

Characterization of graphs using degree cores

John Healy¹, Jeannette Janssen², Evangelos Milios¹, William Aiello³

¹ Faculty of Computer Science,
Dalhousie University, Halifax, Canada,
<http://users.cs.dal.ca/~{healy,eem}/>

² Dept. of Mathematics and Statistics,
Dalhousie University, Halifax, Canada,

<http://www.mscs.dal.ca/~janssen/>

³ Department of Computer Science,
University of British Columbia, BC, Canada

Abstract. Generative models are often used in modeling real world graphs such as the Web graph in order to better understand the processes through which these graphs are formed. In order to determine if a graph might have been generated by a given model one must compare the features of that graph with those generated by the model. We introduce the concept of a hierarchical degree core tree as a novel way of summarizing the structure of massive graphs. Hierarchical degree core trees are representations of the subgraph relationship between the components of the degree core of the graph, ranging over all possible values of k . From these trees we extract features related to the graph's local structure from these hierarchical trees. Using these features, we compare four real world graphs (a web graph, a patent citation graph, a co-authorship graph and an email graph) against a number of generative models. All the graphs, with the exception of the email graph, show markedly different features from our generative models. Conversely, the email graph appears to have similar features to a number of our generative models, particularly to the partial duplication model of Chung and Lu.

1 Introduction

The primary motivation behind this paper is to compare real-life graphs such as the Web graph with various models to determine the best fit. There are many reasons for studying the structure of the Web. The most predominant of these is improving our ability to search the Web. Other research occurs in sociological analysis of communities represented by the Web graph. The ability to model the formative process underlying a real-life graph provides useful insight into its structure.

There are several methods available for comparing two graphs to determine their similarity. The most common of these approaches is a comparison of the degree distribution of two graphs. Other descriptive statistics include the distribution of clustering coefficients, the frequency of occurrence of isomorphic copies of various subgraphs [1] and the diameter. We focus on using the degree core

decomposition of a graph to extract features which can be used for summarizing a graph and performing model validation.

A k -core of a graph is a maximal induced subgraph of minimum degree k . As we are using degree to induce these subgraphs we will refer to them as degree cores. It is straightforward to show [2] that the degree core is unique for a given core and a given k , and can be obtained by recursively removing all nodes with degree at most k . The degree cores of a graph can consist of multiple components. For our purposes, we generate every non-empty degree core of a graph. The components of the degree cores for all values of k form a hierarchy where two components have a parent-child relationship, when the child component is a subgraph of the parent component. We will refer to the tree thus generated as a *hierarchical degree core tree*.

To model a real world graph, we compare its hierarchical degree core tree structure to those of several generative models. More specifically, we examine the distribution of the number of components in each degree core decomposition. In our experiments we try to model four real world graphs in this manner.

2 K-cores

Cores were first introduced by Seidman [3] and popularized by Wasserman and Faust[4]. Batagelj and Zaversnik [2] generalize Seidman’s work beyond simple degree to include any monotone function p . Examples of such functions range from the degree (in-degree, out-degree, directionless degree) of a vertex to the number of cycles of length k passing through a vertex. For the purposes of this paper we will use degree cores dealing with vertex degree in a similar context as Seidman used his original cores.

A k -core is the subgraph generated by recursively removing all nodes with degree smaller than k from a graph. As we are using degree to induce these subgraphs we will refer to them as degree cores. The difference between this and simply filtering out all vertices with degree $< k$ is best illustrated by comparing their effects on a simple tree. In the case of a tree, the filtering of all degree-one nodes results in the pruning of all of a tree’s leaves, whereas the degree core with $k = 2$ would prune back the leaves of a tree at each recursion, thus destroying the tree completely.

More formally, let $G=(V,E)$ be a simple graph. A degree core is defined as:

Definition 1. *A subgraph H of a graph $G = (V, E)$ induced by the set $C \subseteq V$ is a k -core or a degree core of order k iff $\forall v \in C \text{ deg}_H(v) \geq k$ and H is a maximum subgraph with this property.*

Batagelj and Zaversnik [2] go on to define the core number of a vertex to be the highest order of a core that contains this vertex.

