EG UK Theory and Practice of Computer Graphics (2010)
John Collomosse, Ian Grimstead (Editors)

An Evaluation of the use of Clustering Coefficient as a Heuristic for the Visualisation of Small World Graphs

Fintan McGee [1] and John Dingliana[1]

[1]GV2, School of Computer Science and Statistics, Trinity College Dublin

Abstract

Many graphs modelling real-world systems are characterised by a high edge density and the small world properties of a low diameter and a high clustering coefficient. In the "small world" class of graphs, the connectivity of nodes follows a power-law distribution with some nodes of high degree acting as hubs. While current layout algorithms are capable of displaying two dimensional node-link visualisations of large data sets, the results for dense small world graphs can be aesthetically unpleasant and difficult to read. In order to make the graph more understandable, we suggest dividing it into clusters built around nodes of interest to the user. This paper describes a graph clustering using the average clustering coefficient as a heuristic for determining which node a vertex should be assigned to. We propose that the use of clustering coefficient as a heuristic aids in the formation of high quality clusters that consist of nodes that are conceptually related to each other. We evaluate the impact of using the clustering coefficient heuristic against other approaches. Once the clustering is performed we lay out the graph using a force directed approach for each clustering individually.

Categories and Subject Descriptors (according to ACM CCS): G.2.2 [Graph Theory]: Graph AlgorithmsI.5.3 [Clustering]: AlgorithmsF.2.2 [Nonnumerical Algorithms and Problems]: Routing and layout

1. Introduction

Many real-world networks across different fields have similar characteristics and can be classified as small world graphs [WS98, CF09, ACJM03, vHW08]. Small world networks are characterised by two properties. The first is the average of the shortest path between each pair of vertices for the entire graph. The second property is the average local clustering coefficient of the graph, which is defined as the average of the clustering coefficients for each vertex. The *clustering coefficient* for a vertex is defined as the ratio of edges connecting the neighbours of a vertex to the number of possible edges between neighbours of the vertex. The clustering coefficient c for a vertex in a directed graph is given by

$$c = \frac{|E|}{K(K-1)}$$

where E is the set of edges connecting neighbours of the vertex (both inbound and outbound edges) and K is the neighbourhood size of the vertex, i.e. the number of vertices the target vertex is connected to. From the above it can be seen that a vertex needs at least two neighbours to have a valid clustering coefficient value. To determine if a graph can be considered a small world graph, it is compared to a randomly generated graph with the same number of vertices and edges. A small world graph has approximately the same average path length, but a considerably higher (by orders of magnitude) clustering coefficient.

1.1. Motivation

Our motivation is to make graphs more comprehensible. We are looking at small world graphs specifically due to the presence of groups of highly connected nodes and the strong relationship they have with clustering coefficients. The purpose of our clustering is to allow a user to divide the graph into more manageable sets of conceptually related nodes. Our ultimate goal is enable a user to generate different perspectives of the data by selecting different sets of nodes as super-nodes. Clustering also has the benefit of reducing the

© The Eurographics Association 2010.

DOI: 10.2312/LocalChapterEvents/TPCG/TPCG10/167-174

delivered by
EUROGRAPHICS DIGITAL LIBRARY
www.eg.org diglib.eg.org

computation required for a force directed layout, due to the fact that nodes are only laid out relative to other nodes within their cluster.

2. Related work

Milgram [Mil67] first described small world graphs in his work focused on social networks. The concept was more recently revived by Watts and Strogatz [WS98] and has been shown to hold true for a variety of networks, such as the relationships between actors and films [ACJM03] as well as computer systems [CF09], models of biological networks [WS98] and citation networks [vH04].

2.1. Clustering

Eades and Feng [EF97] describe a clustered graph as a graph with recursive clustering structure over the vertices. In their example the clustering structure is an attribute of the graphs and vertices, however in many cases if a graph is to be clustered there may be no intrinsic attribute or parameter which describes the clustering hierarchy. There are many different approaches to generating an optimum clustering as it is a difficult problem that is NP-complete [NG04].

