# Improved Network Bridge Cutting

Aaron Zampaglione
Florida Institute of Technology
Melbourne, Florida
azampagl2007@my.fit.edu

## ABSTRACT

In this paper, we introduce three improvements to the BridgeCut algorithm introduced in [1]: rank tie breaking, depth bridging coefficient, and re-ranking.

## 1. INTRODUCTION

The algorithm introduced by Hwang is a very effective technique for producing clusters in networks using the concept of bridges. It is an improvement over previous bridge identification methods, using a local metric (bridging coefficient) as well as a global metric (betweenness). Idealistically, we would like to create clusters that are as large as possible, as long as they meet some threshold (in this case, a density threshold). However, there are cases in which the original BridgeCut algorithm provided in [1] would split edges or vertices and create lots of small, very dense, clusters. Take for example, Figure 1:

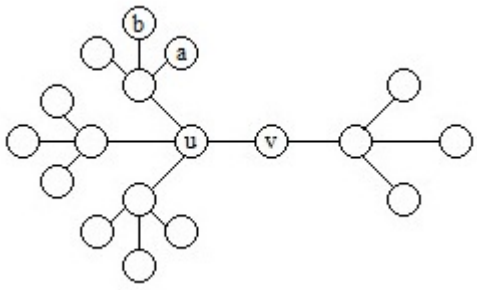


**Figure 1**

Using definition 4 in [1], we find the bridging centrality of nodes *u* and *v*:

$$C_{Br}(u) = 4\ x\ 2 = 8$$

$$C_{Br}(v) = 3\ x\ 2 = 6$$

Assuming a higher rank is preferable, the BridgeCut algorithm will split on *u* and create 3 clusters, each of size 4, that have a density of 0.5 and 1 cluster of size 5 with a density of 0.4.

Now assume that we had a density threshold of 0.25. If the algorithm had split on *v*, rather than *u*, BridgeCut would have created a cluster of size 13 that had a density of 0.26 and a cluster of size 4 that had a density of 0.4. Both of the clusters meet the density threshold, and are larger than the clusters that were obtained when using *u* as the bridge.

In this paper, we introduce two improvements, rank tie breaking and depth bridging coefficient, which will alleviate the original BridgeCut algorithm from this predicament.

The paper is structured as follows: Section 2 will introduce work related to the subject of improving bridge detection in graphs. Section 3 will describe in detail both the rank tie breaking and depth bridging coefficient improvements. Section 4 will provide the metrics used to evaluate the new algorithm. The next section will run the algorithm on real world results and compare to the original BridgeCut algorithm and another improved version found in [2]. Section 6 will then introduce possible complications with the improved approach. Finally, the paper will conclude with an overall summary, caveats to this approach, and future improvements.

## 2. RELATED WORK

Below are works that used different metrics in order to find ideal bridges (and clusters) in networks:

Kotz and Nanda in [2] introduce the concept of an LCB (Localized Bridging Centrality) metric. Their approach substitutes the global betweenness centrality in Hwang's approach for an "egocentric" (local) betweenness centrality metric. Although the approach promises higher efficiency, there is no guarantee that the "egocentric" (local) betweenness metric is more effective than the "sociocentric" (global) betweenness metric.

Bonaich [3] proposes the idea of using adjacency matrices to find central nodes in a network. A node's centrality is determined by the "summed connection to others" (direct neighbors). Although this approach might solve the predicament in the introduction, this approach ignores the global perspective of a node's centrality.

In "Ranking of Closeness Centrality for Large-Scale Social Networks", Oakamoto, Chen, and Li [4] broach the concept of closeness centrality. The closeness centrality of a node is the inverse of the average shortest path distance from that node to any other node in the network. Essentially, it looks for nodes that are highly connected to other nodes by a short distance. However, like the betweenness centrality, it is mostly a global metric and lacks the local perspective that Hwang's approach includes.

## 3. APPROACH

### 3.1 Rank Tie Breaking

In the original paper, there is a chance that multiple vertices (or edges) could end up with the same bridging centrality ranking. By default, the algorithm chooses the vertex, or edge, that is


*Melbourne, Florida, United States of America*

processed first. This is not ideal because no heuristics are considered in the selection process. In order to handle this dilemma, the following is proposed:

Assuming multiple vertices have the same bridging centrality ranking, we will choose the vertex with the lowest degree. We choose the vertex with the lowest degree assuming the removal will create a small amount of large clusters. If we were to remove the vertex with the highest degree, there is a larger possibility of multiple small clusters with a density way over the threshold (as mentioned in the introduction, we would like large clusters, even if it means they barely pass the threshold).

Assuming multiple edges have the same bridging centrality, we will choose the edge that has the lowest average degree between the edge's two nodes for the same reasoning.

