Patterns on the Connected Components of Terabyte-Scale Graphs

U Kang, Mary McGlohon, Leman Akoglu, and Christos Faloutsos
SCS, Carnegie Mellon University
{*ukang, mmcgloho, lakoglu, christos*}@*cs.cmu.edu*

***Abstract*—How do connected components evolve? What are the regularities that govern the dynamic growth process and the static snapshot of the connected components? In this work, we study patterns in connected components of large, real-world graphs. First, we study one of the largest static Web graphs with billions of nodes and edges and analyze the regularities among the connected components using GFD(Graph Fractal Dimension) as our main tool. Second, we study several time evolving graphs and find dynamic patterns and rules that govern the dynamics of connected components. We analyze the growth rates of top connected components and study their relation over time. We also study the probability that a newcomer absorbs to disconnected components as a function of the current portion of the disconnected components and the degree of the newcomer. Finally, we propose a generative model that explains both the dynamic growth process and the static regularities of connected components.**

***Keywords*-Evolution of Connected Components, CommunityConnection Model, Graph Mining**

I. Introduction

Given a large, time-evolving network, what regularities can we observe? Large network structures are ubiquitous, including on-line social networks, the World Wide Web, and more. The study of these networks has gained much attraction in recent years. However, computational difficulties for analyzing Terabyte-scale graphs limited the analysis of large networks. In this work, we study the structure of large networks using HADOOP, an open-source implementation of MAPREDUCE. We study both the static and dynamic network properties, focusing on fractal dimensions and connected components. To our knowledge, this is the first study on the relationships of connected components of billion-scale graphs. The main contributions are the following:

1) *Graph Fractal Dimension (GFD)* : We study the connected components in the static snapshot of the Web graph and introduce the concept of GFD. We find they share statistical properties with the GCC(Giant Connected Component), although there are some surprising large star-like shapes (small GFD), and even more surprising large full cliques (large GFD, and possibly dangerous: with financial, or adult content, which existed several years ago but non-existing now).
2) *ERP pattern*: We study the time evolution, and specifically the merge process of connected components. We discover the ERP (Exponential Rebel Probability) pattern: the probability of a newcomer not to join the GCC ('rebel' probability) is exponentially decaying with the degree of the newcomer node (Equation (1)).
3) COMMUNITYCONNECTION *model*: We give a generative model that explains both the static and dynamic regularities of connected components. We also show the ERP pattern can be derived from our model.

The rest of the paper is organized as follows: Section II studies patterns of the connected components in a static WWW graph. In Section III, we describe the ERP finding, and study the dynamics of components. We further explain the underlying mechanisms by proposing a generative model in Section IV with discussions in Section V. After describing related works in Section VI, we conclude in Section VII. Table I lists the symbols and terms used in this paper, and Table II describes the networks we study.

Symbol	Definition
G	a graph
V	set of nodes in a graph
E	set of edges in a graph
AER	Average effective radius
MER	Maximum effective radius
CC	Connected component
GCC	Giant connected component
DC	Disconnected(non-giant) component

Table I
SYMBOLS AND DEFINITIONS

Name	$\|V\|$	$\|E\|$	Storage	Time?	Desc.
YahooWeb	1.4B	6.6B	120GB	No	Web pages crawled by Yahoo in 2002
U.S. Patent	6M	10M	169MB	Yes	Patent citations
HEP-TH	27K	351K	8MB	Yes	Physics citations
HEP-PH	30K	347K	8MB	Yes	Physics citations

Table II
ORDER AND SIZE OF NETWORKS. M: MILLION, B:BILLION, K: THOUSAND.

II. Homogeneity of Components

Do small components have the same structural properties as the giant connected component (GCC)? In this section, we study the homogeneity of components in large networks. The main questions are the following:

1) How can we characterize the density of a connected component? Is there an intrinsic measure invariant to the growth of the connected component?
2) Do connected components have same densities?

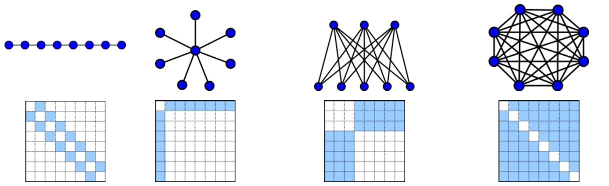
3) How can we characterize the connected components in terms of radius?

