International Journal of Multimedia and Ubiquitous Engineering
Vol. 9, No. 12 (2014), pp. 167-180
http://dx.doi.org/10.14257/ijmue.2014.9.12.16

A Fast Multi-level Layout for Social Network Visualization

Xiaolin Du[1], Yunming Ye[1], Yueping Li[2] and Ge Song[1]

[1]Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen Graduate School, Harbin Institute of Technology, China
[2]ShenZhen Polytechnic, Shenzhen, China
skliic@126.com

Abstract

We describe a fast multi-level layout for visualizing social networks, which can visualize social networks high quality and rapidly. There are two innovations in our fast multi-level layout. Firstly, we proposed a new graph multi-layered compression method based on random walk. The multi-layered compression process groups vertices to form "planet" systems and then abstract these "planet" systems as new vertices to define a new graph and is repeated until the graph size falls below some threshold. And we also proposed a new single level force-directed layout based on sampling. The multi-level layout process can be accelerated based on these two innovations. Finally, we have evaluated our layout on several well-known data sets. The experimental results show that our layout outperforms the state-of-the-art method.

Keywords: *graph compression, random walk, multi-level layout, sampling*

1. Introduction

Social networks appear everywhere in our modern lives, such as twitter, micro-blog, MSN, Facebook, co-citation relation, credit network, *etc.*, The modern science of networks has brought significant advances in our understanding of complex systems [1]. In research, visualization techniques are always employed to illustrate social networks to users and assist social networks analysis. Social networks are usually represented by different types of graphs. Vertices represent entities, and edges represent interactions between pairs of entities. Graph visualization helps users to gain insight into social networks by turning the network elements and their internal relationships into graphs. There have been many graph layout algorithms designed for graph visualization. Each layout algorithm has its own characteristic and pertinence to different types of graph and different applications.

A graph $G=(V_G,E_G)$ is an abstract structure that is used to model a relation E_G over a set V_G of entities. Graph drawing is a conventional tool for the visualization of relational information, and its usefulness depends on its readability, that is, the capability of conveying the meaning of the diagram quickly and clearly. In recent years, many algorithms for drawing graphs automatically were proposed (the state of the art is surveyed comprehensively in [2], [3]).

In this paper, we concentrate on the problem of drawing an undirected graph with straight-line edges. Major advantages of force-directed methods are their relatively simple implementation and their flexibility (heuristic improvements are easily added), but there are some problems with them too. One severe problem is the difficulty of

ISSN: 1975-0080 IJMUE

minimizing the energy function when dealing with large graphs. The other problem is that it is very slow to layout the whole graphs high quality for larger graphs.

We propose a new fast multi-level layout method for drawing graphs that could in principle improve the speed of every force-directed method. We first proposed a new graph multi-layered compression method based on random walk. The multi-layered compression process groups vertices to form "planet" systems and then abstract these "planet" systems as new vertices to define a new graph and is repeated until the graph size falls below some threshold. And we next proposed a new single level force-directed layout based on sampling, which can drastically reduce the computing time of repulsion. Thus the multi-level layout process can be accelerated based on these two innovations.

The rest of paper is organized as follows: Section 2 reviews several areas of related work; Section 3 introduces the detailed description of our fast multi-level graph layout algorithm; Section 4 will evaluate the proposed layout through some comparable study; finally, the paper concludes in Section 5 with a review and discussion of future work.

2. Related Works

2.1. Graph Layouts

Graph visualization helps users to gain insight into data by turning the data elements and their internal relationships into graphs [5]. Graph layout problems are a particular class of combinatorial optimization problems whose goal is to find a linear layout of an input graph in such a way that a certain objective function is optimized [6]. Given a general graph consisting of vertices and edges, graph layout is a problem of drawing the graph. Vertices are assigned coordinates, and if two vertices share an edge it is drawn between them as curves. The popular graph layouts include Node-link layout [7-10], Space filling layout [11, 12], Matrix Layout [13-15] and so on. Node-link layout is one of the most used graph layouts, which uses links between vertices to indicate the relationships of vertices. As one of the well-known Node-link layouts for drawing general graphs, spring layout is proposed by Eades [16] in 1984. Since then, his method is revisited and improved [17-21] in different ways. There are mainly two kinds of space filling layout: space division layout and space nested layout. In space division layouts, the parent-child relationship is indicated by attaching child vertices to the parent vertices. Since the parent-child and sibling relationships are both expressed by adjacency. Space nested layouts, such as Treemaps [22], draw the hierarchical structure in the nested way. They place child vertices within their parent vertices. Matrix Layout is an alternative approach to graph visualization which is using matrix-based representations. Graphs can be presented by their connectivity matrixes. Each row and each column corresponds to a vertex. The glyph at the interaction encodes the edge from corresponding vertex.

