DETERMINING CRITICAL LOCATIONS IN A ROAD NETWORK

CASEY CROSON, LOUIS JOSLYN, AND SARA REED

SIMPSON COLLEGE
INDIANOLA, IA

ABSTRACT. Critical locations in infrastructure are roads that, if damaged, would cause a large disruption in the ability of vehicles to navigate a city. In this paper, we consider critical locations in the road network of Indianola, Iowa. The presence of cut vertices and values of betweenness for a given road segment are used in determining the importance of that given road segment. We present a model that uses these critical factors to order the importance of separate road segments. Finally, we explore models that focus on betweenness to improve accuracy when discovering critical locations.

1. Introduction

Critical locations in infrastructure are roads that, if damaged, would cause a large disruption in the ability of vehicles to navigate a city. Identifying critical locations in a network can be instrumental in protecting a city during acts of terrorism or natural disasters and providing aid if these disasters occur. Our research consists of discovering critical locations in the road network of Indianola, Iowa. While we are concentrating on networks in infrastructure, our findings can be applied to online, social and even complex brain networks.

Indianola is a town in central Iowa with a population of 14,782. [2] The town includes a private college and holds the county seat. Two state highways intersect within the city. There are approximately 411 intersections and 631 separate road segments within the town. [6]

We included several assumptions in the mapping of Indianola and creation of our model. First, we will use the map created by Record Herald & Indianola Tribune as the official map of Indianola. [6] Consequently, we decided to ignore minor roads that are not named on this map. We also assumed all streets to be two way streets because in an emergency situation, all one-way streets can become two way streets. We considered culdesacs to be dead ends and ignored all dead ends because they experience no through traffic. Finally, roads leaving the town were considered dead-ends following their last intersection with a road in Indianola's city limits.

In Section 2, we discuss basic graph theory terms including adjacency matrices and line graphs. In Section 3, we discuss critical factors including cut vertices and betweenness. Graph theory concepts and betweenness are implemented through the algorithms discussed in Section 4. In Section 5, the model presented by Demšar, Špatenková and Virrantaus [7] was implemented to determine critical locations. Modified models are presented in Section 6 and Section 7. Statistical analysis and comparison of each model is presented in Section 8. Specifically, we have developed a Standardized Betweenness Model that includes a novel solution to the weakness of uneven road segments in previous models.

2. Basic Graph Theory

2.1. **Definitions.** Graph theory is the study of graphs and the relationships graphs represent. A graph consists of vertices and edges. Vertices represent objects and edges represent the connection between those objects. Two vertices are adjacent if they are connected by an edge. Two edges are adjacent if they share a common vertex. Graphs can be directed or undirected. If edges do not specify a direction, normally through the use of arrows, then the graph is undirected. We will be considering undirected graphs that model the streets of Indianola, Iowa. The vertices of the graph will represent the intersections and the edges will represent the road segments between those intersections.

Within a graph, there are walks, trails and paths. A walk is defined as an alternate sequence of vertices and edges, a trail is a walk in which no edge is repeated, and a path is a trail in which no vertex is visited more than once. The length of a walk, trail, or path is determined by the number of edges it contains. Two vertices are connected if there exists a path from the first vertex to the second. Moreover, a graph is connected only if there exists a path between every two vertices in the graph. A connected graph that is undirected can be seen in Figure 1.

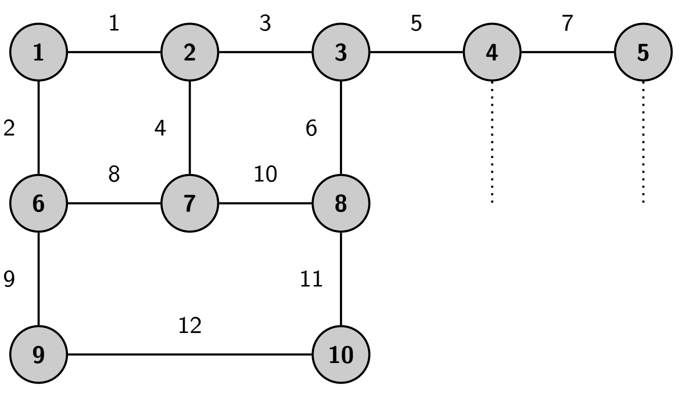

Figure 1. The graphical representation of a portion of Indianola, IA. Note the dotted lines extending from vertices 4 and 5 represent dead ends.