In the unlikely event that the bridging centrality is the same and the degrees of the vertices or edges are the same, then the algorithm will default to the first-come, first-serve method as presented in the original paper.

## 3.2 Depth Bridging Coefficient (DBC)

The objective of the depth-bridging coefficient (DBC) is to provide a metric to the BridgeCut algorithm that considers a "local" neighborhood instead of just a direct neighborhood.

We will define the depth-bridging coefficient for a vertex as:

**Equation 1**[1]

$$\psi_d(v) = \frac{1}{|N_d(v)|} \sum_{i \in N_d}^{i} \frac{\delta_d(i)}{d(i) - 1}$$

where $N_d$ are the neighbors of a node at a minimum of depth $d$ (by minimum, we imply that if a node has a shorter path to v than $d$, than it is not a neighbor of depth d) and $\delta_d$ is the number of edges leaving the neighborhood of depth $d$.

The depth-bridging coefficient for an edge is defined as:

**Equation 2**[2]

$$\psi_d(e) = \frac{|N_d(i)|\psi_d(i) + |N_d(j)|\psi_d(j)}{(|N_d(i)| + |N_d(j)|)(|C_d(i,j)| + 1)}, \qquad e(i,j) \in E$$

where $N_d$ are the neighbors of a node at a minimum of depth $d$ and $C_d$ is the number of common neighbors for nodes $i$ and $j$ at depth $d$.

## 3.3 Re-Ranking

The BridgeCut algorithm will perform its normal process in determining the rankings for all of the nodes, or edges, using definitions 4 and 5 in [1], respectively. However, instead of returning the top ranked vertex or edge, we consider the top $k$ percent of vertices or edges and run a new ranking calculation:

**Equation 3**

$$RRC_{Br}(v) = R_{\phi}(v) \cdot \sum_{1..k}^{d} R_{\psi_d}(v)$$

**Equation 4**

$$RRC_{Br}(e) = R_{\phi}(e) \cdot \sum_{1..k}^{d} R_{\psi_d}(e)$$

The new ranking algorithm will take into account the bridging coefficient at k levels. Once the re-ranked centrality is performed, the algorithm will return the highest-ranking vertex or edge, and continue as normal.

## 3.4 A Simple Example

For demonstration purposes, provided below is a step-by-step analysis of the vertex re-ranking process on the example graph depicted in Figure 1. As mentioned in section 3.3, the BridgeCut algorithm performs its normal process and determine the ranks of all the vertices in the graph. Once obtained, only the top 20% of nodes are evaluated for re-ranking:

$$C_{Br}(u) = 4\ x\ 2 = 8$$
$$C_{Br}(v) = 3\ x\ 2 = 6$$
$$C_{Br}(a) = 1\ x\ 2 = 2$$
$$C_{Br}(b) = 1\ x\ 2 = 2$$

For each of the vertices, the algorithm evaluates the bridging coefficient at a max depth of 2. Since the bridging coefficient for a depth of 1 was already obtained during the first phase of the algorithm, the calculation for the bridging coefficient at depth 2 was the only step necessary:

$$\psi_2(u) = \frac{1}{10}\left(0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + \frac{3}{3}\right) = 0.1$$
$$\psi_2(u) = \frac{1}{6}\left(\frac{3}{3} + \frac{3}{3} + \frac{3}{3} + 0 + 0 + 0\right) = 0.5$$
$$\psi_2(a) = \frac{1}{3}\left(0 + 0 + \frac{3}{3}\right) = 0.333$$
$$\psi_2(b) = \frac{1}{3}\left(0 + 0 + \frac{3}{3}\right) = 0.333$$

Ranking the scores for nodes $u$, $v$, $a$, and $b$ at a depth of 2 returns 1, 3, 2, and 2 respectively. The ranks at each depth are then summed and multiplied by their respective betweenness centrality:

$$RRC_{Br}(u) = 4\ x\ (2 + 1) = 12$$
$$RRC_{Br}(v) = 3\ x\ (2 + 3) = 15$$
$$RRC_{Br}(a) = 1\ x\ (2 + 2) = 4$$
$$RRC_{Br}(b) = 1\ x\ (2 + 2) = 4$$

Clearly, vertex $v$ now out-ranks vertex $u$. BridgeCut will choose to split on vertex $v$ and return a cluster of size 13 that has a density of 0.26 and a cluster of size 4 that has a density of 0.4; our original objective.

# 4. EVALUATION

Performance of the improved BridgeCut algorithm was evaluated using silhouette coefficient, Davies-Bouldin index, and clustering

[1] At depth 1, the equation is identical to definition 2 in [1].

[2] At depth 1, the equation is identical to definition 3 in [1].

coefficient on the original network. Both silhouette coefficient and Davies-Bouldin index are ideal because the modification introduced in this paper purely focuses on improving the quality of the resultant clusters without any domain knowledge. Clustering coefficient is useful for evaluating the vertex and edge selection process of the algorithm and can additionally be used to compare the original BridgeCut algorithm to the improved algorithm presented in this report.