2.2. Multi-level Layouts

Multilevel layouts are largely used in graph visualization as multilevel graph drawing methods can accelerate run time and also improve the visual quality of graph drawing algorithms. Chris Walshaw [23] presents a multilevel optimization of the Fruchterman's and Reingold's spring embedder algorithm. The GRIP algorithm [24] coarsens a graph by applying a filtration to the vertices. This filtration is based on shortest path distance. Fast Multiple Multilevel Method (FM^3) [25] is also a force-directed layout algorithm.

FM^3 [25] is proved subquadratic (more precisely in $O(\text{NlogN}+\text{E})$ in time, contrary to previous algorithms. Work in [26] is based on the detection of topological structures in graphs. This algorithm encodes each topological structure by a meta-node to construct a hierarchical graph.

2.3. Our Proposed Method

The proposed fast multi-level layout combines force-directed layout method, graph partition method and graph compression method. More specifically, the fast multi-level layout proposed a graph multi-layered compression method based on random walk and the layout process groups vertices to form “planet” systems and then abstract these “planet” systems as new vertices to define a new graph and is repeated until the graph size falls below some threshold. In each level graph, a new force-directed method based on sampling is employed to assign vertex coordinates. The characteristic of our proposed multi-level layout is visualizing social networks high quality and quickly.

3. Fast Multi-level Layout

In this section, we will introduce the model of Fast Multi-level Layout (FML) detailed. The FML model consists of two steps. The first is fast multi-level graph compress model and the second is the layout initialization and interpolation. In this research, we propose a new graph compress method based on random walk and a new quick method to calculating repulsion based on sampling. Next we will introduce these two steps respectively.

First we give some terminologies that are frequently used in this paper. Throughout this paper we assume that we are working with a graph $G=(V_G,E_G)$, with $|V_G|$ vertices and $|E_G|$ edges. V_G represents the vertex set of G and E_G represents the edge set of G.

3.1. Multi-level Compression

The multi-layered compression in FM^3 [25] is a very complex process. There, solar systems are created, which consist of vertices at a distance of two edges or less from the center of the solar system. It needs more spaces to store the paths between compression units and more complexity to calculate the initial positions of vertices. Walshaw [23] algorithm gives a more global quality to the force-directed placement through compressing vertex with one of its neighbors, but it may lead too many levels, which would complicate the algorithm. Thus, in this paper our compress strategy mixes the advantages of these two classical algorithms. We next detailed introduce the compression process of FML algorithm.

In FML algorithm, we create a “Planet System” on graphs. Some vertices are chosen as “planet” vertices, and then the neighbors of planet vertices are the matching “moon” vertices of the “planet”. A planet and its matching moons construct a planet system. Figure 1 shows an original graph with 17 vertices and 16 edges. Our compress strategy is that we compute the weights of all vertices by Random Walk algorithm, and then compress the vertices with less weight and their uncompressed moons in the graph.

A random walk of length k on a graph G is a stochastic process with random variables $W_0,W_1,W_2,\cdots,W_k$ such that $W_0^T=(1/\text{n},1/\text{n},\cdots,1/\text{n})$ and W_{i+1} is a vertex chosen uniformly at random from the neighbors of W_i. Let B be the column-normalized adjacency matrix of the graph G.

$$W_{k+1}=BW_k \tag{1}$$

By random walk on graph, we get the weights of all vertices. Figure 1 shows an example graph G_E with 17 vertices and 16 edges and Table 1 shows the vertex weights after running random walk on example graph G_E .

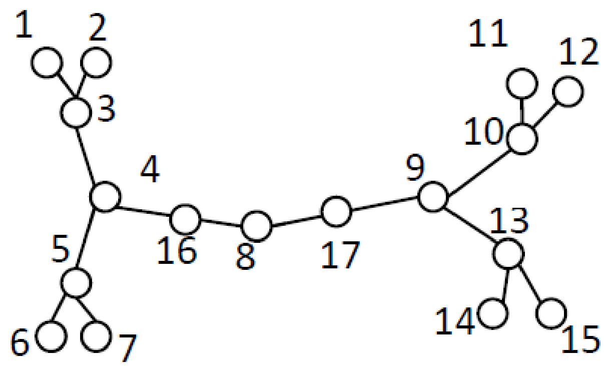


Figure 1. Example Graph

According to the weights in Table 1, we first compress the vertex with minimum weight and its "moons". Then, if there still are vertices uncompressed, we continue choose the rest minimum weighted vertex to compress until all vertices are compressed. Figure 2 demonstrates the compress process. Figure 2 (a) is the compress process and Figure 2 (b) presents the compressed graph.