2.2. **Adjacency Matrix.** An adjacency matrix is the representation of a graph in a matrix. The adjacency matrix for the graph in Figure 1 is

$$
\begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{bmatrix}. \tag{1}
$$

Each numerical value in the matrix represents the number of connections from its corresponding row and column. For example, in the first row, second column there is a value of 1. This value represents the edge between the vertices 1 and 2 in Figure 1. The edge is also represented with the 1 located in the first column, second row. Note that the adjacency matrix is symmetric because the graph is undirected.

2.3. **Line Graphs.** A line graph is constructed when the edges of a graph become its vertices. This process is applied when the properties of edges need to be translated into properties of vertices. When creating the line graph of Figure 1, we know there will be 12 vertices as there are 12 edges on the graph. The connections on the line graph define the edges that the given edge intersects with. For example, Edge 1 intersects with the edges 2, 3 and 4. Therefore, on the line graph, Vertex 1 will connect with the vertices 2, 3 and 4. This method is completed for every edge in the original graph when creating the line graph. The corresponding line graph to the graph in Figure 1 can be seen in Figure 2. Now, the vertices of the line graph represent the road segments and the edges represent adjacent road segments.

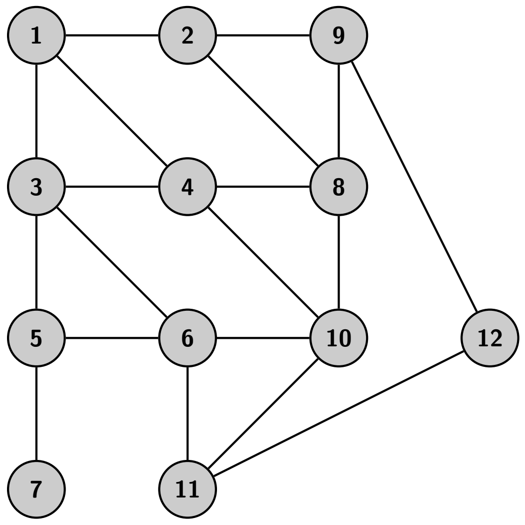

FIGURE 2. The line graph corresponding to the graph in Figure 1.

3. Critical Factors

3.1. **Cut Vertices.** A cut vertex occurs when the removal of a given vertex and its incident edges increase the number of connected components of a graph. A bridge occurs when the removal of an edge increases the number of connected components in a graph. For example, in Figure 1, Vertex 3 is a cut vertex as its removal would increase the number of connected components from 1 to 2. Furthermore, the edge labeled 5 is a bridge. Bridges of a graph correspond to cut vertices in its line graph. Thus, the vertex labeled 5 in Figure 2 is a cut vertex because the edge labeled 5 in its original graph in Figure 1 is a bridge.

3.2. **Betweenness.** Betweenness is a centrality measure which quantifies the importance of a vertex relative to other vertices in a graph. Vertices with a higher betweenness than others have more importance in the graph. Freeman [3] originally introduced betweenness and according to Demšar, Špatenková and Virrantaus [7], the formula for betweenness is

$$b(v) = \sum_{\substack{s \neq t \neq v \\ s<t}} \frac{\sigma_v(s,t)}{\sigma(s,t)} \tag{2}$$

where s and t are two distinct vertices not equal to vertex v, $\sigma_v(s,t)$ is the number of shortest paths from s to t that pass through v and $\sigma(s,t)$ is the total number of shortest paths from s to t. For our research, betweenness calculates the ratio of the number of shortest paths from point A to point B using a given road segment to the total number of shortest paths from point A to point B, where points A and B could be located at any intersection on the map.

The values of betweenness for the vertices in Figure 2 can be seen in Figure 3. Vertex 6 contains the highest betweenness value with a sum of 11. Whereas Vertex 7 has the lowest betweenness value with a sum of 0. Vertex 6 is considered more critical as it is necessary in the shortest paths between vertices. Vertex 7 is considered less critical because it is not included in the shortest path between any two vertices. Note that the values of betweenness are relative to the individual graph. In this example, the highest value of betweenness is 11. However, in Indianola which includes 630 road segments, the highest value of betweenness is about 27815.

4. Algorithms

4.1. **Matrices.** While developing our algorithm, we created an adjacency matrix to represent the graph in Figure 1. This adjacency matrix can be seen in Equation 1. Now, an adjacency matrix can be transformed to represent the line graph of the original graph. This transformation is instrumental in labeling the edges of the origianl graph, a process which allows us to identify each road segment in Indianola.