The current literature indicates that degree cores have been used for variety of purposes. Most research implementing degree cores involved using them as a tool to filter the data. Our research differs from the majority of previous work in that we are interested in degree cores for their own sake, for the the insight they

give into the structure of our data. Alvarez-Hamelin *et al.* [5, 6] have examined the structures revealed by degree cores. Their research included an analysis of features of these degree cores across internet graphs [6] as well as the development of a visualization tool which makes use of the core numbers of a graph’s vertices [5].

Some areas where degree cores have been employed in the past include:

- 1) Visualization: Visualization research has been done to examine the degree core structure of both real world graphs and generative models [5–7]. The main use of degree cores in the field of visualization is the filtering ‘unimportant’ nodes, referring to nodes which have a low core number. A node’s degree core number has also been experimentally shown to correspond to many notions of centrality within a graph. This allows a node’s core number to be used as a computationally inexpensive approximation of its centrality. As with most visualization techniques, the visualization research using degree cores has limited relevance to very large graphs with millions of nodes and tens of millions of edges.
- 2) Protein Networks: When studying protein networks, researchers are often interested in proteins that interact with other highly interactive proteins. These proteins appear in degree cores with high values of k [8, 9].
- 3) Internet graphs: When analyzing large graphs the filtering of irrelevant nodes is often used as a pre-processing step before large graphs are analysed. Low degree nodes are often the nodes filtered when examining Autonomous System graphs of the internet. The core number of a node has been used in place of the degree for filtering these graphs [10].
- 4) Approximation of betweenness scores: The betweenness score reflects the number of shortest paths between all node pairs that a node lies on. This is a computationally expensive feature to calculate across a large graph. It has been shown in experiments with web graphs that the core number of a vertex is highly correlated with this score and might be used as a more efficient substitute [6].

3 Methodology

In order to summarize the local behaviour of a graph, we compute every non-empty degree core of a graph and identify the connected components of these subgraphs. These components, in turn, form a hierarchy where two components have a parent-child relationship, when the latter has been immediately split from the former. We say a component, A , has split from a component B iff A is a subgraph of B and A is a component of the k -core and B a component of the $(k+1)$ -core for some integer k . Here $V(A)$ is taken to the set of vertices contained within the component A . We refer to the tree thus generated as a hierarchical degree core tree and use this new structure for both graph summarization and feature extraction. An example of a graph and its corresponding hierarchical degree core tree can be seen in Figure 1.

It is clear from this definition that, given a node within the k th level of our tree, we have a representative cluster, i.e. the connected component containing the node, within the k -core of our graph. When proceeding to the $(k + 1)$ st layer

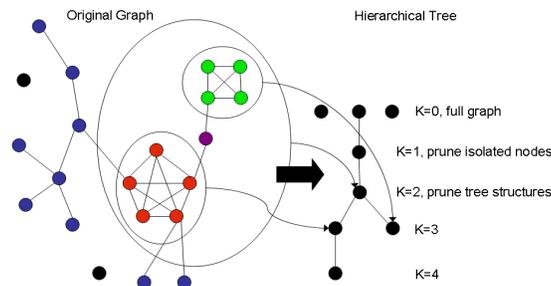


Fig. 1. A sample graph and its corresponding hierarchical degree core tree. Three components and their corresponding vertices within the hierarchical degree core tree have been circled for clarity. The larger circle is the 2-core while the union of the smaller circles makes up the 3-core.

of our tree this vertex may have:

- 1) no children, implying that the component it represents is no longer present within the $(k + 1)$ -core.
- 2) a single child, implying that the component remains as a single component within the $(k + 1)$ -core, though it may or may not have been reduced in size.
- 3) multiple children, implying that the removal of vertices with core number k resulted in the component splitting into multiple components.

This hierarchical tree can, at a glance, reveal a great deal of information concerning the local structure of very large graphs. It efficiently allows one to identify highly connected regions of a graph across a variety of filtering resolutions which are represented by the values of k . This tree could then be used by a domain expert to identify components of interest, similar to the way in which dendrograms are used to identify clusters of interest when performing hierarchical clustering [11].

In order to meaningfully compare massive, real-life graphs with graphs generated according to a given model, it becomes necessary to represent them as a set of features rather than as a full graph structure. The difficulty lies in determining a set of descriptive features sufficient for describing the structure of our data. Our research investigates the extraction of such features from the hierarchical degree core tree. We focus primarily of the number of components in the degree core subgraph across all k values. We also examine the size of the components in our hierarchical tree and the distribution of children between each layer of this tree. It should be noted that these are only a small subset of features that might be extracted from the hierarchical degree core tree. One might also be tempted to examine the distributional statistics such as the mean and standard deviation of component sizes at each level or the degree distribution within each of the degree core components.