Table 1. Weights of Vertices After Running Random Walk

Vertex	Weight	Vertex	Weight
14	0.315522	17	0.493981
15	0.315522	13	0.563252
11	0.315522	3	0.563252
12	0.315522	10	0.563252
1	0.315522	5	0.563252
2	0.315522	8	0.737913
6	0.315522	4	1
7	0.315522	9	1
16	0.493981	--	--

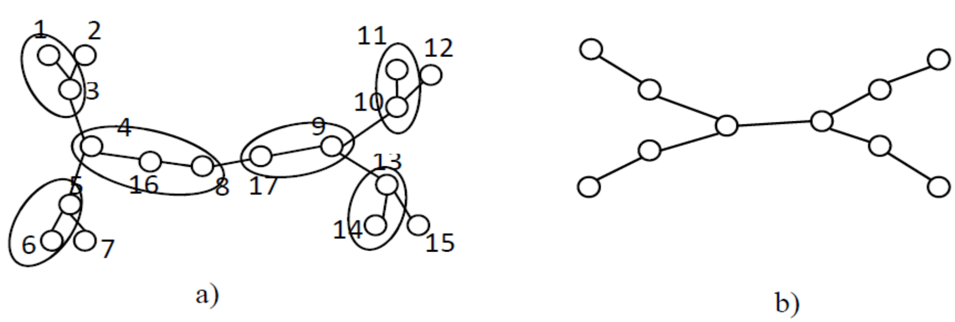


Figure 2. Compress Process

After getting the compressed graph G_1, we continue to compress G_1 to high level graphs. The compressing process is the same as the process described below. We compute the weights of vertices in G_1 and compress the vertices according to their weights. By this analogy, we can get G_2, G_3 until G_L. The termination condition of compression is that the number of vertices in G_L less than our preset maximal vertex count. Then we can obtain the multi-level compression graph just like Figure 3.

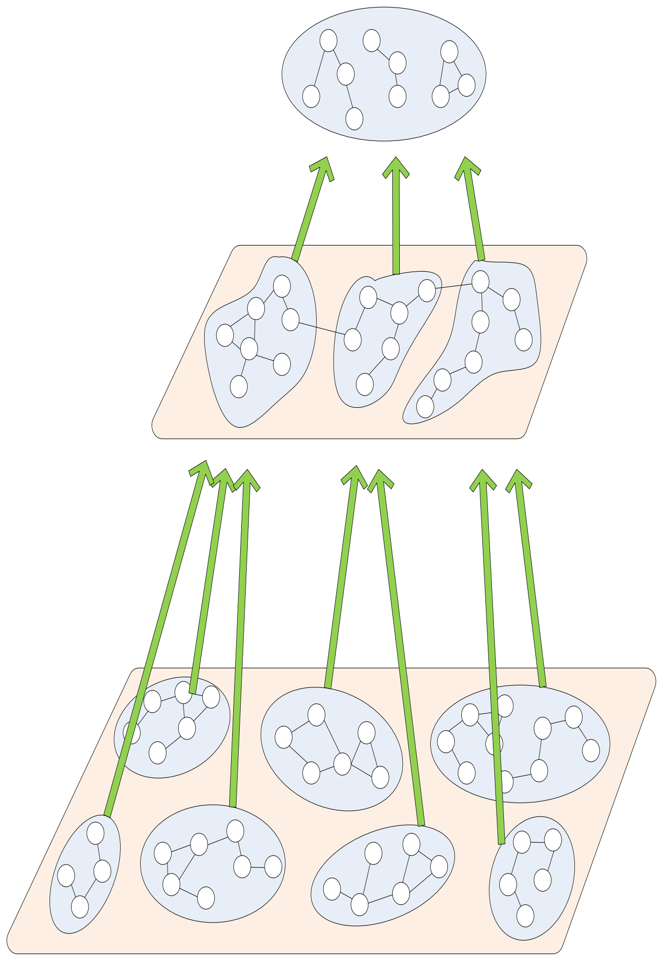


Figure 3. Multi-level Compression

Figure 4 shows the flowchart of multi-level compression of graph.

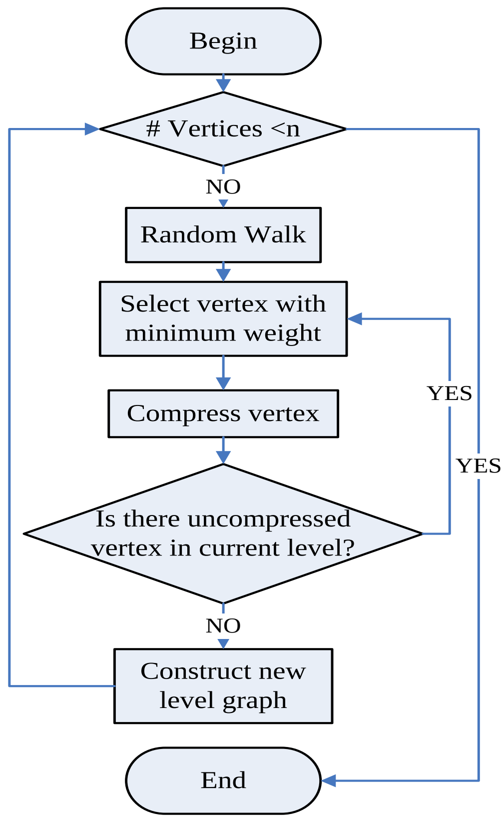


Figure 4. Flowchart of Multi-level Compression

3.2. Layout Initialization and Interpolation

3.2.1. Single Level Force-directed Layout based on Sampling

We apply the force-directed layout algorithm to every level graph. To accelerate the layout process, we propose a new method to compute the repulsion f_r based on sampling.