## 4.1 Silhouette Coefficient

Silhouette Coefficient, referenced in "Introduction to Data Mining" [7], compares a vertex's cohesion to it's own cluster versus it's separation to all other vertices. To calculate the Silhouette Coefficient of a single vertex, the following formula can be used:

**Equation 5**

$$s_i = \frac{b_i - a_i}{\max(b_i, a_i)}$$

where $a_i$ represents the vertex's distance to all other vertices' in it's cluster and $b_i$ represents the distance to the center of the nearest cluster that is not it's own. To find the Silhouette Coefficient of a solution, the average of all the vertices' Silhouette Coefficient's is calculated.

The metric helps determine if clusters intersect with one another, and if so, by how much. Ideally, clusters will not intersect with one another and will be far apart. Therefore, a solution should look to maximize this metric (with a maximum of 1).

## 4.2 Davies-Bouldin Index

Davies-Bouldin index takes in to account the topological quality of clusters [1]. For each cluster, it finds a comparable cluster that created the worst-case scenario: two large clusters that are very close. Once each cluster is evaluated, it finds the "average" worst-case for the clustering solution.

**Equation 6**

$$DB = \frac{1}{k}\sum_{i=1}^{k} max_{i \neq j}\left[\frac{diam(C_i) + diam(C_j)}{d(C_i, C_j)}\right]$$

Normally, the goal is to minimize this metric; i.e. a clustering solution that creates compact, distant clusters. However, we are looking to create "large" (more meaningful) distant clusters. Although we are still looking to minimize this metric, there is a possibility that the results of the improved algorithm will be slightly larger than the normal BridgeCut algorithm due to the possible larger cluster size.

## 4.3 Clustering Coefficient

Clustering Coefficient measures the interconnectedness of a vertex's direct neighborhood. Ideally, a cluster should be highly interconnected (maximizing this metric).

**Equation 7**

$$C_v = \frac{2\left|\cup_{i,j \in N(v)} e(i,j)\right|}{d(v)(d(v)-1)} : e(i,j) \in E$$

To find the clustering coefficient of a solution, a simple average can be calculated over all of the produced clusters.

## 4.4 Singleton Dilemma

Unfortunately, both silhouette coefficient and Davies-Bouldin Index are very weak when evaluating singleton clusters. Neither metric properly handles the concept that singletons are not desired. On the contrary, both metrics actually return better results with more singletons. With this under consideration, one must note the possible quirks in using these metrics to evaluate clusters.

# 5. RESULTS

To demonstrate the advantages of the improved BridgeCut algorithm, it was compared to the original algorithm and the LBC BridgeCut algorithm provided in [2]. The two data sets selected to evaluate the algorithms were the "Books about US Politics" [5] data set and the "Les Miserables" [6] data set.

## 5.1 "Les Miserables"

The "Les Miserable" data set is a co-appearance network of characters in the novel "Les Miserables".

### *5.1.1 Comparing Bridging Coefficient Depths*

The first step was to determine at which depth the improved Bridging Coefficient performed best. We ran the improved BridgeCut algorithm up to a depth of 4 and plotted the results:

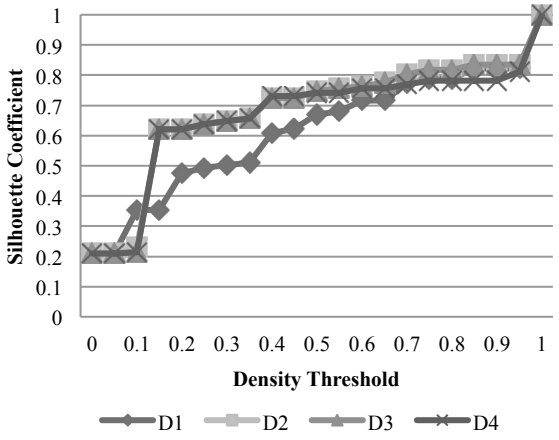


**Figure 2: Comparing Depths for Vertex Centrality using Silhouette Coefficient vs. Density Threshold**

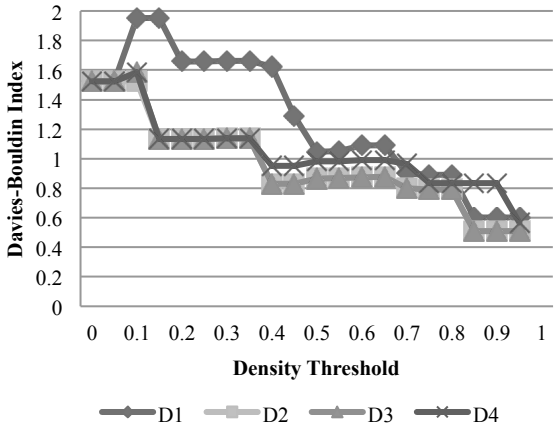


**Figure 3: Comparing Depths for Vertex Centrality using Davies-Bouldin Index vs. Density Threshold**

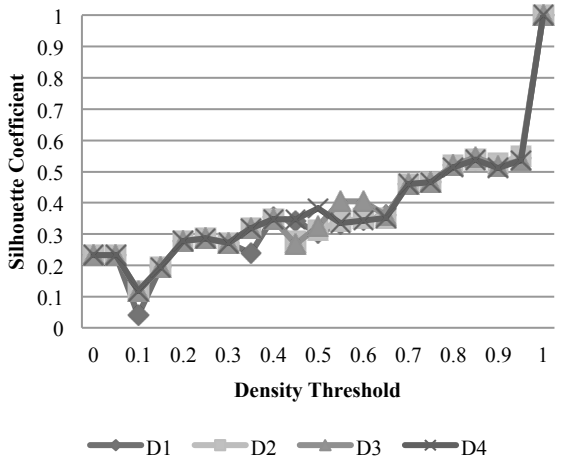


**Figure 4: Comparing Depths for Edge Centrality using Silhouette Coefficient vs. Density Threshold**

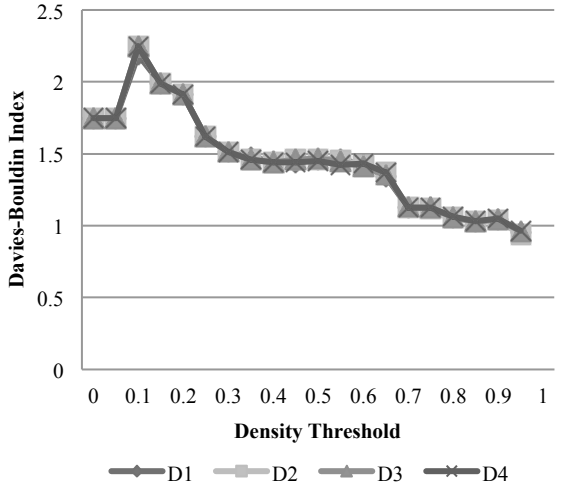


**Figure 5: Comparing Depths for Edge Centrality using Davies-Bouldin Index vs. Density Threshold**

From Figure 2 and Figure 3, we can clearly see that a depth based bridging coefficient was beneficial for vertex centrality based on the silhouette coefficient and Davies-Bouldin index. On average, vertex centrality at greater depths outperformed vertex centrality using a bridging coefficient at depth one.

Visually, there didn't seem to be a distinct difference in the varying depths for edge centrality based on Figure 4 and Figure 5. However, a close analysis of the raw data revealed that an edge bridging coefficient at depths two and three slightly outperformed a depth of one.

Based on these findings, it would appear that the optimal depth for both vertex centrality and edge centrality is two for the "Les Miserables" data set.

### *5.1.2 Comparison of Algorithms*

Once the best depth was chosen, we compared the original algorithm, our improved algorithm at the best depth (two), and a different improvement deemed "Localized Bridging Centrality" presented in [2].