4 Data

We look at a number of generative models described in the current literature [12, 13] and compare these against four real world graphs in order to determine the likelihood of their being generated by one of these models. We also compare the degree cores of our various generative models to determine if we can differentiate between these families of graphs.

Our four real world datasets consist of a webcrawl of: the .gov domain [14], the NBER patent citation graph [15], an email graph [16], and a subset of the DBLP coauthorship graph [17]. Each of these graphs possesses a power law degree distribution, though they possess distinctly different structures. The web graph is a web crawl of the .gov websites from 2002 and was used as a Text REtrieval Conference (TREC) data set. It contains approximately 1.2 million unique URL's and 9.7 million links between them. The NBER patent database contains all U.S. patent citations from January 1st, 1975 to December 31 1999, consisting of approximately 3.8 million nodes and 16 million edges. The email graph is an anonymized collection representing 16 months of email sent and received by 16,000 users in the computer science department at Dalhousie University. It consists of approximately 5 million distinct email addresses and 12.7 million edges. Here an edge represents the fact that two email addresses communicated, but does not take into account the direction or frequency of communication. The subset of the DBLP citation graph consists of some 15,000 non-isolated vertices and 360,000 edges. The vertices represent journal articles with edges connecting two papers if and only if they share a common author. This graph was originally used in [17] for the purposes of classification. To this end each article has been tagged with a subject: Databases, Machine Learning or Theory, based on the conferences in which the papers were published. Each of the graphs consists of one major component and a number of smaller disconnected components.

The Web, along with many other real world graphs such as the four graphs above, exhibits a power law degree distribution. It has been shown that preferential attachment models mimic this and other properties of these graphs. As such, these models are ideal for the purposes of our study. Preferential attachment models are a family of models which are generally built over time, and in which the probability of a new edge connecting to a previous node is directly proportional to the degree of said node. For comparison, we also examine the degree core structure of models generated by Erdős and Rényi (ER) models [18, 19]. Such models are generally referred to as $G(n, p)$ where n is the number of nodes in the graph and $p \in (0, 1)$ is the probability of an edge existing between any two nodes.

The linear cord diagram (LCD) model was first proposed by Barabási [20] and more rigorously defined by Bollobás [21]. In this model we start with an initial graph G_0 and add a vertex to this graph at every time step. Thus, after t time steps the graph G_t has is of size $|V(G_0)| + t$. When a vertex, v_t , is added at time step t , m edges are also added connecting v_t to m vertices in G_{t-1} . These vertices are selected with probability proportional to their degree.

The main difficulty with the LCD model is that its structure is purely additive. As such, the nodes with the highest degree are simply the nodes which have been in existence for the longest time. In order to circumvent this, we examine the deletion model of Chung and Lu [22], in which there is a random probability at each step for the following changes: adding a new node connected to m previous nodes selected via preferential attachment, adding m edges whose endpoints are chosen by preferential attachment, deleting a node selected uniformly at random, or deleting m edges selected uniformly at random. Here the model is specified by the probabilities of choosing each of these options and $m \in \mathbb{Z}$.

Finally, the Chung and Lu partial duplication model [13] was examined. This model requires an initial graph for which we used K_{10} and K_{500} . Here K_i represents the complete graph of size i , or equivalently the graph of size i where an edge connects every vertex pair. At each time step, a vertex is randomly selected and copied. Each of this vertex's edges are then copied with a probability p . Chung and Lu show that the partial duplication model generates power law graphs whose power law exponents are dependent only upon the growth process, and thus independent of the initial graph.

5 Results

The hierarchical degree core trees derived from the web graph, patent citation graph, co-authorship graph and email graph can be seen in Figures 2, 3, 5 and 4 respectively. All these graphs, with the exception of the co-authorship graph, consist of a single large component and a large number of very small components which vanish completely for values of $k > 3$. In this graph, a number of components persist for $k < 22$. For clarity, we examine the degree core component distribution of the single large component of the first three graphs and the hierarchical tree for $k > 12$ in the case of the co-authorship graph.