For a vertex v in G, the repulsion of $f_k(v)$ comes from two parts. v could be partly repelled by its neighbors and partly repelled by the other vertices in the graph. For accelerating repulsion computing, we only consider the repulsion from the neighbors whose length to v is no more than 2, and the repulsion from a random sample of vertices from other part of graph. We compute the repulsion by the following equation.

$$f_k(v) = \sum_{u \in \Gamma(v)} \frac{c_1^2}{d_{uv}} + \sum_{u \in V_S} (C_1 * \frac{L}{l} * \frac{c_1^2}{d_{uv}}) \tag{2}$$

Where $\Gamma(v)$ represents the neighbor set with length to v no more than 2, V_S represents the sampling vertices set, d_{uv} represents the Euclidean distance between u and v, L denotes the number of levels of the compressed graph and l denotes the current level.

We calculate the attraction by the following equation:

$$f_a(v) = \sum_{u \in \Gamma(v)} C_2 * c_2 * {d_{uv}}^2 \tag{3}$$

After much experimenting, we get the best parameter values, where $C_1 = 4$, $c_1 = 0.25 * c$ and $c = \sqrt{W * H / |V|}$ (W and H are the width and height of layout area respectively), $C_2 = 2$ and $c_2 = 0.75$.

3.2.2. Multi-level Initialization and Interpolation

Let $ListG = \{G_0, G_1, \cdots, G_{i-1}, G_i, G_{i+1}, \cdots, G_L, \}, \mathrm{i} = 0,1,2,\cdots,L$ be the compressed graph. G_0, G_i and G_L are original graph, i level graph and top level graph respectively. G_{i+1} is compressed from G_i. In the multi-level initialization and interpolation stages, we first position the vertices in top level graph G_L by single level force-directed layout based on sampling. The refinement process from G_i to G_{i-1} consists of two steps. Frist we initialize the position of vertices in G_{i-1} by the vertices in G_i, and then adjust the positions of vertices in G_{i-1} by single level force-directed layout based on sampling. Iteratively repeat the above process until all vertices in original graph are positioned.

Initializing the position of vertices in G_{i-1} consists of two steps: planet vertices initialization and moon vertices initialization. First, we put the planet vertices in G_{i-1} at the position of its ancestor vertex in G_i and then we position the moon vertices in G_{i-1}. Suppose s_0 and t_0 are two planet vertices, v is a moon vertex between s_0 and t_0. Then the position of v can be calculated by Equation. (4)

$$Pos(v) = Pos(s_0) + \frac{desire\_edgelength(s_0, v)}{length(s_0, t_0)}(Pos(t_0) - Pos(s_0)) \tag{4}$$

If v is contained by more than one path, then the position of v is the centroid of these paths.

$$Pos(v) = \frac{1}{r}\left(\sum\nolimits_{v \in p_i(s_0, t_0)} Pos_i(v)\right) \tag{5}$$

Where $length(s_0, t_0)$ denotes the length of shortest path between s_0 and t_0, $desire\_edgelength(s_0, v)$ denotes the length of shortest path between planet vertex s_0 and moon vertex v.

Figure 5 demonstrates the vertex initialization process of G_{i-1} from G_i. Figure 5 (a) shows i-level graph G_i. Figure 5 (b) shows $(i-1)$-level graph G_{i-1}, vertices with larger radius are planet vertices and vertices with smaller radius are moon vertices. We first position the planet vertices in G_{i-1} on the locations of their ancestor vertices, then calculating the positions of moon vertices by Eq. (4). In Figure 5 (c), suppose u and v are two moon vertices in G_{i-1}, we temporarily put u and v at the circumference with radius 1 of their planet vertex, $length(s_0, t_0)$ is the length of shortest path between s_0 and t_0, $desire\_edgelength(s_0, v)$ is the length of shortest path between planet vertex s_0 and moon vertex v. There is only one planet path $P_{(s_0, t_0)}$ through u, so the position of u is $Pos(s_0) + \frac{1}{2}(Pos(t_0) - Pos(s_0))$. There are two planet path $P_{(s_0, t_0)}$, $P_{(s_0, t_1)}$ through v, so the position of v is $\frac{1}{2}[(Pos(s_0) + \frac{1}{3}(Pos(t_0) - Pos(s_0))) + (Pos(s_0) + \frac{1}{3}(Pos(t_1) - Pos(s_0)))]$, which is shown in Figure 5 (c).