The process began by recognizing a pattern in the original graph adjacency matrix. Since the adjacency matrix for our example is symmetric along the main diagonal, we used only the top right triangle when determining edge labels. From left to right and then top to bottom, we labeled the edges in numerical order. Due to the symmetry of the matrix, we reflected these values over the main diagonal. The resulting matrix is

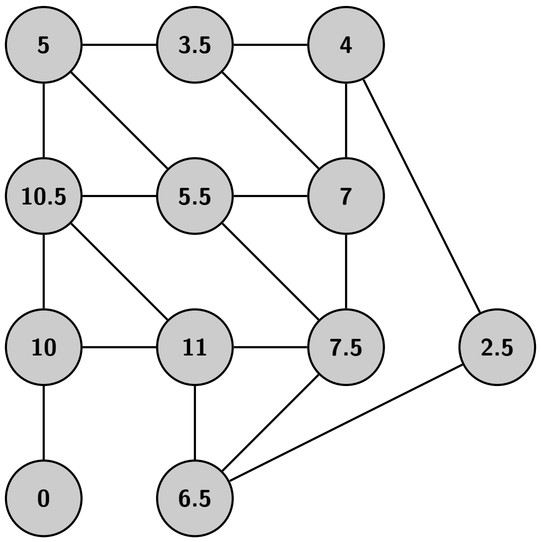

FIGURE 3. This graph is the line graph represented in Figure 2 with the values of betweenness identified.

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 0 & 3 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 3 & 0 & 5 & 0 & 0 & 0 & 6 & 0 & 0 \\ 0 & 0 & 5 & 0 & 7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 9 & 0 \\ 0 & 4 & 0 & 0 & 0 & 8 & 0 & 10 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 & 0 & 10 & 0 & 0 & 11 \\ 0 & 0 & 0 & 0 & 0 & 9 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 11 & 12 & 0 \end{bmatrix}. \tag{3}$$

Now, we can use this altered adjacency matrix to create the line graph adjacency matrix. Starting in the top left corner of the matrix, we search for the first nonzero value. In this example, the first nonzero value should be 1. Then, we examine the value's entire row and column, searching for other nonzero values. In 1's row, there is a value of 2. Values of 3 and 4 also occur in its column. Essentially, this equates to a connection within the line graph between vertices 1 and 2, vertices 1 and 3, and concurrently, vertices 1 and 4. We continue this process for all of the values in the upper right triangle of the matrix in Equation 3. All of the connections recorded are translated into the line graph matrix which is

$$\begin{bmatrix} 0&1&1&1&0&0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&1&1&0&0&0 \\ 1&0&0&1&1&1&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&1&0&1&0&0 \\ 0&0&1&0&0&1&1&0&0&0&0&0 \\ 0&0&1&0&1&0&0&0&0&1&1&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&1&0&1&0&0&0&0&1&1&0&0 \\ 0&1&0&0&0&0&0&1&0&0&0&1 \\ 0&0&0&1&0&1&0&1&0&0&1&0 \\ 0&0&0&0&0&1&0&0&0&1&0&1 \\ 0&0&0&0&0&0&0&0&1&0&1&0 \end{bmatrix}. \tag{4}$$

The line graph matrix can now be used to find cut vertices and calculate betweenness.

4.2. **Cut Vertices.** We took a depth-first search approach when determining cut vertices of the graph from its adjacency matrix. We used code developed by Sedgewick and Wayne [5] and adapted it to pass in the line graph adjacency matrix to be entered as an array. In determining cut vertices, the program uses three methods: *biconnected*, *dfs* and *adjacent*. Outlines of these algorithms can be found in Algorithm 1, Algorithm 2 and Algorithm 3 respectively.

```
Function: to determine cut vertices in a line graph.
Input: lineGraphMatrix = adjacency matrix for line graph (2D int array)
Output: articulation = true or false for each vertex as a cut vertex or not
        (boolean array)
Initialize: low = -1 (int array of length of lineGraphMatrix), pre = -1 (int
array of length of lineGraphMatrix), articulation = (Boolean array of length
of lineGraphMatrix) foreach v do
    if (pre[v] == -1) then
        dfs(lineGraphMatrix,v,v);
    else
        go back to the beginning of current section
    end
end
```

Algorithm 1: The method for determining cut vertices is adapted from the work of Sedgewick and Wayne. [5]

4.3. **Betweenness.** We have created an algorithm which determines the betweenness value of every vertex within a graph. In its simplest form, our algorithm consists of counting and comparing shortest paths between vertices.