Though the hierarchical trees and the component distributions differ between these two graphs, the web graph and the patent citation graph do share one potentially interesting similarity, a unimodal component distribution. As k increases, both graphs fragment into multiple components. The number of components increases until reaching a peak, then decreases as k continues to rise. The components that split off are small compared to the largest component, only consisting of, at most, a few hundred highly connected vertices compared to the main component which possesses thousands of vertices. Though smaller in size, these nodes remain significant in that the components fragment from the main graph at very high values of k . In the case of the web graph, 4 components have fragmented off by $k = 83$ indicating large highly connected subregions of our graph that are only loosely connected with our main component. This unimodal distribution is much more striking in the case of the patent citation graph, with 36 components separating off from the main component at $k = 14$. The smooth decrease in the number of components is likely due to the wide variance in the component sizes of our highly connected subgraphs. As k increases slowly these components are pruned equally slowly. The more interesting feature is the

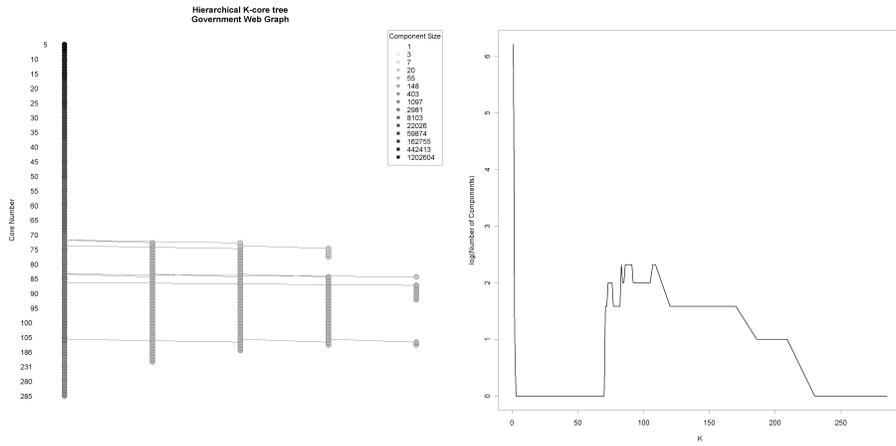


Fig. 2. The hierarchical degree core tree and corresponding component distribution for the .gov web graph. The hierarchical tree only shows the tree derived from the major component of the email graph and the component distribution is on a \log_2 scale. This is due to the fact that the initial graph contains a large number of small isolated components.

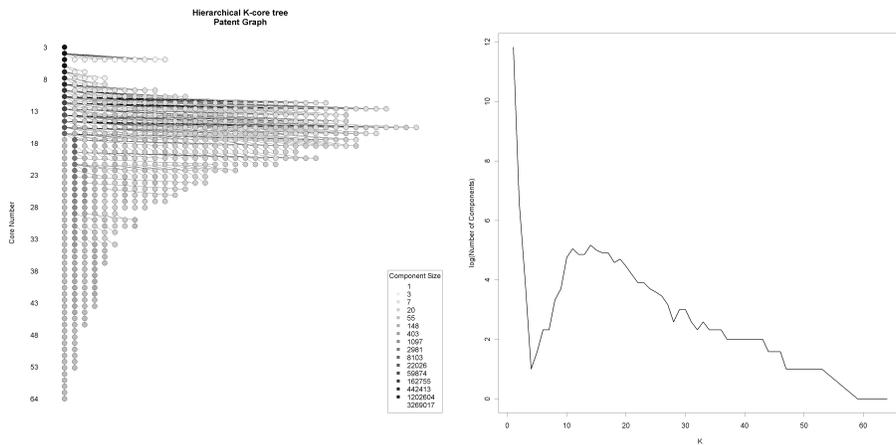


Fig. 3. The hierarchical degree core tree and corresponding component distribution for the NBER patent citation graph. The hierarchical tree only shows the tree derived from the major component of the email graph and the component distribution is on a \log_2 scale. This is due to the fact that the initial graph contains a large number of small isolated components.