For the study, we investigate the connected components of YahooWeb graph. It is a static snapshot of the WWW which was crawled by the Yahoo-Altavista search engine in 2002. It contains 1.4 billion web pages and 6.6 billion links among them. The giant connected component contains 690 million web pages (about 50% of the total pages). The second and the third largest connected components contains 57,000 and 21,000 pages—much smaller than the GCC. Except the isolated (one node) connected components, there are 2.6 million connected components. Our goal is to find interesting patterns on the connected components of different sizes to better understand the evolution of networks.

We first characterize the density of connected components. Many different definitions can be possible: the most intuitive one is arguably the ratio of the size (number of edges) and the order (number of nodes) of the graph. However, there is a power-law relationship between the size and the order of the whole graph, and the exponent remains constant over time [1]. Therefore, scaling it by log is more natural and thus we propose the Graph Fractal Dimension(GFD) to characterize the 'density' of a connected component.

*Definition 1 (**Graph Fractal Dimension**):* The graph fractal dimension GFD(C) of a connected component C is the ratio of the size and the order in log scale. That is, $GFD(C) = \frac{log|E(C)|}{log|V(C)|}$.

For example, a clique will have GFD ≈ 2, because there are $n^2 - n$ edges for n nodes. A chain will have GFD ≈ 1, because there are $n - 1$ edges for n nodes. Several graphs and their fractal dimensions are shown in Figure 1.



(a) Chain: GFD 1.27 (b) Star: GFD 1.27 (c) Bipartite-Core: GFD 1.64 (d) Clique: GFD 1.94

Figure 1. Graphs, adjacency matrices and their fractal dimensions. Notice that the GFD can be used as a measure of density of graphs: chain or star graphs have smaller GFDs while cliques have higher GFDs.

A. Graph Fractal Dimension

How are the graph fractal dimensions of connected components distributed? Do they share regularities? Figure 2 shows the graph fractal dimension of connected components in YahooWeb graph. In Figure 2 (a), there exists various components with wide range of the number of edges for a fixed number of nodes, between the minimum (tree) and the maximum (clique). However, after averaging the number of edges in Figure 2 (b), we observe a striking pattern:

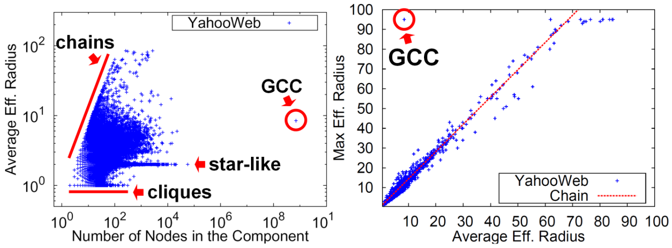
*Observation 1 (**Homogeneity of Components GFD**):* Graph fractal dimensions of connected components in YahooWeb graph are constant on average.

This observation implies that the connected components of the Web graph are self-similar, regardless of the size of the network. Then, it is important to have a graph evolution model that produces this feature. Our proposed COMMUNITYCONNECTION model, which is described in Section IV, has the feature: see Figure 2 (c) and (d) which shows the homogeneity of GFD in the synthetic graph generated from our model.

B. Radius of Connected Components

Next, we look at the relation between the average effective radius and the order of connected components. Recall the effective radius of a node in a component is the 90th percentile of all distances to other nodes in the component. The average effective radius (AER) of a connected component is the average of the effective radius in the component.

Figure 3(a) shows the number of nodes and the AER of connected components. We first see that there are many components which form cliques (AER close to 1), stars (AER close to 2), and chains (AER proportional to the number of nodes). Since a component that behaves like a clique seems suspicious, we looked at the top 4 largest clique-like component, and found they all belong to Germany, they have the same number of nodes (305), and have similar contents. They seem to be phishing sites from the same owner, and do not exist any more.