Our betweenness program consists of two major methods: *computeShortestPaths* and *computeBetweenness*. Outlines of these algorithms can be seen in Algorithm 4 and Algorithm 5 respectively. We used code from Peterson [4] for *computeShortestPaths* which evaluates distances and shortest paths between vertices in the line graph. Our method, *computeBetweenness*, calls upon *computeShortestPaths* when comparing matrices and determining betweenness values.

```
Function: perform depth-first search.
Input: G=adjacency matrix (2D int array), u=previously visited vertex (int),
       v=vertex that the dfs is being performed on (int)
Output: articulation = true or false for each vertex as a cut vertex or not
        (boolean array)
Initialize: children=0, pre[v]=cnt++, low[v]=pre[v].
foreach w in adjacent(G,v) do
    if (pre[w]==-1) then
        children++;
        dfs(G,v,w);
        low[v]=minimum between low[v], low[w];
        if (low[v] >= pre[v] and u ≠ v) then articulation[v] = true;
        /* non-root of dfs is a cut vertex if low[v] >= pre[v] */
    else if (w ≠ u) then low[v]=minimum between low[v], pre[v]; /* update
    low number,ignore reverse of edge leading to v */
end
if (u==v and children > 1) then articulation[v]=true;/* root of dfs is a
cut vertex if it has more than one child */
```

Algorithm 2: The method for performing depth first search is adapted from the work of Sedgewick and Wayne. [5]

```
Function: determines the neighboring vertices to a specific vertex.
Input: g = adjacency matrix of line graph (2D int array), v = vertex to
       determine neighbors of (int)
Output: neighbors = vertices that are neighbors of v (arraylist of integers)
Initialize: neighbors.
foreach row do
    if (g[row][v] == 1) then neighbors.add(row);
end
return neighbors
```

Algorithm 3: The method for determining adjacent neighbors to a vertex is adapted from the work of Sedgewick and Wayne. [5]

5. Results and Analysis Using Model Created by Demšar, Špatenková and Virrantaus [7]

Following the model implemented by Demšar, Špatenková and Virrantaus [7], vertices are assigned vulnerability risk values in order to incorporate the importance of both critical factors discussed in Section 3. The presented model is topological in nature with the length of the road segment not being considered. Through the use of cut vertices as a critical factor, this model places a higher priority with road segments that disconnect the city. Cut vertices are assigned the highest risk value of 1. All other vertices are assigned a risk value equal to its given normalised betweenness ranging from 0 to 1. Betweenness is useful in determining the importance of a road segment because people tend to navigate along shortest paths. So roads that are part of many shortest paths between two locations in a city would tend to be used more often and thus be more critical. Streets are intially ordered according

```
Function: computes the shortest paths for an adjacency matrix.
Input: n = number of vertices (int), adj = adjacency matrix of line graph
       (2D int array) Output: dists = matrix of distances for shortest paths
       (2D int array), nPaths = matrix of shortest paths (2D int array)
Output: dists = matrix of distances for shortest paths (2D int array),
        nPaths = matrix of shortest paths (2D int array)
Initialize: dists, nPaths and curAdj to 2D array of size n; power = 1;
nFound = -1; dists = 0; nPaths = 1.
foreach i,j where $i \neq j$ do
    curAdj[i][j] = adj[i][j];
    if (adj[i][j] == 0) then
        set dists[i][j], nPaths[i][j] = -1;
    else
        set dists[i][j], nPaths[i][j] = -1;
    end
end
while $nFound \neq 0$ do
    nFound=0;
    curAdj = $matrixProduct$(curAdj,adj);
    power++;
    foreach i,j where $i \neq j$ do
        if (dists[i][j]==-1 and $curAdj[i][j] \neq 0$) then
            dists[i][j] = power;
            nPaths[i][j] = curAdj[i][j];
            nFound++;
        else
            go back to the beginning of current section
        end
    end
end
```

Algorithm 4: The method for finding shortest paths between vertices is adapted from the work of Peterson. [4]

to risk value. If risk values are equal, then order is determined by betweenness values. The results of this model on the road network of Indianola, IA can be seen in Table 1.

In the city of Indianola, we found a weakness in this model occured when we assigned the higest risk value to cut vertices. Each cut vertex had a betweenness value ranging from 628 to 3738. When compared to other vertices' betweenness values, these figures are significantly lower. For example, the highest betweenness value was 27815. Thus, the importance of the cut vertices seems to be overestimated relative to the importance of other vertices within the entire city of Indianola.