There are two main approaches to graph clustering (or partitioning as it is often referred to), geometric and non geometric. The aim of *geometric clustering* is to have vertices that are geometrically close to each other share a cluster and vertices that are distant from each other appear in separate clusters. An example of such a clustering is given by Quigley and Eades' FADE algorithm [QE01] in which a quad tree is used alongside a modified force directed algorithm. There are many different methods of non-geometric clustering. Some methods such as Markov Clustering (MCL) [vD00] and spectral partitioning [FT07] use an algebraic approach, working on a mathematical representation of the graph. Other methods such as edge betweenness centrality clustering [NG04] use a graph based approach, calculating graph theory characteristics of vertices or edges that are then used to partition the graph into clusters. An in-depth review of clustering methods is available from [Sch07].

2.2. Layout

Force directed layout algorithms are one of the most common approaches to graph layout. These algorithms use a physically based model to lay out nodes within the graph space. The algorithms work iteratively, repositioning nodes based on forces until the energy within the system is minimal. Early examples include Eade's spring embedder [Ead84], Kamada and Kawai [KK89] and Fruchterman and Reingold's algorithm [FR91]. More recent force directed algorithms [HK01, GK01, HJ05] aim at drawing larger graphs and use a multilevel approach. Multi-level approaches generate coarse versions of graphs and use these for the layout of more refined versions. Force directed algorithms are often highly parallelisable and Frishman and Tal [FT07] have demonstrated the benefits of using the parallel processing capabilities of the GPU to aid in graph layout. Further graph layout approaches include algebraic algorithms [HK02] and topology based [AMA07] algorithms. Algebraic algorithms apply algebraic operations to a matrix form of the graphs, such as the adjacency matrix. Topological algorithms consider the graph structure for features such as cliques or trees to aid in layout.

3. Proposed approach

3.1. Cluster Building

To determine which cluster a node conceptually belongs to, we use the average clustering coefficient of a cluster as a metric. Each cluster initially only contains a super-node, selected by the user. For each super-node a single neighbour is added as the calculation of a clustering coefficient requires a node to have more than one neighbour. The node added to the cluster is the neighbour of the super-node, with the largest neighbourhood size. This results in each cluster containing two connected nodes. The set of all nodes of the graph that have a neighbourhood size larger than one and have not already been assigned to clusters is then stored in an ordered node list, built by traversing the graph from each of the super-nodes using a breadth first search. Nodes are stored primarily in order of their increasing graph distances from a super-node and secondarily by the size of the node's neighbourhood from largest to smallest. For the purposes of clustering, edge direction is ignored in the breadth first search, as nodes can still be conceptually related regardless of the direction of the edge between them. The clustering algorithm is not impacted by whether the graph is directed or unidrected.

Nodes that are connected to only one other node, are not added to the list, as a node which only has one neighbour is guaranteed to have a negative impact on the local clustering coefficient value of a cluster and can only ever be added to the cluster that it is connected to. Therefore these nodes are added to the cluster that their only neighbour is assigned to once all other nodes are assigned.

The motivation for ordering the list secondarily by neighbourhood size is to allow nodes of a lower neighbourhood size to be added to a node where as many as possible of their neighbours have already been added. If nodes of a large neighbourhood size already are processed, then any future node added is more likely to find several of its neighbour nodes already assigned to a cluster. Furthermore, this ordering results in more balanced cluster sizes, as it will prevent the clusters which are initially based on more highly connected nodes from taking all the nodes with a small neighbourhood size.

In the next stage of the algorithm, the ordered list is iterated through adding each node temporarily to a cluster. The average clustering coefficient of the cluster is then recalculated to determine the impact of addition of the node. The node is then permanently assigned to the cluster which has the highest resulting coefficient.

Finally, once all other nodes have been assigned to a cluster, the single neighbour nodes are then assigned to the cluster of their neighbour. In summary:

1. Specify the super-nodes used as a basis for clusters.
2. Add one neighbour node to each super-node to form a basic cluster.
3. Build a list of remaining nodes, sorted by distance from a super-node and node neighbourhood size.
4. Add each node to the cluster that has the highest clustering coefficient if the new node is included.
5. Assign the single neighbour nodes to the clusters of their neighbours.