(a) AER vs. $|V|$ (b) MER vs. AER

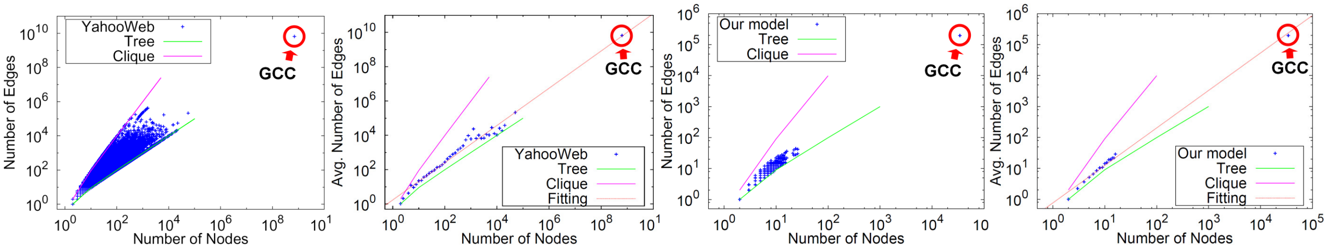
Figure 3. **(a)** Connected components map of the YahooWeb graph, showing the Average Effective Radius(AER) vs. number of nodes in each component. Notice the effective radii are bounded by the maximum(by chains) and the minimum(by clique). **(b)** Maximum Effective Radius(MER) vs. Average Effective Radius(AER). Each point represents a connected component. We halted the computation at the effective radius 95. Notice that small disconnected components behave like chains in terms of radius, and the GCC is very different from others due to the large MER compared to the AER.

Another observation is that the upper left boundary of Figure 3(a) forms a near-line at the maximum AER (indicating these components are chains). We can prove the boundary is a near-line since the AER and the number of nodes in chains have the following near-linear relationship:

*Lemma 1 (**AER of Chain Graph**):* The average effective radius of a n-node chain graph is $0.6525n - 0.45$.

Proof: See the journal version of this paper [2]. ■

Next, we look at the maximum effective radius versus the average effective radius of each component in Figure 3(b). We have the following observation.



(a) $|E|$ vs. $|V|$ (b) Avg. $|E|$ vs. $|V|$ (c) $|E|$ vs. $|V|$ (d) Avg. $|E|$ vs. $|V|$

Figure 2. Homogeneity in the fractal dimension of components. **(a)(b)**: YahooWeb graph. **(c)(d)**: Synthetic data from our CommunityConnection model in Section IV. In (a) and (c), each point corresponds to a connected component. In (b) and (d), Y axis represents the average number of edges of components with the corresponding number of nodes. Notice the fractal dimension of components fits in a line, for both the real-world graph and the synthetic graph generated from our model.

*Observation 2 (**Chain-like Disconnected Components**):* Disconnected components have relatively small MER (Maximum Effective Radius) vs. AER (Average Effective Radius) ratio. Only the GCC has the high ratio.

The reason of the high MER vs. AER ratio of the GCC can be explained by the thick cores containing majority of nodes and several tendrils it has. The thick cores decrease the AER, while the tendril increases the MER. On the contrary, disconnected components of the YahooWeb graph do not have enough number of nodes to form thick cores which could have decreased the average effective radius. In terms of MER vs. AER ratio, the disconnected components are similar to chain graphs.

III. Absorption of Components

In previous sections, we studied static web graphs without timestamps. Next, we broaden our focus to time-evolving graphs and study the dynamic aspects of the connected components. The main questions are the following:

1) Will the connected components grow with the same rate? Will there be a change of growth rate around the "gelling" point[3] where the diameter of the graph starts to shrink?
2) Given the degree of a newcomer, what is the probability that it will be absorbed, or not absorbed to GCC?

To answer the first question, we look at the GFD of top 3 largest connected components over time in Figure 4 and summarize findings in Observation 3. We refer to the second and third largest connected components as the first and second DC, as they are "disconnected" from the GCC.

*Observation 3 (**Evolution of Top 3 C.C.**):* The GCC, the largest DC, and the second largest DC grow with the same rate. Their GFDs remain the same until a deviation point. The deviation point is close to the "gelling" point where the diameter starts to shrink.

This observation is interesting: it implies that some barriers between the nodes seem to collapse after the gelling point, and the nodes in the network are connected with higher rate than before the gelling point.