Moreover, we found the removal of the cut vertex with the highest value of betweenness only disconnects six road segments from the map relative to the total of 630 road segments in Indianola. Every other cut vertex disconnected a lesser number of road segments from the rest of the city. In our example, we found the

```
Function: computes the betweenness value of a given vertex.
Input: matrix = adjacency matrix of line graph (2D int array),
       totalShortestPaths = original nPaths of adjacency matrix (2D int
       array), originalDists = original distances of adjacency matrix (2D int
       array), vertex = vertex to compute betweenness value for (int)
Output: sum = the betweenness value of vertex(double)
foreach x,y do
    if (nPaths[x][y]==-1) then
        nPaths[x][y] = 0;
    else
        go back to the beginning of current section
    end
    if ($dists[x][y] > originalDists[x][y]$) then
        nPaths[x][y]=0;
    else
        go back to the beginning of current section
    end
end
foreach row,col in the upper right triangle from main diagonal in matrices do
    if ($totalShortestPaths[row][col] > nPaths[row][col]$ and $row \neq vertex$
    and $col \neq vertex$) then
        sum+=$\frac{totalShortestPaths[row][col]-nPaths[row][col]}{totalShortestPaths[row][col]}$;
    else
        go back to the beginning of current section
    end
end
return sum;
```

Algorithm 5: The method that computes the betweenness value of a given vertex.

cut vertex of the line graph to be labeled 5. As seen in Figure 1, the removal of the edge labeled 5 will only disconnect one road segment from the rest of the map. With respect to the entire map of Indianola and not just the portion shown in Figure 1, the betweenness value of Vertex 5 in this paper is actually 628. Relative to the highest value of about 27815 in the city, the betweenness of this vertex is insignificant. All of these cut vertices were located along the outskirts of the city. Their lower betweenness values when compared to other road segments can be attributed to their location within the city. It appears that within small, structured towns such as Indianola, discovering a cut vertex which disconnects large portions of the town is rare, if impossible. Therefore, we found the cut vertices to be overall insignificant.

6. Modified Model

Due to the relative insignificance of cut vertices, we created a second model which relies strictly on betweenness. The results of this model are more accurate than the previous model when determing critical locations. Table 2 displays the top ten most critical locations according to our modified model. Notice that none of

Order	Street	Intersections	Betweenness	Risk Value
1	C St.	Girard Ave. & Clinton Ave.	27815	1
2	9th St.	G-36 & Summit Place	3738	1*
3	E. 12th St.	Highway 65/69 & S. 3rd St.	2500	1*
4	N. O St.	Jackson Ave. & Iowa Ave.	1254	1*
5	Country Club Road	Quail Ridge & Scenic Valley Dr.	628	1*
5	N. O St.	W. Kentucky & Jackson Ave.	628	1*
5	W. Henderson Ave.	N. U St. & N. T Court	628	1*
5	Boston Ave.	17th St. & 18th St.	628	1*
5	20th Ct.	Highway 92 & E. 1st Ave.	628	1*
10	Iowa Ave.	6th St. & 8th St.	23130	0.8316
*The risk value of 1 was assigned as this road segment is a cut vertex.				

Table 1. Our results based on the model created by Demšar, Špatenková and Virrantaus [7]. The values of betweenness have been truncated.

these locations are cut vertices, whereas the last model included eight cut vertices in the top ten.

Figure 4 displays the portion of Indianola which surrounds C Street (the most critical location according to our new model) which is represented by the edge labeled 13. The edge labeled 12 represents D Street. D Street and C Street, while virtually in the same location and serve the same purpose, are assigned vastly different betweenness values according to our modified model. The betweenness values for C Street and D Street are 27185 and 2352, repectively.

The difference in betweenness values of C and D Street allowed an observation of the major weakness in this modified model. Unfortunately, this model did not incorporate a distance factor for each road segment. Our algorithm for betweenness did not consider the lengths of the individual road segments. A road segment which spans a block in the city is considered the same length as a road segment which spans a mile. Thus, the calculated shortest distances are based on the shortest number of road segments and not the shortest total distance relative to time. Consequently, C Street, as one of the longest streets in the town, had a much larger betweenness value than D Street, a street of slightly less length, but located parellel to C Street. The distances of C and D street are shown in Figure 4. With this model, the road segments with the highest betweenness values were also some of the longest road segments in Indianola.

7. Standardized Betweenness Model

The models discussed in Section 5 and Section 6 neglected the comparative length of individual road segments when computing betweenness. However, the city of Indianola has many streets that do not have intersections at standard distances. For instance, in Figure 4, if one was traveling from vertex 3 to vertex 8, the shortest distance between these two points would be along edge 3 and down Buxton Street to vertex 8. However, our program goes along edge 2, down C Street, and across Clinton Avenue to vertex 8 which is the shortest path. Our program identifies this route as the shortest path because C Street has few crossroads and is one of the

Order	Street	Intersections	Betweenness
1	C St.	Girard Ave. & Clinton Ave.	27185
2	Iowa Ave.	6th St. & 8th St.	23130
3	Iowa Ave.	6th St. & Highway 65/69	22140
4	Plainview Ave.	Highway 65/69 & 104th Ave.	19399
5	Iowa Ave.	Stephen Ct. & E St.	18268
6	150th Ave.	Iowa Ave. & Highway 92	16988
7	Highway 65/69	Iowa Ave. & Girard Ave.	16926
8	Girard Ave.	C St. & B St.	16809
9	Highway 65/69	E. 12th Ave. & 10th St.	16539
10	Iowa Ave.	8th St. & 9th St.	15675

Table 2. This is the results from using the modified model based strictly on betweenness. The values of betweenness have been truncated.