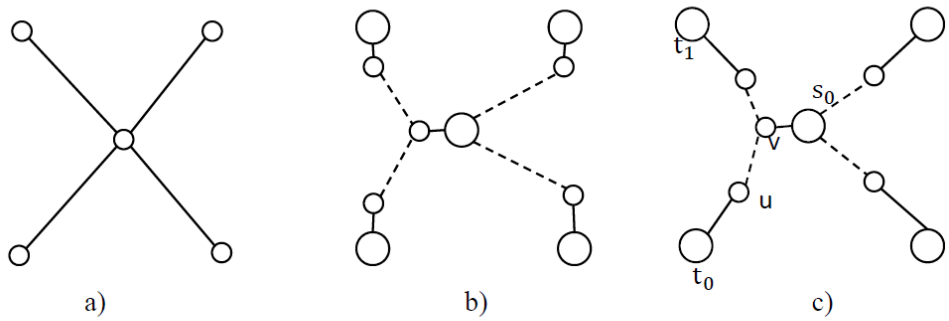


Figure 5. Initializing Process

Figure 6 shows the flowchart of multi-level initialization and interpolation.

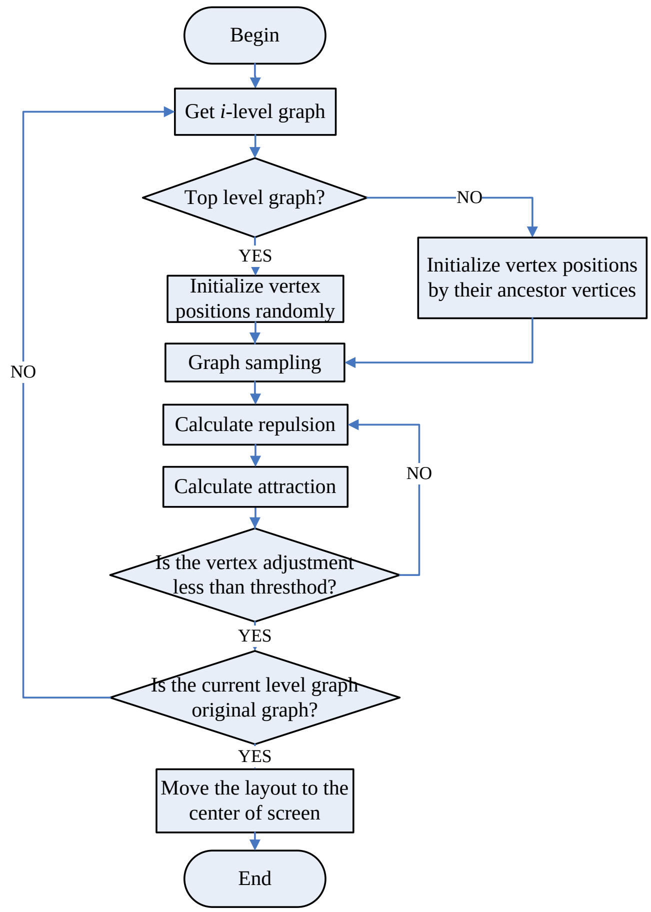


Figure 6. Flowchart of Multi-level Initialization and Interpolation

3.3. Time Complexity Analysis

The time complexity of single level force-directed layout is:

$$O(\mathrm{T}) = \mathrm{O}(\mathrm{k}_0 * (|\mathrm{V}_0|^2 + |\mathrm{E}_0|) + \mathrm{k}_1 * (|\mathrm{V}_1|^2 + |\mathrm{E}_1|) + \cdots + \mathrm{k}_L * (|\mathrm{V}_L|^2 + |\mathrm{E}_L|)) \qquad (6)$$

Our proposed FML method takes $k*(\frac{l}{L}|V|^2+|E|)$ time to run single level force-directed layout based on sampling in every level graph. With the increase of vertices, k will be much less than $|V|$. With the sampling probability p, the number of sampled vertices is $\frac{1}{L}|V_0|$ for 0-level graph, $\frac{2}{L}|V_1|$ for 1-level graph, and $\frac{L}{L}|V_{L-1}|$ for ($L-1$)-level graph. So the time complexity of FML can be represent by the following equation.