The calculation of the clustering coefficient for a large set of nodes is a time consuming task, however the clustering coefficient for each node in a cluster can be calculated in parallel. Therefore we compute the new clustering coefficient of each node within a cluster on the GPU (The described algorithm 1 executes in parallel on multiple nodes). The result is averaged by the CPU to determine the average clustering coefficient for the nodes within a cluster.

Algorithm 1 GPU Shader Program algorithm for calculating the clustering coefficient of a node within a cluster C.

```
$v \in V_c$
**for all** $u \in V_c$ **do**
  **if** $\{u,v\} \in E_c$ **then**
    $v.neighbourhoodSize := v.neighbourhoodSize + 1$
    **for all** $w \in V_c$ **do**
      **if** $u \neq w \wedge \{u,w\} \in E_c \wedge (\{w,v\} \in E_c \vee \{v,w\} \in E_c)$ **then**
        $v.neighbourhoodEdgeCount := v.neighbourhoodEdgeCount + 1$
        {As w is also a neighbour of v}
      **end if**
    **end for**
  **end if**
**end for**
**if** $v.neighbourhoodSize > 1$ **then**
  $v.clusteringCoefficient := v.neighbourhoodEdgeCount / v.neighbourhoodSize * (v.neighbourhoodSize - 1)$
**else**
  $v.clusteringCoefficient = 0$
**end if**
```

In order to process our graph data on the GPU we load the graph data into textures, which are then uploaded onto the graphics card for processing.The input textures required for the clustering store the nodes within the cluster, and the edge map for the graph. The calculations are done using a fragment shader written using GLSL and the results are passed back to the CPU using a frame-buffer object. GLSL is chosen as it is relatively straight-forward to process the graph adjacency matrix as a texture.

3.2. Layout

To lay out the nodes contained in each of the individual clusters we have implemented Fruchterman and Reingold's [FR91] force directed algorithm on the GPU. The clusters themselves are distributed evenly across the graph spaces by applying this algorithm to the set of super-nodes and treating this set as a fully connected graph. Once the super-nodes are in place, their position is locked, and during layout of the clusters, the distance between a node and its super node is constrained. Our clustering approach improves layout performances for large graphs. It results in fewer computations of forces between nodes, as forces are only calculated for nodes within the same cluster.To reduce the clutter resulting oorm edges between clusters we draw inter-cluster edges as bundled splines in an approach similar to Holten [Hol06].

4. Evaluation

4.1. Evaluation approach

In order to evaluate the effectiveness of the clustering we compare our algorithm to variations where cluster coefficient impact was not taken into account. For the *round robin* clustering nodes are initially sorted in the same manner as before, but assigned to each cluster in a sequential fashion. Nodes are only assigned to clusters which they are connected to. A more thoroughly random approach was also taken, by assigning nodes to a cluster chosen entirely at random, from a list of all clusters that neighbours of the node have already been assigned to. Furthermore we have also analysed using the change (delta) of the clustering coefficient and assigning the node to the least negatively impacted cluster, instead of assigning the node to the cluster with the highest clustering coefficient.

4.2. Evaluation metrics

Newman and Girvan [NG04] define a measure of the quality of a division of a network graph, referred to as modularity. This metric is based upon the number of edges that start and end in the same cluster (referred to as communities in Newman and Girvan's paper). The modularity, Q, is calculated as

$$Q = \sum_i (e_i i - a_i^2)$$

Where $e_i i$ is the fraction of all edges that start and end in cluster i and a_i is the fraction of all edges that terminate in cluster i. Modularity depends solely on the relationships between nodes.

Where contextual meta-data about the node is available we use this to determine if the node conceptually fits in with the cluster it is assigned to.

4.3. Evaluation Graphs

We evaluate our algorithm using a wide variety of graphs. For an evaluation using real world data, we have generated a set of four graphs based on connectivity between Wikipedia articles. We evaluate these graphs using Newman and Girvan's modularity metric.