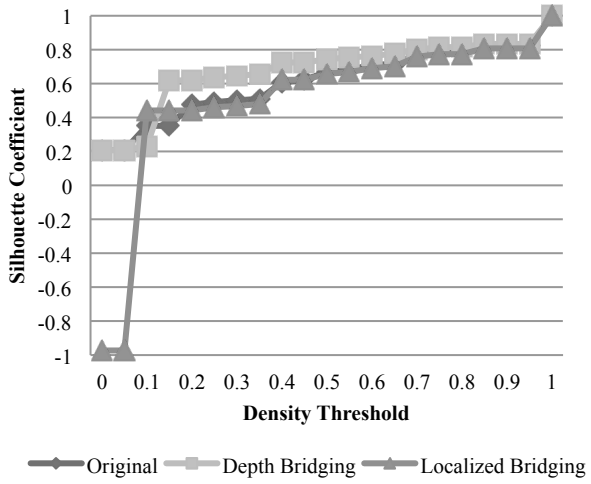


**Figure 6: Comparison of Algorithms for Vertex Centrality using Silhouette Coefficient vs. Density Threshold**

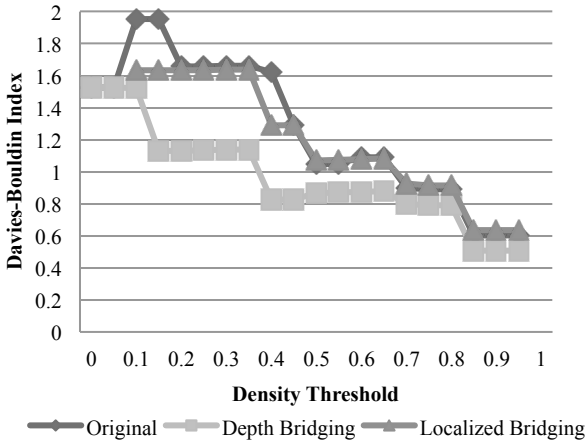


**Figure 7: Comparison of Algorithms for Vertex Centrality using Davies-Bouldin Index vs. Density Threshold**

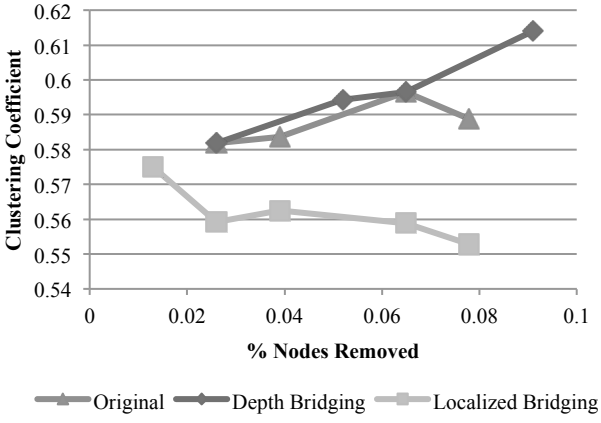


**Figure 8: Comparison of Algorithms for Vertex Centrality using Clustering Coefficient vs. Nodes Removed (0.7 DT)**

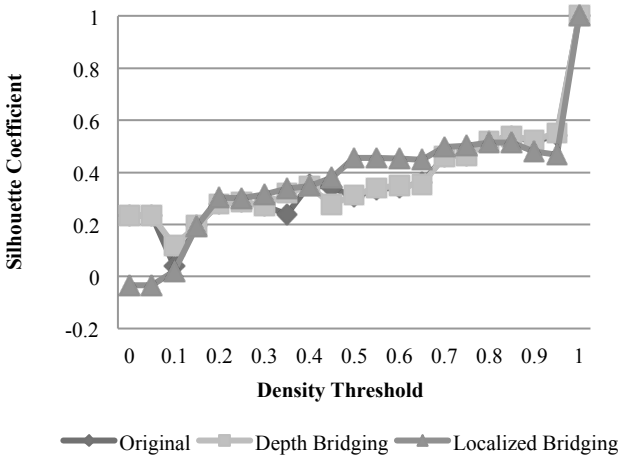


**Figure 9: Comparison of Algorithms for Edge Centrality using Silhouette Coefficient vs. Density Threshold**

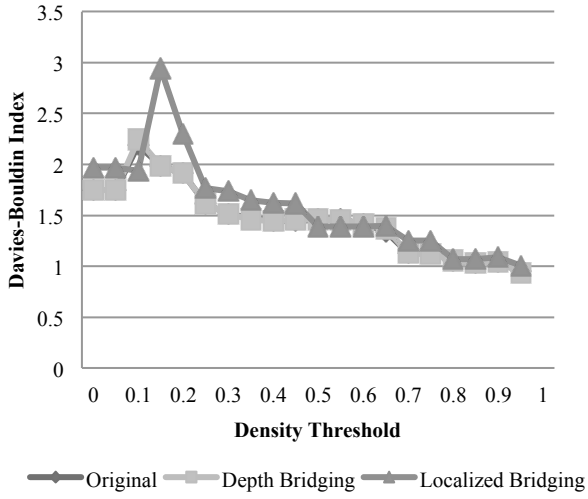


**Figure 10: Comparison of Algorithms for Edge Centrality using Davies-Bouldin Index vs. Density Threshold**

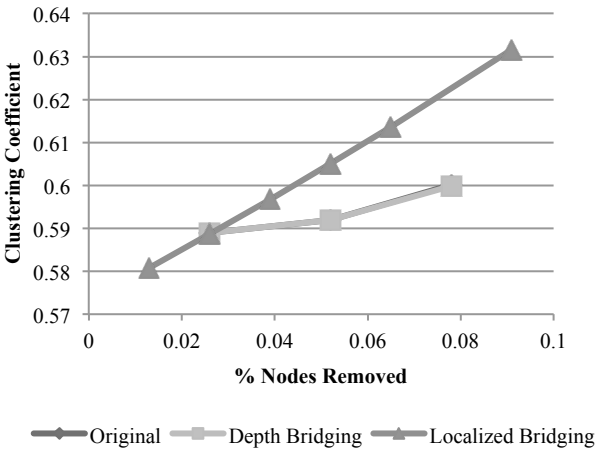


**Figure 11: Comparison of Algorithms for Edge Centrality using Clustering Coefficient vs. Nodes Removed (0.7 DT)**

On average, the depth BridgeCut algorithm outperformed the original BridgeCut and the LBC improvement on all three metrics for vertex centrality. According to the clustering coefficient metric, the DBC implementation was the only one to constantly select "high-valued" nodes (constant positive slope); the other two algorithms began to decline after removing approximately 7% of the nodes from the original graph.