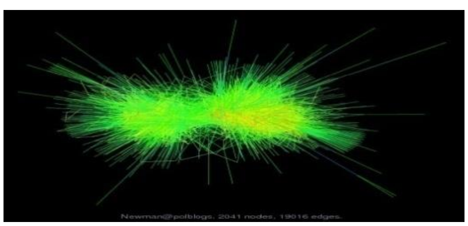
$$O(T)=O(k_0*(\frac{1}{L}|V_0|^2+|E_0|)+k_1*(\frac{2}{L}|V_1|^2+|E_1|)+\cdots+k_L*(|\frac{L}{L}V_{L-1}|^2+|E_{L-1}|)) \quad (7)$$

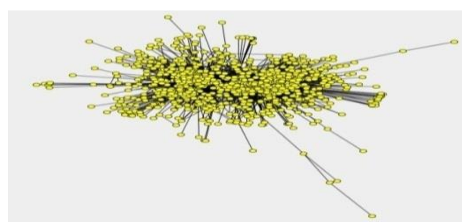
By contrasting Equation (6) and Equation (7) we can find our proposed fast multi-level layout can reduce the calculation of repulsion between vertices and accelerate the process of layout.

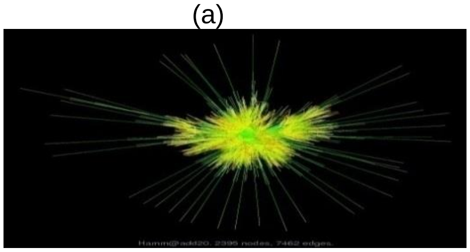
4. Experiments

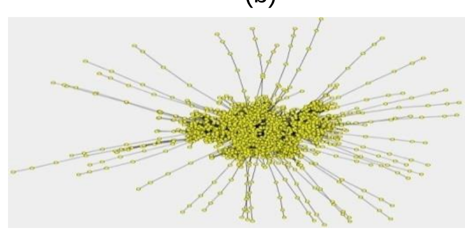
4.1. Experiments 1: Visualization Comparison

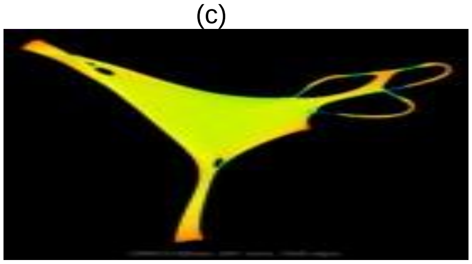
In this section, we evaluate our proposed fast multi-level layout on several graph datasets. For the first experiment, we evaluate the visualization effects of our proposed fast multi-level layout. Figure 7 shows the visualization effects of DIMACS10 test sets and Hamm data sets. Right column are the visualization effects of our fast multi-level layout and left column are the visualization effects published on the internet. By comparing our layout to the published layout effects, we can find our FML can display the real structures of social networks correctly and quickly whatever the density of social networks is uniformity.

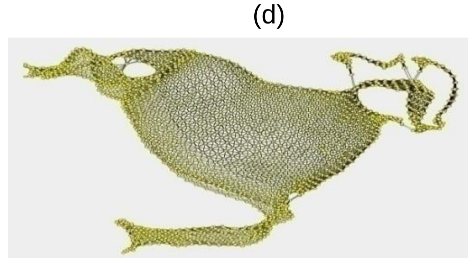

(a)

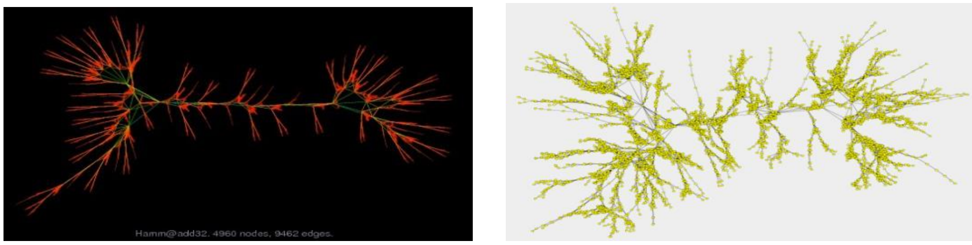

(b)


(c)


(d)


(e)


(f)

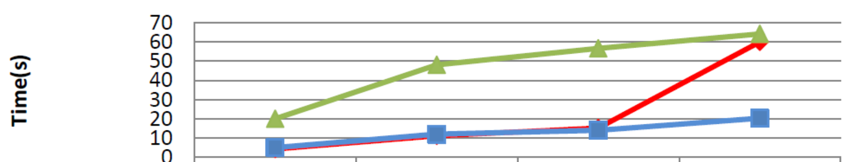


(g) (h)