We also randomly generate small world graphs which are clustered using our approach, as well as the round robin and random approaches. We use Watts and Strogatz's approach for creating small world graphs [WS98]. This approach begins by creating a lattice like structure with edges uniformly across vertices. Each edge is rewired to a random target vertex with a probability P. For a low value of P the resulting graph exhibits small world properties, as P approaches 1 the graph becomes more like a completely randomly connected graph. At P equal to one, the graph can be considered a completely randomly generated graph. These randomly generated graphs contain 60 nodes which vary in edge density. They are evaluated based on the modularity of the resulting clustering.

In order to analyse how effective the use of the average local clustering coefficient of a cluster is in building conceptually related clusters, we have created an artificial social network data set modelling activities at a sports club. Our model contains 100 nodes each representing a member of the club. Each member is assigned a level of interest in six activities, between 0 and 1.0. The sum of a member's interest across all activities is equal to 1. In order to generate the graph we calculate the euclidean distance between each member's 6 levels of interest. If the euclidean distance is less than a threshold value of 0.5 we assume that due to the common amount of activities the members are socially connected. Therefore we add an edge to the graph connecting the nodes representing the members. The resulting graph contains 803 edges. Using this graph to evaluate our clustering we can see exactly the ratings for each node for each activity and hence determine if they have been placed in a conceptually correct cluster. The super-node selected for each cluster is a person who undertakes only one activity with the maximum level of interest. This means that each cluster member should have some level of interest in the activity of the super-node.

Finally we use our algorithm to cluster and layout a real-world social network data set. We chose the genealogy of influence data set [Lov10],which contains prominent figures in the field of art, science and entertainment and relates them using "influenced by" relationship. The generated graph contains 1929 nodes and 4364 edges.

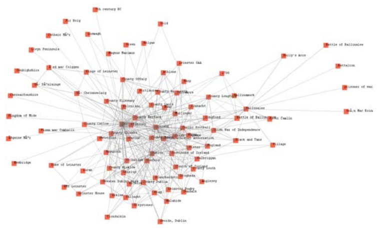
Figure 1: *Wikipedia graph with 91 vertices and 567 edges laid out using a simple force directed algorithm.*

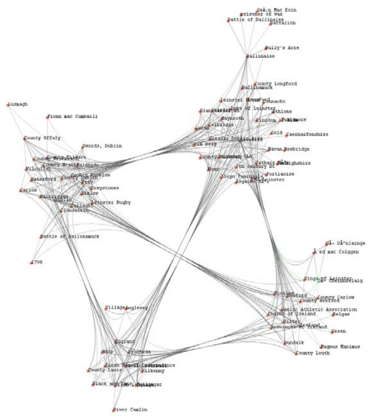
Figure 2: *Graph from Figure 1 using our approach.*

5. Results

5.1. Wikipedia Data Set

The clustering coefficient based, round robin and random algorithms were each run on the Wikipedia test graphs, where the four nodes with the highest degree were selected for clustering. The random clustering was run three times for each graph and averaged, as it resulted in a different clustering each time. The resulting modularity of each graph is displayed in Table 1.The use of clustering coefficient as a metric produces a significantly higher level of modularity than a round robin or random assignment of nodes to clusters.

$\|V\|$	$\|E\|$	Clustering Coefficient	Round Robin	Random Average
91	567	0.1279	0.049	0.0561
358	3729	0.0931	0.038	0.0424
506	3962	0.1545	0.0692	0.0645
1000	28534	0.0251	0.0038	0.005

Table 1: *Modularity values for Wikipedia based graph using different approaches to clustering.*

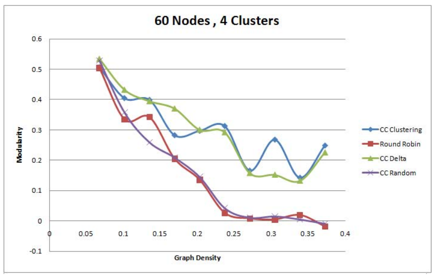

Figure 3: *The modularity of graphs containing 60 nodes and increasing in density, using the described clustering approaches for building 4 clusters.*