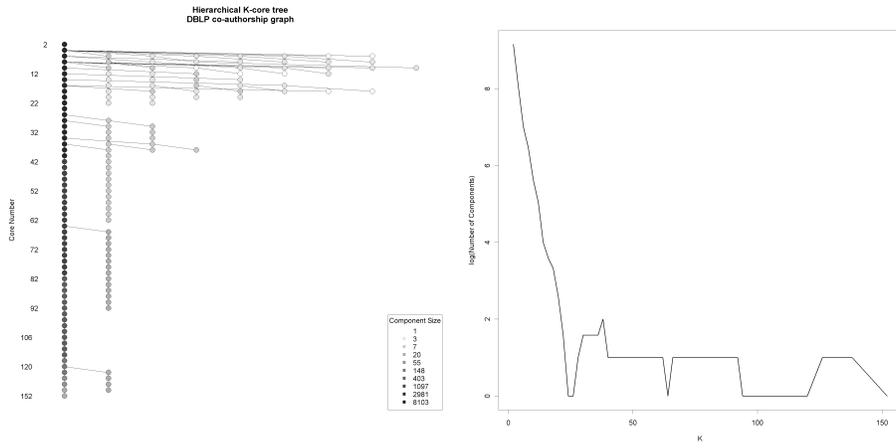


Fig. 4. The hierarchical degree core tree and corresponding component distribution for the DBLP co-authorship graph. The hierarchical tree only shows the tree derived from the major component of the email graph and the component distribution is on a \log_2 scale. This is due to the fact that the initial graph contains a large number of small isolated components.

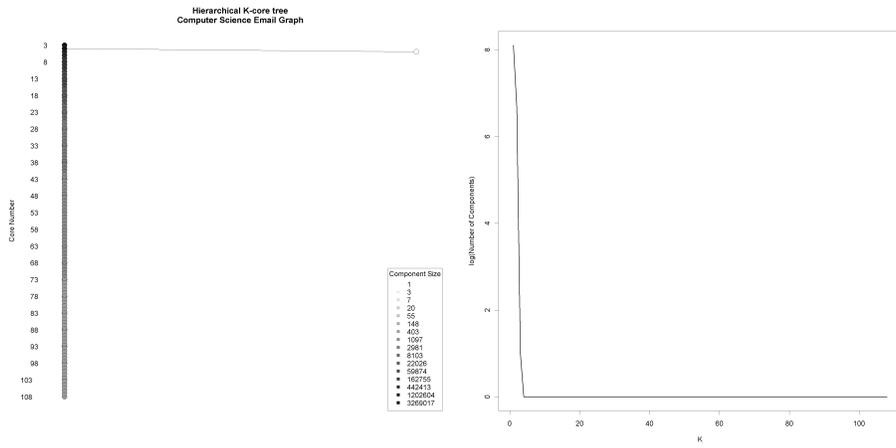


Fig. 5. The hierarchical degree core tree and corresponding component distribution for the Dalhousie computer science email graph. The hierarchical tree only shows the tree derived from the major component of the email graph and the component distribution is on a \log_2 scale. This is due to the fact that the initial graph contains a large number of small isolated components.

smooth growth in this distribution. This is a feature shared by none of the generative models examined up to this point. In both graphs the vast majority of components fragment directly from the main component and then vanish without splitting themselves. In both graphs this points towards interesting structures occurring near the modes of these distributions.

This strongly implies that there is a great deal of local structure within both of these real world graphs. Further, it implies that the local regions of these two graphs behave in a similar fashion, breaking apart from the main component around a single value of k . This implies that the behaviour of our vertices is far from uniform and that there is an unexplored underlying process at work.

The co-authorship graph shows striking differences to both of these graphs. Firstly, its initial components are better connected and remain in the hierarchical tree to a much greater depth. Secondly, there are two components in the tree across the vast majority of the levels, occasionally vanishing to be replaced by a new child of the large component. As before, the secondary component is much smaller than the major component (consisting of less than one hundred vertices) and fragments directly from the major component.

The main component of the email graph has a different structure entirely, as seen in Figure 5. With the exception of a tiny component early on, the single large component fails to fragment at any point along the degree core hierarchy. Though this differs strongly from our previous three real world graphs, it does bear a striking resemblance to the majority of the generative models examined in this paper.

When we compare the distributions of our first three real world graphs against those of our generative models we see a clear difference. All of the models examined generate single-component graphs that vanish after reaching a given k instead of breaking apart as do most of our real world graphs. The LCD model, differs most significantly in structure, as no nodes are pruned until a given k is reached and then it vanishes entirely. This behaviour is unsurprising as it is predicted by Theorem 1.

