).