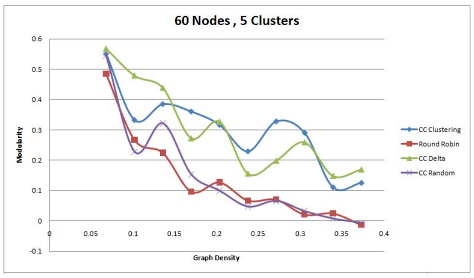

Figure 4: *Modularity for building 5 clusters.*

5.2. Randomly Generated Small World Graphs

For the randomly generated small world graphs, we compare the change in modularity over graphs of various sizes and increasing density of edges relative to nodes with four different approaches. We use the clustering coefficient approach, round robin assignment and a random assignment as well as the delta of the clustering coefficient in determining the assignment of nodes to clusters. In each case, the use of clustering coefficient as a heuristic showed an improvement. Figure 3 shows the improvement in using clustering coefficient over the round robin and random method for creating 4 clusters for a graph with 60 nodes and edge density increasing from 120 to 660 edges. On average use of the maximum average clustering as opposed the the delta of the average clustering coefficient resulted in a more modular clustering but the difference is not as distinct as with the other approaches. Similar results can be seen for forming 5 or 6 clusters (see Figures 4 and 5).

We also analysed the effect of altering clustering coefficient for a graph of a given size and density. We generated 20 graphs of the same size, with an increasing probability of rewiring , resulting in a set of graphs with a decreasing clustering coefficient but the same number of nodes and edges. As can be seen from figure 6 the impact of changing the clustering coefficient is not consistent, but use of the absolute clustering coefficient still improves modularity even as the graph becomes less small world like. The fluctuation in

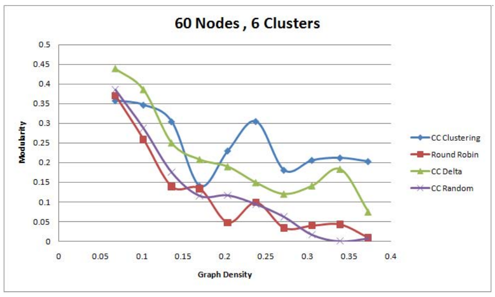

Figure 5: *Modularity for building 6 clusters.*

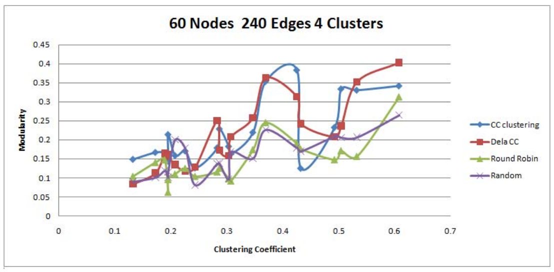

Figure 6: *The modularity of graphs containing 60 nodes and 240 edges increasing in clustering coefficient, when clustered using 4 super-nodes.*

modularity is a result of the randomness in generating the graphs and the most well connected nodes are automatically selected as super-nodes, which can change significantly for each random graph. It can can be seen in Figure 7 that a similar result was achieved for 5 clusters.

5.3. Sports Club Data Set

We generated 6 clusters for the sports club data set and compared the use of the clustering coefficient approach to the round robin approach. For each cluster we calculated the sum of each activity for each person in the group and normalised this value by dividing it by the number of people

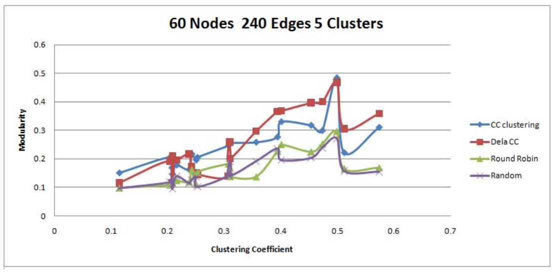

Figure 7: *The modularity of graphs increasing in clustering coefficient, when clustered using 5 super-nodes.*

© The Eurographics Association 2010.