Theorem 1. *For any graph, G , generated by the LCD model, the k -core $H_k = G$ (and is made up of a single component) for $k \leq m$ and $V(H_k) = \emptyset$ for $k > m$, where m is the number of edges added to the graph at each step.*

Proof. At each step of the iterative construction process, a vertex is added along with m edges connected to vertices already present in the graph. This process generates a single component graph with a minimum vertex degree of m . Therefore, for $k \leq m$ no vertices will be pruned, resulting in a single component.

For $k > m$ the most recently added vertex is guaranteed to have degree $= m$ and is thus removed. This pruning guarantees that the previously added vertex will now have degree $= m$. This iterative argument continues until all vertices are pruned, resulting in zero components.

In the case of the ER graph, we see a single component that persists across a very small range of k values before vanishing. Though not guaranteed, this is not surprising given the homogeneous nature of the vertices in an Erdős-Rényi

graph. This difference is not entirely unexpected, as the ER graph possesses an entirely different degree distribution from our real world graphs. This analysis was included as a comparative baseline.

It appears that the CL-Deletion model is too similar to the LCD model to generate the localized behaviour which we see in our real world graphs. The addition and deletion of nodes and edges in this model occur uniformly at random across the graph. As such, they do not serve to add sufficient local structure to mimic the component distributions found in our real world examples. They do, however, serve to elongate our hierarchical tree structure by creating several distinct degree core graphs as k varies.

The duplication model was examined with the intent that it might contain more local structure. Every simulation of this model resulted in a single large component and a number of very small isolated vertices. In each case, the single main component remained in our tree across a large range of k , never generating more than a single child at each step. Though the duplication model did not match our first three real world graphs, its distribution was the most similar to that of our email graph. Both hierarchical degree core trees possessed a single large component slowly shrinking across a wide range of k values and never fragmenting.

6 Conclusion

Matching any number of features possessed by two graphs should not be sufficient to determine if both graphs were generated by the same family of models. However, failing to match features is sufficient for us to reject the hypothesis that a particular model generated a graph. Very few strong independent features have been proposed to characterize the structure of a graph. We have shown that hierarchical degree core trees possess a number of features which are useful in identifying the local structures within a graph.

The degree core component distribution illustrates a rich local structure contained within our real world graphs. The main component in the NBER patent citation graph and the government web graph both possessed a roughly unimodal degree core component distribution, while the DBLP co-authorship and email graphs possessed a more uniform structure. The email graph demonstrated the simplest structure with a single large component persisting through the entire tree. None of these structures would have been identified without the aid of our hierarchical degree core trees.

It is clear that none of the generative models examined up to this point have demonstrated any form of complex structure under the degree core component distribution. Each of them generates large single component graphs which fail to split into smaller components for any value of k . This structure, particularly that shown by the duplication model, seems to match that of the email graph. One should be careful to refrain from making the assumption that the process underlying the email graph was a partial duplication process. As seen with power law models, there are many different models that could be used to generate a

graph with a particular feature. It is necessary to examine a larger number of features before making any claims about the underlying process of a given graph.

The hierarchical degree core trees introduced in this paper effectively summarize the local structure contained within our real world graphs, which allows easy identification of interesting structures contained within a given graph. Even for very large graphs these trees are easily visualized. They can assist a user to determine the k value for which the degree core subgraph will contain the richest amount of information. This technique can be useful for both filtering and data exploration. In the case of at least two of our real world graphs, a good initial filter might be selecting the mode of the unimodal component distribution.

7 Future Work

Future research will involve enriching our hierarchical trees with extra information to provide a more complete summary of the graph in question. Some methods we have begun to examine include colouring vertices by the size of the component it represents or by examining the number (or proportion) of vertices of interest contained within the given component. We will continue to examine other features derived from our hierarchical trees which may prove useful in the characterization of large graphs.

Batagelj and Zaveršnik[2] extend the concept of degree cores to vertex features besides degree, such as the number of cycles passing through a given vertex. Using these other features to induce our hierarchical degree core trees may provide greater insight into the structure of our graphs.