The next question is, what is the probability of newcomers not joining to the GCC? We call this the "rebel" probability and show its relation to the degree of newcomers and the portion of nodes in the DCs. The relationship of the rebel probability and the degree of newcomers is shown in Figure 5. We see that the probability is linear to the degree in log-lin scale where the slope decreases as the network grows. In addition, we show the relationship of the rebel probability and the portion of nodes in DC in Figure 6. From Figure 6, we see the probability is linear to the portion of nodes in DC in log-log scale, and the slope increases as the degree increases. Given these two observations, we give empirical rebel probability of newcomers as a function of the degree d and the portion s of nodes in DC in Observation 4 which we call the ERP (Exponential Rebel Probability) pattern.

*Observation 4 (**Exponential Rebel Probability**):* Given the node portion s of DCs, the probability P_{rebel} of a newcomer to be absorbed in DCs is exponential to the product of a constant α, the degree d of the newcomer, and the log of the node portion s:

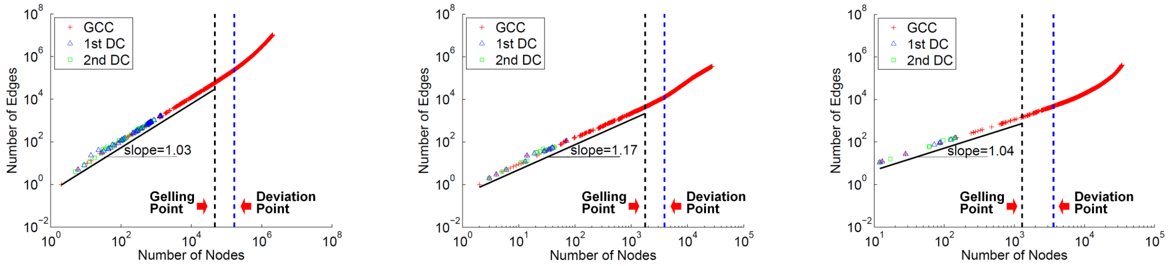
$$\boxed{P_{rebel} \propto e^{\alpha d (log s)}} \tag{1}$$

IV. Proposed Model

In this section, we propose a generative model to explain the static and dynamic aspects of connected components.

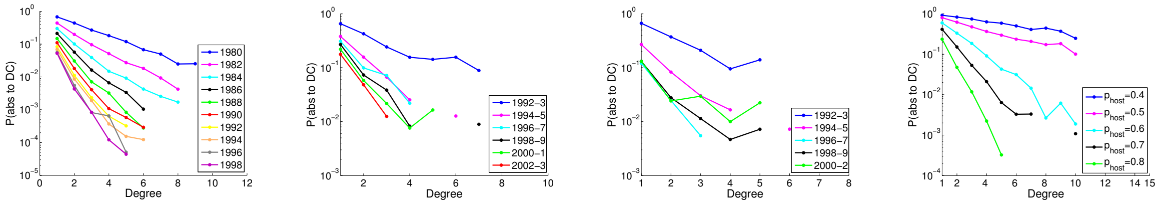
A. CommunityConnection Model

We propose CommunityConnection model for explaining the evolution of the connected component. The model proceeds as follows. Given a network (starting with a single node), nodes arrive one at a time, behaving as "social connectors". There are two main processes that a new node n_{new} alternates between: `choose_host` and `visit`. At the top level, in `choose_host`, a new node flips a p_{host} coin. If the result is negative, the node stops linking and returns. If the result is positive, n_{new} chooses a random node in the network as a "host", and then enters the `visit` process, which is a random walk within a component. Starting with a host, n_{new} flips a p_{step} coin. If the result is positive, the node forms a directed edge to the visited node and chooses a random neighbor to visit next. The random walk continues until p_{step} coin-flip is false, and the node returns back up to the `choose_host` procedure. In



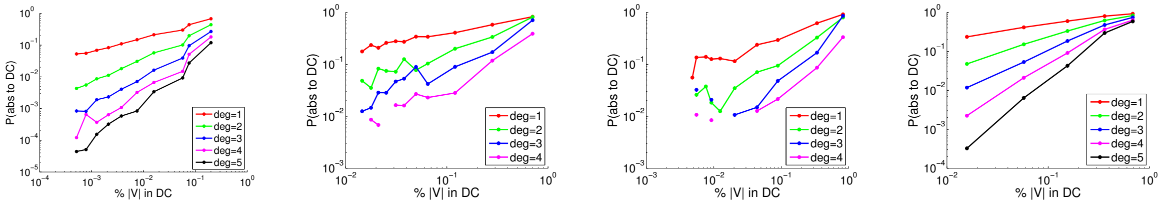
(a) U.S. Patent: Top 3 CC (b) HEP-TH: Top 3 CC (c) HEP-PH: Top 3 CC