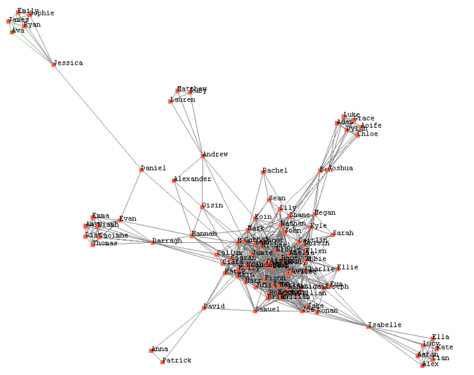

Figure 8: *Sports club graph with 100 vertices and 803 edges laid out using a simple force directed algorithm.*

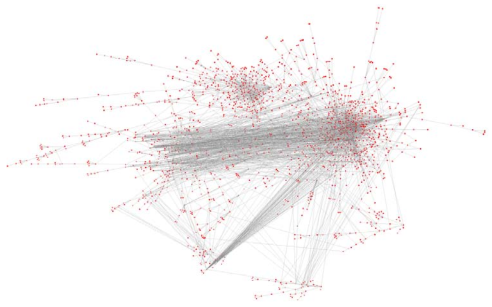

Figure 10: *Genealogy of Influence graph with 100 vertices and 803 edges laid out using a simple force directed algorithm*

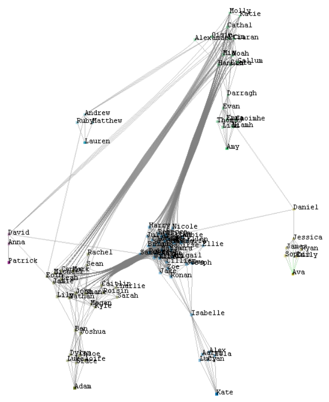

Figure 9: *Graph from figure 8 using our approach.*

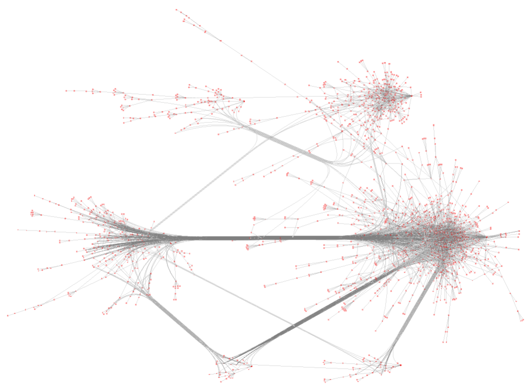

Figure 11: *Graph from figure 10 using our approach.*

in the group. In Table 2 we display the scores for the specific activity associated with the cluster. It can be seen that for every activity, apart from tennis, use of clustering coefficient results in a better score for the cluster. The lower value for tennis using the round robin approach results from a larger number of members being interested in tennis, but not as their primary activity. Examining the number of members assigned to clusters that they have no interest in reveals a clear result. Using the clustering coefficient approach, this only happened for four members. Using the round robin approach 17 members are assigned to clusters where the super-node represents a sport that they have no interest in.

	Clustering Coefficient	Round Robin
Soccer	0.9	0.9
Swimming	0.6038	0.5877
Tennis	0.4208	0.5563
Gym	0.565	0.441304
Running	0.9	0.4088
Cycling	0.925	0.010265

Table 2: *Level of interest in the sport of the super-node by cluster.*

5.4. Influence Data Set

For our analysis of the influence data set each node representing a person has been labelled with a primary profession (as also done by van Ham and Wattenberg [vHW08]). Each node has been classified by the person's primary profession, from each of the following professions: Actor (including comedians, 321 nodes), artist (209 nodes), mathematician (170 nodes), musician (159 nodes), philosopher (586 nodes) and scientist (484 nodes). The previous described clustering and layout approach has been applied to this data set, with the super nodes being manually selected as one prominent individual from each set. Respectively these individuals were George Carlin, Vincent van Gogh, Carl Gustav Jakob Jacobi, Ludvig Van Beethoven, Friedrich Nietzsche, and Albert Einstein. A completely accurate assignment of nodes to clusters based on profession is not expected as there

	Clustering Coefficient		Round Robin		Delta C.C.	
	Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
Actor	286	37	282	41	286	37
Artist	54	154	104	104	59	149
Mathematician	23	147	60	110	47	123
Musician	44	115	68	91	27	132
Philosopher	520	66	183	403	393	191
Scientist	133	350	151	331	40	443
Total	1060	869	848	1081	854	1075
Total %	54.9508	45.0493	43.9606	56.034	44.27164	55.72836