There was no notable difference between the algorithms for edge centrality. All the algorithms were on par with one another, with depth bridging slightly outperforming the other two according to the silhouette coefficient and Davies-Bouldin index. The LBC implementation slightly outperformed the DBC algorithm, but both had a constant positive slope, which is desired.

## 5.2 "Books About US Politics"

The "Books About US Politics" obtained from [5] is a network of books about recent US politics. For this data set, we performed the same experiments and evaluations with the same parameters.

### *5.2.1 Comparing Bridging Coefficient Depths*

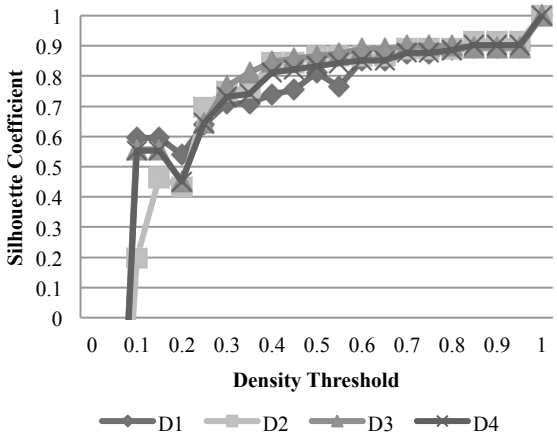


**Figure 12: Comparing Depths for Vertex Centrality using Silhouette Coefficient vs. Density Threshold**

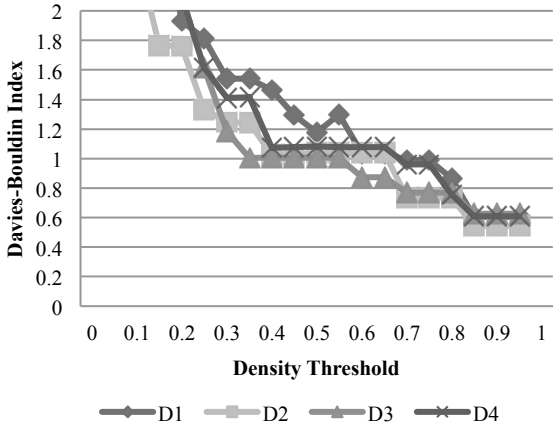


**Figure 13: Comparing Depths for Vertex Centrality using Davies-Bouldin Index vs. Density Threshold**

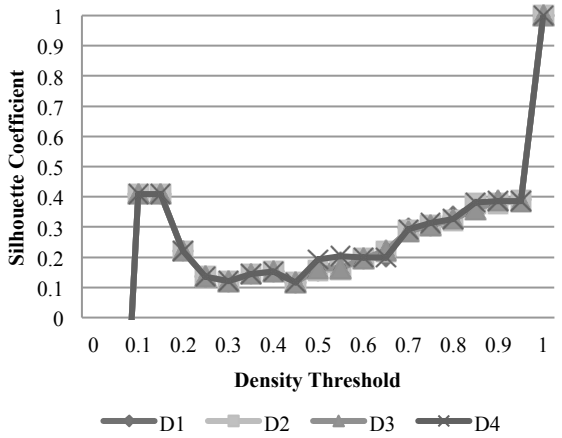


**Figure 14: Comparing Depths for Edge Centrality using Silhouette Coefficient vs. Density Threshold**

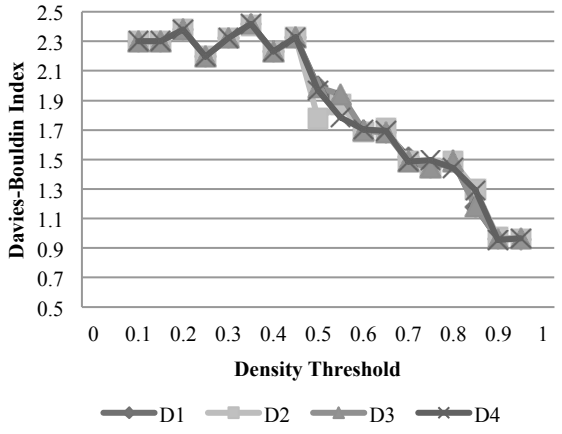


**Figure 15: Comparing Depths for Edge Centrality using Davies-Bouldin Index vs. Density Threshold**

Once again, the depth bridging coefficient, especially at a depth of 2, seems to outperform the original algorithm (depth 1) in all cases.