Figure 7. The Comparison between Layouts

4.2. Experiments 2: Layout Time

The main characteristic of our proposed FML method is layout speed. Thus we evaluate the layout time with other methods on several different scale data sets in this section. There are four test data sets. They are “data” with 2851 vertices and 15093 edges, “add32” with 4690 vertices and 9462 edges, “power” with 4941 vertices and 6594 edges and “whitaker” with 9800 vertices and 28989 edges. Figure 8 shows the layout time for these four data sets by different layout methods. Our FML method takes the shortest time to layout the networks when the number of vertices is less than 5000. As the increasing of vertices, the advantage of FM^3 become prominent gradually. However, for social networks visualization, the size of network should be not very large. Otherwise such a dense layout is a visual disaster for human eyes. So our proposed method has advantages when the number of vertices is less than 5000.



	data(2851 15093)	add32(4960 9462)	power(4941 6594)	whitaker(9800 28989)
FML	4.36	11.17	15.178	60
fmmm	5.05	12.1	14.01	20.3
walshaw	20.05	48.09	56.57	64.1

Figure 8. Layout Time Comparison with Different Methods

Figure 9 shows the run time variation of our FML while the scale of network increasing. There are 7 data sets. They are “subScience” (379 vertices and 914 edges), “polblogs” (1490 vertices and 3430 edges), “add20” (2395 vertices and 7462 edges), “data” (2851 vertices and 15093 edges), “add32” (4960 vertices and 9462 edges), “whitaker” (9800 vertices and 28989 edges) and “4elt” (15606 vertices and 45878 edges). Through Figure 9 we can also observe our FML method can run a good performance when the vertex scale is less than 5000. The layout process can be completed within 20 seconds. Limited by the area and resolution of layout screen, it makes little sense to layout the social networks with vertex scale larger than 10000. Thus our FML method has the important theoretical significance and the practical application value.

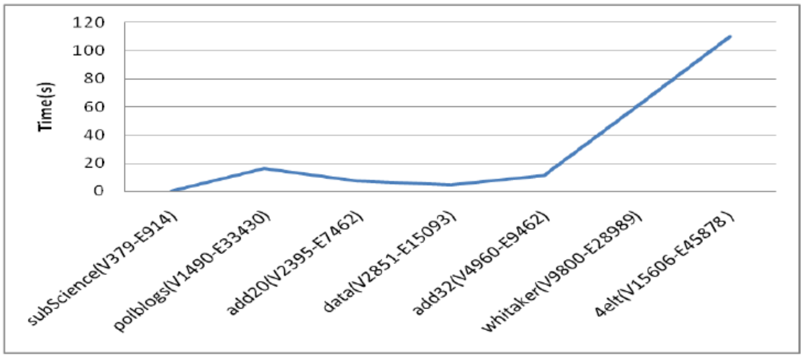


Figure 9. Layout Time Comparison with Different Scale Data Sets

5. Conclusions

In this paper, we firstly proposed a new graph multi-layered compression method based on random walk, and then we presented a new fast multi-level layout based on the compressed graph structure. We detailed described the fast multi-level layout and performed experiments on several datasets. We compared our layout with the FM^3 layout [25] and Walshaw [23] layout from three aspects. The comparable results show our fast layout can visualize the social networks high quality and rapidly.

Acknowledgements

This research was supported in part by Shenzhen Strategic Emerging Industries Program under Grants No.JCYJ20120613135329670, NSFC under Grant no.61100190 and Shenzhen Science and Technology Program under Grant No.CXY201107010163A and No.GJHS20120627112429515.This paper is a revised and expanded version of a paper entitled Social Network Visualization Oriented Multi-level Layout method presented at ACN 2014, Indonesia, June 19-22, 2014.

References

[1] S. Fortunato, “Community detection in graphs,” Physics Reports, vol. 486, **(2010),** pp. 75-174.
[2] M. Kaufmann and D. Wagner, “Drawing Graphs Methods and Models”, Lecture Notes in Computer Science, Springer Verlag, vol. 2025, **(2001)**.
[3] G. Di Battista, P. Eades, R. Tamassia and I. G. Tollis, “Algorithms for the Visualization of Graphs”, Prentice-Hall, **(1999)**.
[4] D. Archambault, T. Munzner and D. Auber, “TopoLayout: Graph layout by topological features”, IEEE Information Visualization Posters Compendium, vol. 3, no. 4, **(2005)**.
[5] W. Cui and H. Qu, “A survey on graph visualization”, Hong Kong University of Science and Technology, **(2007)**.
[6] J. Díaz, J. Petit and M. Serna, “A survey of graph layout problems”, ACM Computing Surveys, vol. 34, no. 3, **(2002)**, pp. 313-356.
[7] Y. F. Hu, “Visualizing Graphs with Node and Edge Labels”, CoRR, **(2009)**.
[8] T. M. J. Fruchterman, “Graph Drawing by Force-directed Placement”, Software- Practice and Experience, vol. 21, no. 11, **(2006)**, pp. 1129-1164.
[9] S. Takahashi and S. Miyashita, “A Constraint-based Approach for Visualization and Animation”, Kluwer Academic Publishers, Constraints, vol. 3, no. 1, **(1998)**, pp. 61-86.
[10] D. Auber, Y. Chiricota and F. Jourdan, “Multiscale Visualization of Small World Networks”, IEEE Conference on Information Visiualization, **(2003)**, pp.75-81.
[11] J. Heer and D. Danahboy, “Vizster: Visualizing Online Social Network”, IEEE Symposium on Information Visualization, **(2005)**, pp. 32-39.