Table 3: *Correct assignments of nodes by profession to super-nodes using each approach.*

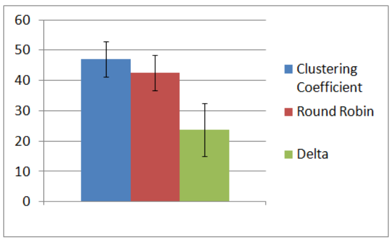

Figure 12: *Chart indicating the number of correctly clustered nodes averaged over 20 different input sets. The error bars indicate the standard deviation.*

are many other characteristics which affect relationships between nodes, such as indistinct boundaries between professions (particularly for mathematicians, philosophers and scientists) and also the era in which the person chosen as a super-node lived impacts relationships and influence.The selected super-nodes were chosen as people who were strong examples of each field, the nodes representing them were well connected, and their professions were more clearly defined. However, choice of super-node does have a large impact on the data set. For example if a more contemporary musician such as Miles Davis is marked as a super node, many of the classical musicians end up being associated with the Philosophers cluster.

The breakdown of the correctness of resulting clustering categorised by profession can be seen in Table 3. We performed a comparison using the round robin approach and clustering coefficient delta approaches of assigning nodes to clusters. It can be seen from the above tables that using clustering coefficient as a heuristic resulted in nodes being assigned to clusters that they were more conceptually related to in terms of profession. It is also clear that assigning nodes to the cluster with the largest resulting average clustering coefficient categorized the nodes more effectively than using the average local clustering coefficient delta or round robin approach. To verify this we also ran our algorithm over 20 different selection sets built using popular nodes for each profession. On average the clustering coefficient approach was correct 46.9% of the time, the round robin approach 42.5% of the time and the average cluster coefficient delta 23.9% of the time (see figure 12). The delta approach fared poorly as there were many cases where a disproportionately large number of nodes ended up in a small number of clusters. The clustering coefficient approach outperformed the round robin approach in 17 out of the 20 cases. In the worst of these 3 cases the round robin approach scored 1.71% higher than the clustering coefficient approach. In the best case overall the clustering coefficient approach scored 11.15% higher than the round robin approach.

5.5. Conclusions

We have presented a novel clustering algorithm to aid in the visualisation of small worlds graphs and have applied it to various different graphs. By taking into account the impact of the addition of a node on the local clustering coefficient of a cluster, we achieve a more modular layout of nodes. In our analysis of randomly generated graphs there was only a slight benefit to using the delta of the coefficient as a heuristic, but it can be seen in our influence graph example that assigning the node to the maximum clustering coefficient is more effective.

We have also shown, using the artificial sports club data set and the genealogy of influence data set that using clustering coefficient as a heuristic result in clusters that are more conceptually alike.

5.6. Future Work

We intend to perform comparisons with cluster building methods such as those by Newman and Girvan [NG04], which use edge betweenness centrality as a metric to determine how to build clusters within a graph based purely on the underlying structure and not on specifying a node to build the cluster around. It may be useful to see if edge betweenness centrality can be used in deciding whether or not a node should be added to a cluster built around a target super-node. Determining an automatic selection of super-nodes that is more robust than selecting nodes of large neighbourhood size for cases where the user is not sure would also be of benefit and may also allow multiple levels to be added to the layout algorithm.It may also be possible to adopt the clustering specified here as a node group selection technique, without clustering the entire graph, which may be of benefit for graph analysis tasks.

We use edge bundling to reduce the visual cluster of the graph, however there is still some interference, between the inter-cluster edges and the edges which start and end within the one cluster. Further work is needed to reduce this, perhaps taking into account the inter-cluster edges during the force directed layout, or using a more perceptual approach such as specific colouring or translucency of the longer inter-cluster edges, in a manner similar to Holten [Hol06]. Some perceptual research similar to the work done by Ware and Purchase [WPCM02] is also required on how to best render the inter-cluster edges as bundles, while allowing paths to be traced between nodes easily. Another option worth investigating would be the impact of the clustering functionality on laying out a graph in three dimensional space. Previous work by Ware and Mitchell [WM08] has shown that three dimensional graphs (using stereoscopic displays), can improve user performance.