Order	Street	Intersections	Standardized Betweenness
1	Highway 65/69	Girard Ave. & Iowa Ave.	50217
2	Highway 92	J St. & 3rd Ave.	49938
3	Iowa Ave.	Highway 65/69 & east 370 ft of Highway 65/69	48735
4	Iowa Ave.	6th St. & 370 ft west of 6th St	48326
5	Highway 92	K St. & J St.	47648
6	Iowa Ave.	6th St. & 370 ft east	46443
7	Iowa Ave.	370 ft East of 6th St. & 370 ft west of 8th St.	45981
8	Iowa Ave.	8th St. & 370 ft west of 8th St.	45544
9	Highway 92	3rd Ave. & G St.	42021
10	Highway 65/69	Girard Ave. & Franklin Ave.	37130

Table 3. These are the results found using the Standardized Betweenness model which gives every street segment approximately the same length. The values of betweenness have been truncated.

longest road segments in the city. Our previous models give C Street a high ranking even though it is not a widely used street within our city.

In order to compensate for the length of the road segments, we divided the roads by placing vertices between intersections. We created a method of dividing road segments by comparing each road segment to one standard block along the downtown square. We used the map of Indianola [6] and Google maps [1] to find a conversion rate: according to our map, 1/2 inch equals roughly 370 ft. Therefore, along each road segment, we created an intersection roughly every 370 ft. Ideally, every road segment becomes equal. When applying this method to the example shown in Figure 4, the result is Figure 5. Notice the additional vertices added to C Street and D Street.

Because every road segment is of standard length, we believe the Standardized Betweenness model is even more accurate than our original model or our modified model. In addition, the final results compare intuitively to actual road usage in Indianola. The top ten most critical road segments are recorded in Table 3.

While analyzing these results, an intriguing pattern emerged: all of these critical locations surround points of interest within the city, namely grocery stores. Most noteably, the critical locations exist around the town's Fareway and Hyvee.

At their current locations, Fareway was established in 1965 and Hyvee in 1999. Fareway is located in the older, southwestern side of town, whereas Hy-Vee is located in the newer, northeastern area. We conjecture that the creation of each respective grocery store caused infrasture development within its area. Therefore, as neighborhoods and streets advanced, the area became increasingly more important, eventually creating the most critical locations in the city. It is also possible that these areas have always been critical to the navigation of the city. In this case, the grocery stores capitalized on the importance of these road segments.

8. Comparison Analysis of Models

In an effort to assess the contribution of our Standardized Betweenness Model, we ran data analysis using R (a statistical computing program) to compare our results with those found using the model created by Demšar, Špatenková and Virrantaus [7]. In particular, we examined the road segments that had not been subdivided within our Standardized Betweenness Model since our goal was to alter the ranking of these road segments without fundamentally changing the ranking model created by Demšar, Špatenková and Virrantaus [7].

Figure 6 is a representation of our data analysis. Clearly there is a linear correlation between the rankings of our Standardized Betweenness Model and the model presented by Demšar, Špatenková and Virrantaus [7]. This coorelation shows that the majority of road segments which were not subdivided did not experience a significant shift in their individual rankings. The percentage of road segments that had a ranking change of less than 10% was 49.13%. The percentage of road segments that changed less than 15% was 61.99%.

While our data analysis demonstrated a linear correlation between the results of each model, we also found a few outliers. We conjectured that these outliers were road segments adjacent to the altered road segments. If a given road segment was assigned a high betweenness value, we would expect the surrounding road segments to also have relatively high betweenness values. Once the given road segment is subdivided to standardize road length, we would expect the adjacent road segments criticality to be affected as well. Our data analysis supports this assertion. Of the road segments that changed in ranking greater than 15%, we found 67.49% of those road segments were adjacent to the road segments that our Standardized Betweenness Model manipulated.