As degree cores naturally extend to other vertex features besides degree, we feel that examining the hierarchical degree core trees induced by some of the features described by Batagelj and Zaveršnik[2] could lead to interesting insights into the structure of our graphs.

The segmentation of our graphs into separate highly connected subgraphs suggests natural clusters within the graphs. Though these subgraphs are too small to represent major clusters in themselves, they could easily represent the backbone of larger, more loosely connected clusters. These components may or may not possess a meaningful interpretation. If they do, we will examine the use of this technique for graph clustering.

Improved graph layout techniques should improve the interpretability of our larger hierarchical trees.

Currently the code in use for computing our hierarchical degree core trees is written in C++ and makes use of the Boost Graph Library [23]. As such, it scales well to graphs containing millions of vertices and tens of millions of edges on conventional hardware. For analyzing larger graphs it will become necessary to either make use of parallel BGL or to write our own external memory algorithm.

References

1. Pržulj, N., Corneil, D.G., Jurisica, I.: Modeling interactome: scale-free or geometric? *Bioinformatics* **20**(18) 3508–3515

2. Batagelj, V., Zaveršnik, M.: Generalized Cores. ArXiv Computer Science e-prints (2002)
3. Seidman, S.B.: Network structure and minimum degree. *Social Networks* (1983) 269–287
4. Wasserman, S., Faust, K.: *Social network analysis: Methods and applications*. Cambridge University Press, Cambridge (1994)
5. Alvarez-Hamelin, I., Dall’Asta, L., Barrat, A., Vespignani, A.: DELIS-TR-0166 - k-core decomposition: a tool for the visualization of large scale networks. techreport 0166, DELIS – Dynamically Evolving Large-scale Information Systems (2004)
6. Alvarez-Hamelin, I., Barrat, A., Dall’Asta, L., Vespignani, A.: k-core decomposition: a tool for the analysis of large scale internet graphs. *Computer Science*, cs.NI/0511007 (2005)
7. Batagelj, V., Mrvar, A.: Pajek - Analysis and Visualization of Large Networks. Volume 2265. (January 2002)
8. Wuchty, S., Almaas, E.: Peeling the yeast protein network. *Proteomics* (2005)
9. Bader, G.D., Hogue, C.W.: An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics* **4** (2003) 2
10. Gaertler, M., Patrignani, M.: DELIS-TR-0003 - dynamic analysis of the autonomous system graph. In *Proceedings 0003, 2004, IPS 2004 (Inter-Domain Performance and Simulation)* (2004)
11. Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer (2001)
12. Bonato, A.: A survey of models of the web graph. In: *Proceedings of the Workshop on Combinatorial and Algorithmic Aspects of Networking (CANN 2004)*, Springer (2004)
13. Chung, F., Lu, L., Dewey, T.G., Galas, D.J.: Duplication models for biological networks. *Journal of Computational Biology* **10**(5) (2003) 677–687
14. TREC: The .gov test collection (last accessed Aug. 28, 2006.)
15. Hall, B.H., Jaffe, A.B., Trajtenberg, M.: The nber patent citation data file: Lessons, insights and methodological tools. NBER Working Papers 8498, National Bureau of Economic Research, Inc (October 2001) available at <http://ideas.repec.org/p/nbr/nberwo/8498.html>, last accessed Aug 30, 2006.
16. Wan, X., Janssen, J., Kalyaniwalla, N., Milios, E.: Statistical analysis of dynamic graphs. In: *Proceedings of AISB’06: Adaptation in Artificial and Biological Systems*. Volume 3. (2006) 176–179
17. Angelova, R., Weikum, G.: Graph-based text classification: Learn from your neighbors. In: *29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval (SIGIR’06)*, Association for Computing Machinery (ACM), ACM (2006) 485–492
18. Erdős, P., Rényi, A.: On random graphs. *Publicationes Mathematicae* (1959)
19. Erdős, P., Rényi, A.: On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.* (1961)
20. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. *Science* **286** (1999) 509–512
21. Bollobás, B., Riordan, O., Spencer, J., Tusnády, G.: The degree sequence of a scale-free random graph process. *Random Structures Algorithms* (2001)
22. Chung, F., Lu, L.: Coupling online and offline analyses for random power law graphs. *Internet Mathematics* (2003)
23. Siek, J.G., Lee, L.Q., Lumsdaine, A.: *The boost graph library: user guide and reference manual*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2002)