**Mobile Ad Hoc Network Visualization**

**Introduction**

The simulation of mobile ad hoc networks is a complex process that involves multiple layers of communication which must all be monitored. The visualization of these layers can help to provide a clearer understanding of the simulations. However there has been limited work done in providing meaningful visualizations for mobile ad hoc networks. Some works focus on small scale networks, primarily for generating data, and then provide some basic visualization which is neither rich nor meaningful. Other works focus on large networks, but typically only focus on visualizing connectivity information. The other layers of the communication model are left unvisualized.

We also examine the inclusion of digital terrain data into both simulations and visualizations of mobile ad hoc networks. The use of terrain data in simulation is fairly common, and adds an increased level of realism to the simulation. Although common in simulation, little to no work has been done on visualizing terrain data along with mobile ad hoc networks.

**Signal Radiation Patterns**

Our first visualization targets the physical layer of the communications model. Transmitting nodes at this layer send out their signals in a defined pattern. This pattern can be distorted by various environmental effects, such as noise, interference, fading, or shadowing. This visualization provides a visual representation of these radiated signal patterns, including the environment effects as well as the effects of channel state. It also provides a visualization of the regions of acceptable SINR, which is the Signal to Interference plus Noise Ratio. The method of visualizing this information is fairly simple. The signal strength of each transmitter is mapped to the HSL color scheme, shown on this slide. The Hue value defines the color gradient of the patterns, the Saturation value is set to be constant at 1.0, and the lightness value is varied based on the strength of the signal.

This visualization was integrated into a mobile ad hoc network simulator called OMAN. Here we see an example of this visualization from the OMAN simulator. Each transmitter has an aura around it representing its radiated signal pattern. This example shows the difference between a fully known channel state, which has no fading, shadowing, or noise, and a highly unstable channel state which has all three of these. In looking at the visualization, we can see how increasingly difficult it becomes to make resource allocation decisions in the presence of uncertainty.

He we can see a simplified version of the same visualization. The number of transmitters has been reduced, the arena size shrunk, and only the transmitters are shown. Here we can better see how transmitters affect each other’s signal radiation patterns.

This example shows the regions of acceptable SINR visualized on top of the radiation patterns. These regions indicates areas that if a transmitter where to enter the region it would be able to transmit to the node associated with that region. We can again see that decision making in a highly unstable channel state is much more difficult since these regions can not be easily predicted.

The simplified versions of this visualization also show how these regions are affected by neighboring nodes.

**Digital Terrain**

**Conclusion**

**Graph Simplification**

**Introduction**

The motivation for our research is that the graph structure is one used to represent a variety of different data sets, such as computer networks, cellular structures, and research citation trends. The use of graphs to represent these structures is natural since they can clearly and distinctly visualize the desired data along with the interrelationships contained within. The problem that occurs is that these data sets tend to be very large and when visualized they become too complex and cluttered to properly convey the desired information. Even the use of traditional graph layout techniques can still leave a graph too visually complex to understand. Graph simplification provides one method of making these graphs easily comprehensible while still maintaining the actual meaning or structure of the graph.

**Algorithm Overview**

Our work has focused on a two step simplification algorithm that can be applied to large complex graphs. The first step of our algorithm calculates a weight value for each node in the graph based on some given importance metric. We have implemented and evaluated the following importance metrics as part of our algorithm. Each metric used calculates a weight value for the nodes based on some topological property of the graph.

Our first metric is the number of N-ring neighbors. In this metric the weight value for each node is based on the number of neighbor nodes that are exactly N unique hops away from the original node. The value used for N can be specified by the user.

The number of shortest paths passing through a node is our second metric. Here the weight of a node is equal to the number of shortest paths that pass through this node. Each node along the shortest path (u,v) has its weight increased by 1. This is also known as a nodes betweenness.

The third metric we implemented was based on node eccentricity. Eccentricity is the maximum distance within the set of shortest paths from the source node to all other nodes in the graph. The nodes weight then becomes is eccentricity value in the graph.

Our fourth metric is the shortest distance to a center node. In a graph, center nodes is the node or nodes with the lowest eccentricity value. This means the nodes are located with in the center or centers of the graph. Thus the weight for nodes is then equal to the shortest distance to the closest center node in the graph.

Our final metric is the shortest distance to a leaf node. This metric requires that the graph contain at least one leaf node. If the graph meets this requirement, the the nodes weights are made equal to the shortest distance to the closest leaf node.

The second step of our algorithm performs node pruning on the weighted graph. Here nodes and their associated edges are removed. The nodes chosen for removal are those that fall below a user-define weight threshold.. Adjusting the threshold value allows the user to control the resolution and complexity of the simplified graph. The higher the threshold, the more nodes are removed and the courser the graph becomes. In this pruning process there are two options that the user may choose to use. First is to prevent the removal of nodes which would break the connectivity of the graph. Any node whose removal from the graph would break the connectivity has its importance escalated such that it will remain in the graph until it is deemed safe to remove. The second option is the ability to perform iterative pruning. Here the pruning process is applied several times starting with a threshold of zero and iteratively adding onto this threshold until the user defined maximum threshold is reached. This provides a better removal of nodes in that it ensures all lower weighted nodes are removed prior to trying to remove higher weighted nodes which are more important.