Figure 4. Growth of connected components in terms of the graph fractal dimension. Each point represents the snapshot of a connected component over time. Notice that the slope remains constant until a 'deviation point'(the second vertical line) close to a 'gelling point'(the first vertical line), and starts to increase after that. The deviation points are about one year after the gelling points.



(a) U.S. Patent (b) HEP-TH (c) HEP-PH (d) Our Model

Figure 5. P(Absorption to DC) vs. Degree in log-lin scale. Notice the linear drop of the probability as the degree increases. Also notice that our CommunityConnection model in (d), which will be described in Section IV, captures the same linear drop.



(a) U.S. Patent (b) HEP-TH (c) HEP-PH (d) Our Model

Figure 6. P(Absorption to DC) vs. Portion of Nodes in DC in log-log scale. Notice that the slopes of curves increase as degree increases. Also notice that our CommunityConnection model, which will be described in Section IV, captures the property as shown in (d).

short, an arriving node alternates between the "host" process and the "random walk" process until there are no more hosts chosen. Pseudocode of CommunityConnection is shown in Figure 7.

Unlike many previous models, this produces disconnected components due to the different number of hosts that are chosen. Some nodes may choose 0 hosts and form a component entirely disconnected from the others. Other nodes may choose multiple hosts in different components, merging components together.

We use the CommunityConnection model to generate a series of edges, and then run analysis on those edges. We find that many properties are replicated under the CommunityConnection model. The component growth resembles that found in real data, as shown in Figure 2(c) and (d). These plots were generated with $p_{host} = 0.5$; results for other values are similar. Figure 5(d) shows the empirical probability of "Rebelling" given degree. Since under the model the percentage of nodes in the GCC remains constant over time, we modified the p_{host} parameter to show varying behavior. Figure 6(d) shows the probability for different degrees under varying the percent of nodes in DC.

B. Theoretical Analysis

We show that the ERP(Exponential Rebel Probability) pattern of the growth of connected components can be derived from our CommunityConnection model. In the model, the probability of choosing to connect to a given component is dependent on the number of nodes in the component, since the hosts are picked at uniform. In order for an arriving node to join a component, it must first choose a host within that component. Noting the random walk can access any node in the component, but not outside, we can derive the probability of a node "rebelling" represented as an event variable R, given the portion s of the graph's nodes in DCs and the degree d of the newcomer.

The algorithm gives us the following distributions: the number of hosts chosen has a geometric distribution with parameter $1 - p_{host}$ (the number of coin flips until "no new host"). The length of the random walk after a given host has a geometric distribution with parameter $1 - p_{step}$. So the total degree is the sum of the random walks. Now, we can show the probability distribution of rebelling given degree.

```
function CommunityConnection(global p_host = 0.5,
global G = new_graph())
{
  for n = 1:N
    current=new_node()
    p_step = SampleUniform(0,1)
    G.add_node(current)
    choose_hosts(current)
  return(G)
}

// input: a new node
// effect: iteratively chooses hosts, starts visit process
//         for each
function choose_hosts(current)
{
  while (SampleUniform() < p_host)
    host = G.random_node()
    visit(current, host)
  return
}

// input: a newcomer, and host node to visit
// effect: probabilistically links and recursively calls next
//         step of random walk
function visit(current, host)
{
  // with probability p_step, link and continue random walk
  if (SampleUniform() < p_step)
    G.add_directed_edge(current, host)
    next_visit = chooseRandom(Union(host.neighbors(), host))
    visit(current, next_visit)
  return
}
```

Figure 7. Pseudocode for the COMMUNITYCONNECTION model.