Our Standardized Betweennes Model determines critical locations best in cities in which roads are not uniform in length. Furthermore, the importance of cut vertices as a critical factor is dependent on the structure of the city. For example, the cut vertices of the road network of Indianola exist on the outer limits of the city as the center of the city is more structured. However, the road network of Helsinki, Finland as discussed in the work of Demšar, Špatenková and Virrantaus [7] includes cut vertices within the center of the city.

9. Conclusions and Further Research

In this paper, we presented three models that determine critical locations in a city. Within these models critical factors of cut vertices and betweenness values were considered. The first model focused on both calculations, with an emphasis on cut vertices. After discovering the relative insignificance of cut vertices, the second model relied on betweenness values. This model, although improved, found to have a major weakness. It failed to consider the differing distances of road segments. In order to eliminate this weakness, we developed a third model that standardized distance. Our research suggests this model to be more successful than previous models in identifying critical locations.

In addition to our developments, there are still enhancements to be explored. Further exploration could include research on the infrastructure around grocery stores to investigate a potential motive behind the placement of Hy-Vee and Fareway (previously discussed in Section 7). Also, we could implement the restrictions of one way streets in our model. Finally, the application of our model to other cities could solidify our conclusions.

10. Acknowledgements

We would like thank our advisor, Dr. Debra Czarneski, for her guidance throughout this research. We would also like to thank Dr. Albert H. and Greta A. Bryan for their generosity and for making this summer research experience possible. Finally, we would like to thank Simpson College for assistance throughout the project.

References

[1] Google maps. https://maps.google.com/.

[2] 2010 population finder, 2010. http://www.census.gov/popfinder/?fl=19:1938280.

[3] Linton Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1:215–239, 1978/79.

[4] E. Peterson. Betweencentrality.java, 2010. https://code.google.com/p/blaisemath/source/browse/extensions/BlaiseGraphTheory/src/org/bm/blaise/scio/graph/metrics/ BetweenCentrality.java?r=657.

[5] R. Sedgewick and K. Wayne. Biconnected.java, 2010. http://algs4.cs.princeton.edu/41undirected/Biconnected.java.html.

[6] Record Herald & Indianola Tribune. Indianola city map, 2009. http://www.indianolaiowa.gov/About/CityMap.aspx.

[7] O. Špatenková U. Demšar and K. Virrantaus. Identifying critical locations in a spatial network with graph theory. *Transactions in GIS*, 12(1):61 – 82, 2008.

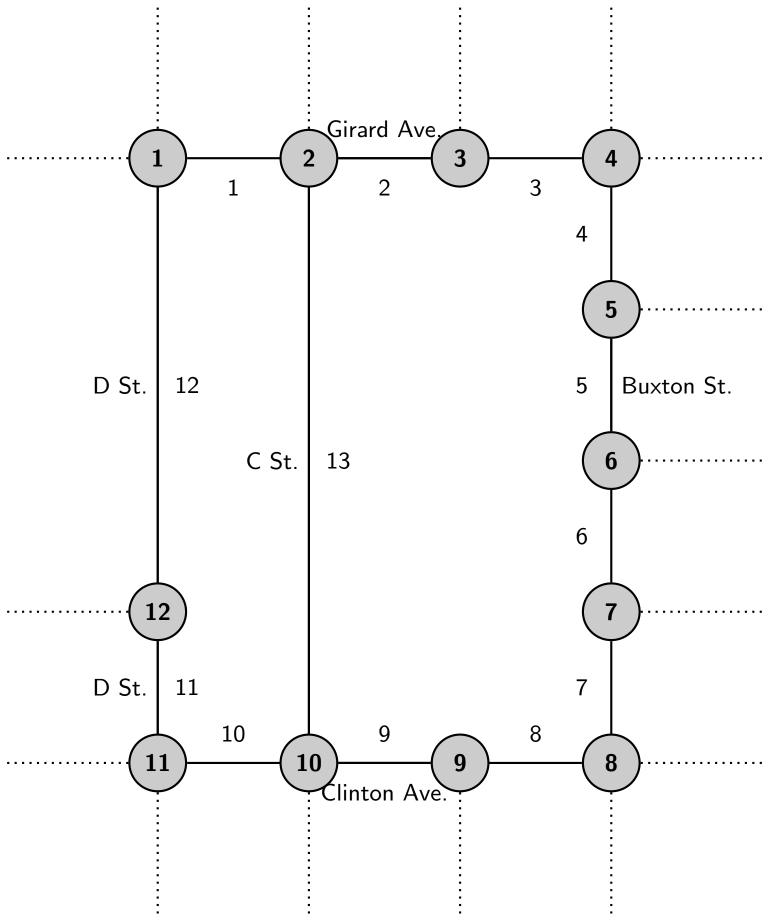

FIGURE 4. The graphical representation of a portion of Indianola, IA. The dotted lines represent other road segments which we neglected for the purpose of the example.