**Results**

The basis of this algorithm is in creating good weighting metrics that are able to properly weight a graph. In evaluating our algorithm we have defined three performance criteria to evaluate the effectiveness of our weighting metrics at simplifying any given graph. The first criteria is that a weighting metric should provide a reasonably wide range of weight values for the nodes in the graph. This is important in order for the user to have finer control over the resolution of the simplified graph and be able to see the desired level of data. Our second criteria is that within a given range of weight values, the distribution of nodes among those weights should be relatively even. A distribution biased in some region creates a sharp drop off of nodes at certain threshold values. This again impacts how much control the user has over the resolution of the final graph. The third and final criteria is that the metric should be capable of producing a meaningful visual representation of the original graph. Simplifying a graph should reveal something about the graphs connectivity or underlying structure.

Given these criteria for good performance, we applied our algorithm to two kinds of large complex graphs. The first graph we chose was one generated by the Inet topology generator, a power-law-based graph generator that generates graphs representing internet topologies. The graph we used has 4500 nodes and 15308 edges. The second graph we chose was generated from citation research trend data provided to us by Dr. Chaomei Chen of Drexel University's iSchool. This graph had 1025 nodes and 15430 edges. Prior to simplifying each graph with our algorithm, the graphs were pre-processed using the Neato layout tool from Graphviz.

**N-ring Neighbors Results**

The first metric we tested was the N-ring neighbor metric. Here we tested 3 different values for N, 1, 2 and 3. This graph show how nodes drop out of the graph over a fixed number of simplification steps that span the entire range of weight values. We can see that N-ring neighbors tends to produce a steady drop off of nodes meaning it has a more even distribution of weights over the range, thus satisfying the second criteria.

Here we can now see the results on 2-Ring neighbors applied to the Inet graph. We can see 4 different simplification steps here along with the corresponding threshold value at which the graph was pruned. As you can see the range of weight values is very wide for this metric satisfying the first criteria. We can also see that the visual structure of the graph does become simpler as the threshold increases ultimately resulting in a core group of nodes being exposed that were previously hidden. This satisfies the third criteria. Overall all three criteria are properly satisfied, meaning that this is a relatively good metric.

**Number of Shortest Paths Results**

Our next metric, the number of shortest paths metric, tells of a different story. On the node drop off graph we can see a sharp drop off when pruning the lower weighted nodes from the graph. This tells us that the lower end of the weight range contains a large number of the nodes in the graph with the middle and high end of the range having an even distribution of the nodes left. This unbalanced distribution does not satisfy the second criteria.

We can see here that the range of weight values is quite large meaning that this metric satisfies our first criteria of good performance. We can again see a core structure emerge from the Inet graph as the simplification progresses which again satisfies our third criteria. This metric overall satisfies two of the three criteria making it a mediocre metric.

**Eccentricity Results**

For the eccentricity metric we can again see a steep drop off of nodes during pruning. Based on the S shape of this graph we can see that the lower and higher weighted nodes are fewer but more or less evenly distributed. The middle range of weights looks to be very heavy again resulting in a quick drop off of nodes.

For this metric we look at the results as applied to the citation graph data. We can see that the range of weight values for this metric is very different than from the previous metric. This range is much smaller in scale than previous metrics. Despite this the visual structure we see emerging is a sort of backbone structure of the graph. Overall this metric only satisfies one of the three performance criteria which makes this one of the weaker metrics.

**Shortest Distance to a Center Node Results**

For the shortest distance to a center node metric we see a trend similar to the eccentricity metric appearing. The drop off graph is again in the S shape, leading us to the same conclusions as before.

The range of weights is again very small compared as it was with the eccentricity metric. This is not surprising since this metric uses the eccentricity metric as its basis. The graph structure that emerges from simplifying with this metric does produce something different though. We can see more of a core group of center nodes emerge instead of a backbone as we saw with eccentricity. Just as with the eccentricity metric, this metric only satisfies one of the performance criteria making it a weak metric as well.

**Shortest Distance to a Leaf Node Results**

Our final metric, the shortest distance to a leaf node, produces results similar to the previous two metrics. We can see that the Inet and citation graphs now produce different drop off rates. This comes from the fact that this metric is heavily influenced by initial structure of the graph. With the Inet graph we can see that a lot of nodes are located very close to a leaf node since so many drop out in the lower and middle weight ranges. The S shape of the citation graphs drop off graph tells us that there are likely fewer leaf nodes meaning most nodes must travel a medium distance to get to the closest leaf node.

As with the previous two metrics, the range of weights is quite small. The visual structure that is revealed to us is again a backbone of nodes who have the shortest distances to the original leaf nodes within the graph. This metric also only satisfies one of the performance criteria making it a fairly weak metric.

**Conclusion**

In conclusion we have presented a simplification algorithm that may be applied to a complex graph in order to produce a controlled thinning of the graph. This simplification of the graph provides an approach to visualizing the fundamental structure of the graph by displaying the most important nodes, where the importance may be based on the topology of the graph or external factors. We have shown several weighting metrics, evaluated their performance on different graphs, and shown the graphs produced by applying them. Our future work includes defining additional importance metrics based on varying properties of graphs as well as testing and evaluating our approach on other types of graphs.