[12] D. Archambault and T. Munzner, "Smashing Peacocks Further: Drawing Quasi-Treeform Biconnected Components", Visualization and Computer Graphics, vol. 12, no. 5, **(2006)**, pp. 813-820.
[13] N. Elmqvist and T. N. Do, "Interactive Large-Scale Graph Visualization", In Proceedings of IEEE Pacific Visualization Symposium, **(2008),** pp. 215–222.
[14] N. Henry and J. D. Fekete, "Matrix Explorer: A Dual-representation System to Explore Social Networks", IEEE Transactions on Visualization and Computer Graphics, vol. 12, no. 5, **(2006)**, pp. 677-684.
[15] N. Henry and J. D. Fekete, "MatLink: Enhanced Matrix Visualization for Analyzing Social Networks", Springer Berlin Heidelberg. Human-computer interaction. Berlin: Springer Berlin Heidelberg, **(2007)**, pp. 288-302.
[16] P. A. Eades, "A heuristic for graph drawing", In Congressus Numerantium, vol. 42, **(1984)**, pp. 149–160.
[17] T. M. J. Fruchterman and E. M. Reingold, "Graph Drawing by Force-directed Placement", Software- Practice and Experience, vol. 21, no. 11, **(1991)**, pp. 1129–1164.
[18] E. R. Gansner and S. C. North, "Improved force-directed layouts", Proceedings of the Graph Drawing Symposium 1998, **(1998),** pp. 364–373.
[19] P. Eades and M. L. Huang, "Navigating Clustered Graphs using Force-Directed Methods", Journal of Graph Algorithms and Applications, vol. 4, no. 3, **(2000)**, pp. 157–181.
[20] P. Gajer, M. T. Goodrich and S. G. Kobourov, "A multi-dimensional approach to force-directed layouts of large graphs", Computational Geometry: Theory and Applications, vol. 29, no. 1, **(2004)**, pp. 3–18.
[21] T. Dwyer, K. Marriott and M. Wybrow, "Integrating Edge Routing into Force- Directed Layout, **(2007)**.
[22] B. Johnson and B. Shneiderman, "Tree-Maps: a space-filling approach to the visualization of hierarchical information structures", Readings in information visualization: using vision to think table of contents, **(1991),** pp. 152–159.
[23] C. Walshaw, "A multilevel algorithm for force-directed graph drawing", Graph Drawing, Springer Berlin Heidelberg, **(2001)**, pp. 171-182.
[24] P. Gajer and S. G. Kobourov, "Grip: Graph drawing with intelligent placement", JGAA, vol. 6, **(2000),** pp. 2002.
[25] I. Benacer and Z. Dibi, "Modeling and Simulation of Organic Field Effect Transistor (OFET) Using Artificial Neural Networks", International Journal of Advanced Science and Technology, vol. 66, **(2014).**
[26] D. Archambault and D. Auber, "Topolayout: Multilevel graph layout by topological features", IEEE Transactions on Visualization and Computer Graphics, vol. 13, no. 2, **(2007),** pp. 305–317.
[27] M. E. J. Newman, "Detecting community structure in networks", The European Physical Journal B-Condensed Matter and Complex Systems, vol. 38, no. 2, **(2004)**, pp. 321-330.
[28] M. Girvan and M. E. J. Newman, Proc. Natl. Acad. Sci. USA 99, **(2002),** pp. 7821-7826.

Authors

Xiaolin Du, she was born in Heilongjiang Province, China in 1983, and received her Master Degree in Computer Science from Harbin Institute of Technology in 2009.

Currently, she is a PhD candidate in Shenzhen Graduate School, Harbin Institute of Technology. Her research interests involve data mining, data visualization and social network discovering.

Yunming Ye, he was born in China in Sep. 1976, and received his PhD degree in Computer Science from Shanghai Jiao Tong University in 2004.

Currently, he is a professor in Shenzhen Graduate School, Harbin Institute of Technology. His research interests include Web mining, Web Search, and social computing.

Yueping Li, he was born in Guangdong Province, China in Sep. 1980, and received his PhD in Computer Science from Sun Yat-sen University in 2008.

Currently, he is a lecturer in Shenzhen Polytechnic. His research interests involve web mining, graph algorithm and optimization.

Ge Song, she was born in Henan Province, China, in 1985.

Currently, she is a PhD candidate in Shenzhen Graduate School, Harbin Institute of Technology. Her research interests involve data mining and text an analysis.