*Lemma 2 (**Probability of "Rebelling"**):*

$$P(R = true|s, D = d > 0) = \frac{\sum_{h=1}^{d} NBin(d,h,1-p_{step}) * Geom(h+1,1-p_{host}) * s^h}{\sum_{h=1}^{d} NBin(d,h,1-p_{step}) * Geom(h+1,1-p_{host})} \quad (2)$$

where NBin and Geom are the PDF of negative binomial and geometric distribution:
$NBin(y, r, p) = \binom{r+y-1}{y} p^r (1-p)^y$ and
$Geom(x, p) = (1-p)^{x-1} p$.

Proof: See the journal version of this paper [2]. ■

We can show numerically that, for degree $0 < d < 10$, the formula exhibits exponential decay for any values of p_{step} and p_{host}. In fact, we can give the intuitive explanation of the rebel probability under COMMUNITYCONNECTION :

$$P(R = true|s, D = d) = s^{(1-p_{step})d} = e^{(1-p_{step})d(logs)} \quad (3)$$

Justification: A rough approximation to the degree of a newcomer node under the COMMUNITYCONNECTION is the number of hosts h it chooses, times the typical length L of each random walk. Since $P(L = l) \sim Geom(l, 1 - p_{step})$ and $E(L) = \frac{1}{1-p_{step}}$, $d \approx h * E(L) = \frac{h}{1-p_{step}}$. Therefore, $P(R = true|s, D = d) = s^h \approx s^{(1-p_{step})d} = e^{(1-p_{step})d(logs)}$.

V. DISCUSSION

In this section, we discuss potential applications of our findings and system issues for mining large graphs.

Potential Applications: We summarize the potential applications of our results.

- *Feature Extraction*: Newly introduced features(e.g., GFD and MER) can be used for learning algorithms.
- *Graph Generator*: Our COMMUNITYCONNECTION model and the ERP pattern can help improve graph generators(e.g., [4], [5]).
- *Extrapolation*: Our patterns and observations can be used to extrapolate unknown quantities.

System Issues: The main computational issue is to extract features of graphs including diameter, degree, and connected components of billions-scale graphs. We used several HADOOP algorithms by Kang et al. [6], [7] for the purpose and the algorithm scaled well for large graphs. We ran our code in Yahoo!'s M45 HADOOP cluster which is one of the top 50 supercomputers in the world, and have 1.5 Petabyte storage and 3.5 Terabyte memory in total.

VI. RELATED WORK

Related works form the following groups: the studies of structural patterns of networks, graph mining, and parallel graph mining using HADOOP.

Structural Patterns of Networks: There has been a significant amount of work in the realm of structural patterns in networks. Some well-known patterns include heavy-tailed degree distributions [8], [9]; "densification," or the super-linear power law relationship between the edge count and node count [1]; several power laws in terms of the edge weights [4]; and the formation of the GCC (first suggested in [10]) While much of the literature has focused on behavior inside the GCC, there has been some work examining the behavior of the smaller components, in terms of their power-law size distribution [11] and their thresholded size before merging with the GCC [3].

Graph Mining: There exists lots of "graph mining" algorithms: subgraph discovery(e.g., [12], [13], gPrune [14], gApprox [15], gSpan [16], Subdue [17], ADI [18], CSV [19]), computing communities (eg., [20], DENGRAPH [21], METIS [22]), attack detection [23], with too many alternatives for each of the above tasks. They are not directly related to the focus of this paper which is the static and dynamic structures of real networks.

Parallel Graph Mining using* HADOOP *: MAPREDUCE is a framework for processing web-scale data in parallel. HADOOP is the open source implementation of MAPREDUCE. Due to its scalability and the relatively cheap cost of building clusters, HADOOP has been used for important graph mining algorithms (see [24] [6] [7]).

VII. CONCLUSIONS

In this paper we found patterns on the evolution of connected components and proposed a model to explain the most striking of them, the ERP (*exponential 'rebel' probability*). The main contributions are the following:

- We proposed the *graph fractal dimension (GFD)* to measure the density of connected components and showed that the connected components in the static snapshot of the large web graph have constant GFDs on average. We also showed that the DCs have relatively small maximum effective radius(MER) vs. average effective radius(AER) ratio. On the other hand, the GCC has high MER vs. AER ratio due to enough number of nodes to form cores and tendrils.