This work has been made possible by the kind sponsorhip of the IRCSET (the Irish Research Council for Science, Engineering & Technology) and Intel.

References

[ACJM03] Auber D., Chiricota Y., Jourdan F., Melancon G.: Multiscale visualization of small world networks. *Information Visualization, 2003. INFOVIS 2003. IEEE Symposium on* (2003), 75–81. 1, 2

[AMA07] Archambault D., Munzner T., Auber D.: Topolayout: Multilevel graph layout by topological features. *Visualization and Computer Graphics, IEEE Transactions on 13*, 2 (march-april 2007), 305 –317. 2

[CF09] Cai-Feng D.: High clustering coefficient of computer networks. In *Information Engineering, 2009. ICIE '09. WASE International Conference on* (2009), vol. 1, pp. 371–374. 1, 2

[Ead84] Eades P.: A heuristic for graph drawing. *Congressus Numerantium 42* (1984), 149–160. 2

[EF97] Eades P., Feng Q.-W.: Multilevel visualization of clustered graphs. In *Graph Drawing*. 1997, pp. 101–112. 2

[FR91] Fruchterman T. M., Reingold E. M.: Graph drawing by force-directed placement. *Software Practive and Experience 21*, 11 (1991), 1129–1164. 2, 3

[FT07] Frishman Y., Tal A.: Multi-level graph layout on the gpu. *Visualization and Computer Graphics, IEEE Transactions on 13*, 6 (2007), 1310–1319. 2

[GK01] Gajer P., Kobourov S.: Grip: Graph drawing with intelligent placement. *LNCS Graph Drawing 2000 1984* (2001), 222–238. 2

[HJ05] Hachul S., Junger M.: An experimental comparison of fast algorithms for drawing general large graphs. *LNCS Graph Drawing 2005 3843* (2005), 235–250. 2

[HK01] Harel D., Koren Y.: A fast multi-scale method for drawing large graphs. *LNCS Graph Drawing 2000* (2001), 183–196. 2

[HK02] Harel D., Koren Y.: Graph drawing by high dimensional embedding. *LNCS Graph Drawing 2002 2528* (2002), 207–219. 2

[Hol06] Holten D.: Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *Visualization and Computer Graphics, IEEE Transactions on 12*, 5 (sept.-oct. 2006), 741 –748. 3, 8

[KK89] Kamada T., Kawai S.: An algorithm for drawing general undirected graphs. *Inf. Process. Lett. 31*, 1 (1989), 7–15. 2

[Lov10] Love M.: Genealogy of influence. `http://www.freebase.com/view/influence`, 2010. 4

[Mil67] Milgram S.: The small world problem. *Psychology Today 2* (1967), 60–67. 2

[NG04] Newman M. E. J., Girvan M.: Finding and evaluating community structure in networks. *Physical Review E 69*, 2 (2004), 026113. 2, 3, 7

[QE01] Quigley A., Eades P.: Fade: Graph drawing, clustering, and visual abstraction. In *Graph Drawing*. 2001, pp. 77–80. 2

[Sch07] Schaeffer S. E.: Graph clustering. *Computer Science Review 1*, 1 (2007), 27 – 64. 2

[vD00] van Dongen S. M.: *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, The Netherlands, 2000. 2

[vH04] van Ham F.: Case study: Visualizing visualization. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on* (2004), pp. r5–r5. 2

[vHW08] van Ham F., Wattenberg M.: Centrality based visualization of small world graphs. In *Eurographics/ IEEE-VGTC Symposium on Visualization 2008* (Eindhoven, Holland, 2008). 1, 6

[WM08] Ware C., Mitchell P.: Visualizing graphs in three dimensions. *ACM Trans. Appl. Percept. 5*, 1 (2008), 1–15. 1279642. 8

[WPCM02] Ware C., Purchase H., Colpoys L., McGill M.: Cognitive measurements of graph aesthetics. *Information Visualization 1*, 2 (2002), 103–110. 942167. 8

[WS98] Watts D., Strogatz S.: Collective dynamics of "small-world" networks. *Nature*, 393 (1998), 440–442. 1, 2, 4