### *5.2.2 Comparison of Algorithms*

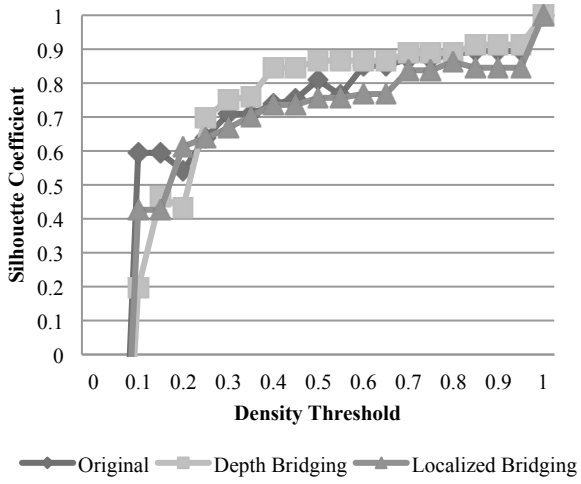


**Figure 16: Comparison of Algorithms for Vertex Centrality using Silhouette Coefficient vs. Density Threshold**

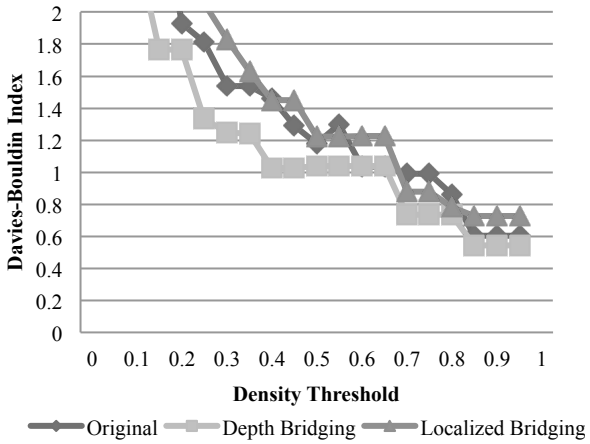


**Figure 17: Comparison of Algorithms for Vertex Centrality using Davies-Bouldin Index vs. Density Threshold**

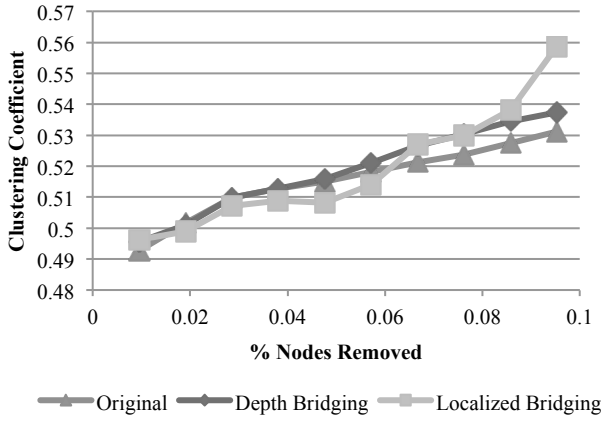


**Figure 18: Comparison of Algorithms for Vertex Centrality using Clustering Coefficient vs. Nodes Removed (0.7 DT)**

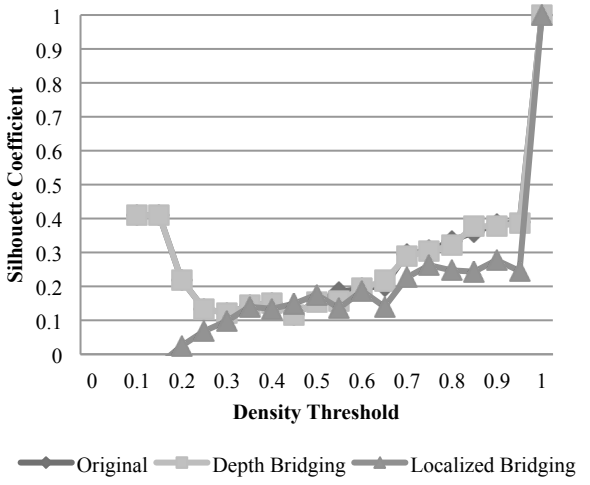


**Figure 19: Comparison of Algorithms for Edge Centrality using Silhouette Coefficient vs. Density Threshold**

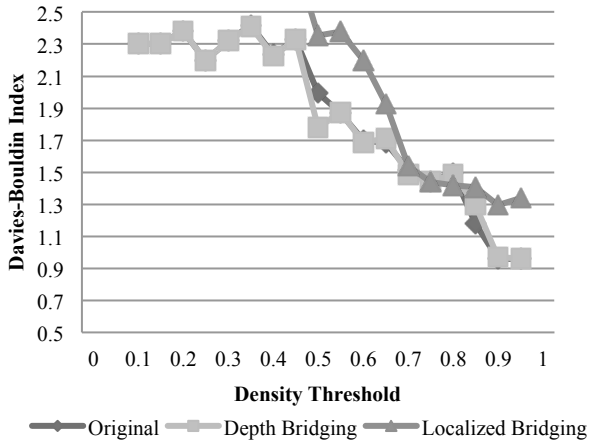


**Figure 20: Comparison of Algorithms for Edge Centrality using Davies-Bouldin Index vs. Density Threshold**

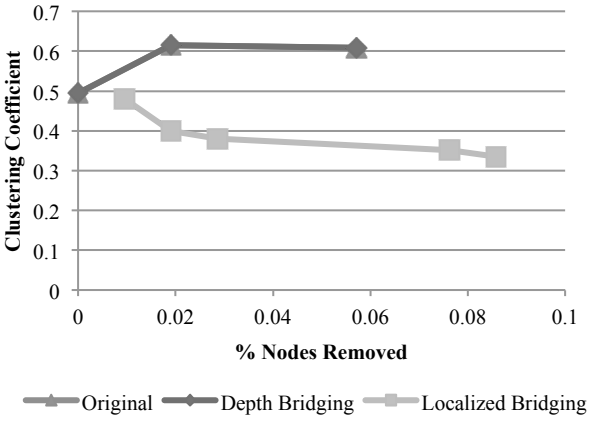


**Figure 21: Comparison of Algorithms for Edge Centrality using Clustering Coefficient vs. Nodes Removed**

On average, for every metric, the depth bridging centrality approach outperformed the original BridgeCut algorithm and the localized bridging centrality approach. We pay careful attention to the clustering coefficient evaluation. For vertex centrality, all the algorithms were on par with one another, with the depth bridging approach slightly ahead. The localized bridging coefficient seemed to be at a disadvantage for edge centrality. Fortunately, depth-bridging centrality performed quite well along with the original algorithm.

# 6. COMPLICATIONS

## 6.1 Efficiency

An obvious drawback of our approach is the necessity for extra computation. The deeper the desired bridging coefficient, the longer the algorithm will take to process. It is difficult to say if this improvement is worth the trade off. Like most computational problems, it is highly dependent on the desired solution: quality or efficiency.

## 6.2 Betweenness Intervention

A major flaw in the concept of a depth bridging coefficient is its intervention into the betweenness domain. As we branch farther away from the directed neighborhood (large depths), we begin to enter a global domain, which is covered by betweenness.

One way to avoid this issue is keep the depth low. A depth of about two or three would probably suffice for most domains, although results may vary depending on the domain.