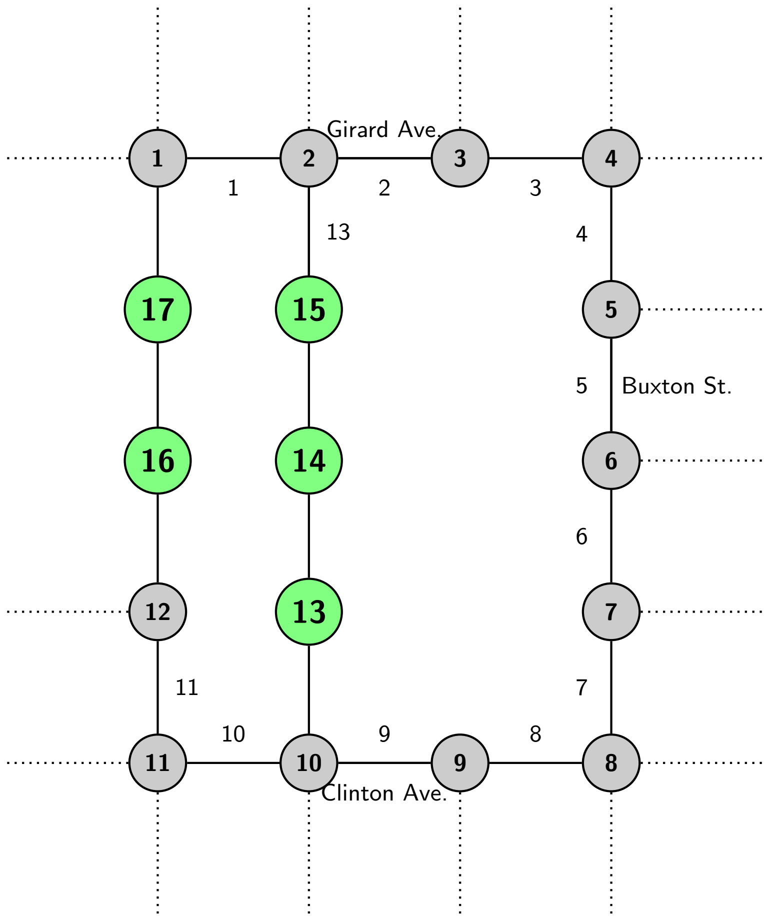

FIGURE 5. The graphical representation of a portion of Indianola, IA according to our Standard Betweenness Model. The dotted lines represent road segments we did not include in this example.

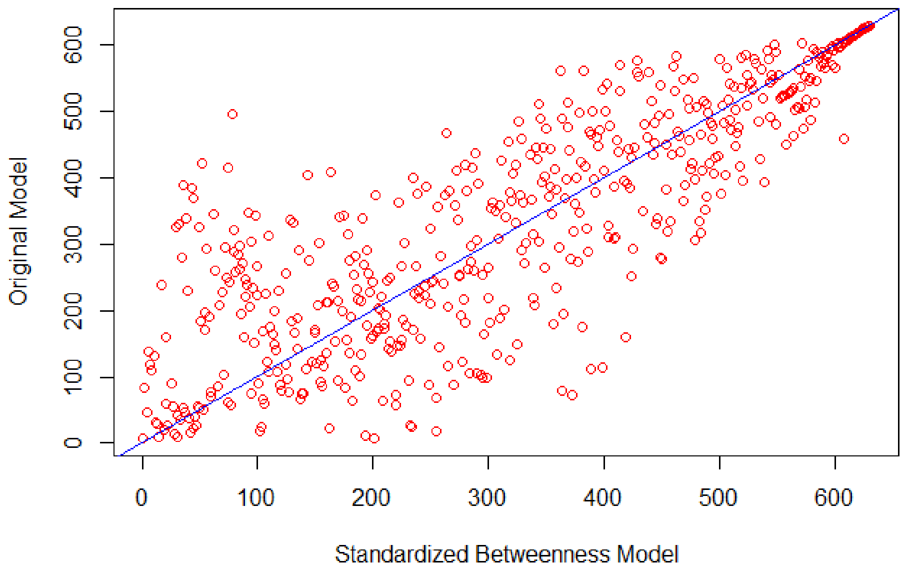

Figure 6. This plot shows a linear correlation between the non-manipulated road segment rankings of our Standardized Betweenness Model and the model created by Demšar, Špatenková and Virrantaus [7]. The line plotted is $y = x$.