- We studied the 'rebel' probability that a newcomer will avoid being absorbed into the GCC, and found that it decays exponentially with the degree.
- We proposed the COMMUNITYCONNECTION model to explain the growth process. We showed that the model captures both the growth of GFD as well as the 'rebel' probability of real-world dataset.

Future research direction includes finding patterns and evolutions in the community structures of large networks.

ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Grants No. IIS-0705359 IIS0808661 and under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract No. DE-AC52-07NA27344.

Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

This work is also partially supported by an IBM Faculty Award. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, or other funding parties.

REFERENCES

[1] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *KDD*, 2005.

[2] U. Kang, M. McGlohon, L. Akoglu, and C. Faloutsos, "Patterns on the connected components of terabyte-scale graphs," *ManuScript*.

[3] M. Mcglohon, L. Akoglu, and C. Faloutsos, "Weighted graphs and disconnected components: Patterns and a generator," *KDD*, 2008.

[4] L. Akoglu, M. Mcglohon, and C. Faloutsos, "Rtm: Laws and a recursive generator for weighted time-evolving graphs," in *ICDM*, 2008.

[5] L. Akoglu and C. Faloutsos, "Rtg: a recursive realistic graph generator using random typing," *Data Min. Knowl. Discov.*, vol. 19, no. 2, pp. 194–209, 2009.

[6] U. Kang, C. Tsourakakis, and C. Faloutsos, "Pegasus: A peta-scale graph mining system - implementation and observations," *ICDM*, 2009.

[7] U. Kang, C. Tsourakakis, A. P. Appel, C. Faloutsos, and J. Leskovec., "Radius plots for mining tera-byte scale graphs: Algorithms, patterns, and observations," *SDM*, 2010.

[8] R. Albert, H. Jeong, and A.-L. Barabasi, "Diameter of the world wide web," *Nature*, 1999.

[9] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," *SIGCOMM*, pp. 251–262, Aug-Sept. 1999.

[10] P. Erdos and A. Renyi, "On the evolution of random graphs," *Publ. Math. Inst. Hungary. Acad. Sci.*, vol. 5, pp. 17–61, 1960.

[11] R. Kumar, J. Novak, and A. Tomkins, "Structure and evolution of online social networks," *KDD*, 2006.

[12] M. A. Hasan and M. J. Zaki, "Output space sampling for graph patterns," *PVLDB*, 2009.

[13] J. Cheng, J. X. Yu, B. Ding, P. S. Yu, and H. Wang, "Fast graph pattern matching," *ICDE*, 2008.

[14] F. Zhu, X. Yan, J. Han, and P. S. Yu, "gprune: A constraint pushing framework for graph pattern mining," *PAKDD*, 2007.

[15] C. Chen, X. Yan, F. Zhu, and J. Han, "gapprox: Mining frequent approximate patterns from a massive network," *ICDM*, 2007.

[16] X. Yan and J. Han, "gspan: Graph-based substructure pattern mining," *ICDM*, 2002.

[17] N. S. Ketkar, L. B. Holder, and D. J. Cook, "Subdue: Compression-based frequent pattern discovery in graph data," *OSDM*, August 2005.

[18] C. Wang, W. Wang, J. Pei, Y. Zhu, and B. Shi, "Scalable mining of large disk-based graph databases," *KDD*, 2004.

[19] N. Wang, S. Parthasarathy, K.-L. Tan, and A. K. H. Tung, "Csv: Visualizing and mining cohesive subgraph," *SIGMOD*, 2008.

[20] J. Chen, O. R. Zaiane, and R. Goebel, "Detecting communities in social networks using max-min modularity," *SDM*, 2009.

[21] T. Falkowski, A. Barth, and M. Spiliopoulou, "Dengraph: A density-based community detection algorithm," *Web Intelligence*, 2007.

[22] G. Karypis and V. Kumar, "Parallel multilevel k-way partitioning for irregular graphs," *SIAM Review*, vol. 41, no. 2, pp. 278–300, 1999.

[23] N. Shrivastava, A. Majumder, and R. Rastogi, "Mining (social) network graphs to detect random link attacks," *ICDE*, 2008.

[24] S. Papadimitriou and J. Sun, "Disco: Distributed co-clustering with map-reduce," *ICDM*, 2008.