# 7. CONCLUSION

This report introduced three improvements to the original BridgeCut algorithm presented by Hwang, Kim, and Ramanathan in [1]. Rank tie breaking introduced a heuristic in the rare scenario that two vertices or edges had the same rank. Depth bridging coefficient enabled the algorithm to consider a "local" neighborhood instead of just a direct one. Finally, the re-ranking process demonstrated how to obtain new ranks for vertices and edges using the depth bridging coefficient.

Based on our results, we can conclude that the improvements suggested in this paper are beneficial additions to the original BridgeCut algorithm, especially for vertex centrality. However, we should take into careful consideration the depth of the bridge coefficient. For best results, the depth should not exceed two or three in most domains.

# 8. REFERENCES

[1] Hwang W., Kim T., Ramanathan M., Zhang A. 2008. Bridging Centrality: Graph Mining from Element Level to Group Level. Proc. KDD, 336-344. DOI= http://dl.acm.org/citation.cfm?id=1401934.

[2] Kotz D., Nanda S. 2008. Localized Bridging Centrality for Distributed Network Analysis. Department of Computer Science and Institute for Security Technology Studies, Dartmouth College, Hanover, NH 03755. Dartmouth Computer Science Technical Report TR2008-612 January 30, 2008. DOI= http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.128.7359.

[3] Bonaich P. Power and Centrality: A Family of Measures. American Journal of Sociology, Vol. 92, No. 5. (March 1987), 1170-1182. 1987. DOI= http://www.jstor.org/stable/2780000.

[4] Okamoto K., Chen W., Li X. Ranking of Closeness Centrality for Large-Scale Social Networks. Proceedings of the 2nd annual international workshop on Frontiers in Algorithmics, 186-195. 2008. DOI= http://www.springerlink.com/content/l007881370x0703.

[5] Krebs M. Books about US Politics. 2008. DOI= http://networkdata.ics.uci.edu/data.php?id=8.

[6] Knuth D. Les Miserables. The Stanford GraphBase: A Platform for Combinatorial Computing, Addison-Wesley, Reading, MA. 1993. DOI= http://networkdata.ics.uci.edu/data.php?id=109.

[7] Tan P., Steinbach M., Kumar K. Introduction to Data Mining. Addison-Wesley 2006.