Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 154 (2019) 33–39

Procedia Computer Science

www.elsevier.com/locate/procedia

8th International Congress of Information and Communication Technology, ICICT 2019

Optimization Algorithms Study and Implementation on Graph Drawing Based on XML Document

Hai Tao Wu[a**], Li Hong Guo[a,b], Mei Juan Wang[b], Jie Yang[a]

[a]*Dept. of Communications Engineering, Nanjing Institute of Technology, Nanjing, 211167, China*
[b]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, China,*

Abstract

In the process of engineering applications, we often use a large number of vertices and edges to represent the complex system model, but sometimes the model is so complex that it is impossible to draw the graph neatly and beautifully by hand. So how to draw the graph automatically using the computer and how to make the graph visual become an important research topic. Based on the study on the genetic algorithm, SHLA algorithm and FR algorithm, we implement these algorithms by analyzing the pseudo-codes. Especially we focus on the study of the source programming about FR-boost and then implement an improved FR algorithm that selects XML document as the input and output graph files. It can effectively reduce the complexity of algorithm and increase the point-line graph's visualization.

Keywords: Graph Drawing; Algorithms; Optimization; Visualization; XML

---

1. Introduction

With the development of Internet, graphical data is in everywhere. In the face of these massive graphic data, we often use the statistical analysis and data mining methods to extract the useful information. However most extracted result is unreadable, we have a strong demand of graph visualization. The study of algorithms for graph drawing and graph visualization has a long history, until now all kinds of new algorithms are emerging in succession. They are

---

[*] * Corresponding author. Tel.: +(86)13952090180
Email: guolihongnj@163.com

10.1016/j.procs.2019.06.007

widely used in network management, program visualization, software engineering and the computer aided teaching. So it is very important to study the graph visualization and the design algorithm.

In this paper, the layout algorithm of two-dimensional point-line graph is studied. Based on the study and analysis of pseudo-codes, we program and implement these algorithms, at the same time the visualization test graphs are provided. Further, an improved FR optimization algorithm is designed and programmed to increase the visualization degree of point-line graph.

The paper's main contributions are summarized as follows:

The study on graph's visualization. These algorithms include the genetic algorithm [1] and the hybrid layout algorithm of skeleton sub graph [2] and so on.

To implement these algorithms and make the graph neatly and beautifully, some parts of their pseudo-codes are modified or rewritten.

Based on the analysis of source programming, we implement some graph's visualization algorithms and provide their visualization test graph.

The rest of this paper is organized as follows. Some studies on graph's visualization and algorithms are described in section 2. Section 3 presents the detailed realization process and visualization display. In section 4, we focus on the FR algorithm. In section 5, the improved FR algorithm is modified and designed, which can effectively reduce the complexity of algorithm and increase the graph's visualization. The last section we conclude these algorithms.

2. Related Work

There are many branches in the process of studying on graph's visualization. These studies can be divided into two categories: the branch of physics and the branch of biology. In the branch of physics, the most famous algorithm is Force-Directed Algorithm (FDA) and all kinds of improved algorithms **[3]**. For example, in 1984, P.Eades proposed Spring-Embedded Model **[4]**. In 1989, T. Kamada and S. Kawai proposed KK algorithm **[5]**. In 1991, T. M. J. Fruchterman and E. M. Reingold proposed FR algorithm **[6]**. In this paper, we mainly learn the layout algorithm of two-dimensional point-line graph. These algorithms include the genetic algorithm and the hybrid layout algorithm of skeleton sub graph and FR algorithm.

2.1. Genetic Algorithm

The genetic algorithm is a kind of search heuristic algorithm, it is a good solution and usually be used to solve some optimization and search problems. In this paper, the essence of genetic algorithm is to imitate the process of biological evolution, and then makes it apply to layout algorithm.

This typical genetic algorithm must satisfy two conditions. One is the genetic representation of the solution domain. And another is the fitness function, its aim is to evaluate the solution domain. The detailed genetic algorithm is followed below several steps: (1) Designing the objective function; (2) Encoding the chromosome; (3) Designing the fitness function; (4) Operating the genetic operators; (5) Controlling the parameters.

2.2. Skeleton-based Hybrid Layout Algorithm (SHLA)

With the development of research and the increase in the number of nodes in the graph, the above-mentioned algorithms can't meet the requirement, hence the idea of hybrid layout comes into the point-line algorithm. SHLA algorithm's main idea is to separate the source graph into a skeleton sub graph and several stub trees, and at last to layout them with different graph drawing algorithms **[7]**.

In the process of designing SHLA algorithm, the aesthetic factors should be considered:

(1) Distinguish the skeleton sub graph and stub trees. In view of the skeleton sub graph and stub trees, different layout algorithms have been used. So their layout style and layout area have significant differences, which will help the user directly identify the skeleton sub graph and stub trees in the graph.

(2) As far as possible to make the nodes' distribution evenly. By the means of calculating stub trees' layout area before layout, which can achieve the overall uniform

The drawback of this algorithm lies in: the skeleton does not necessarily reflect the relationships of core nodes in the graph, which will result in the higher overlap and poor satisfaction.

3. The Implement of Algorithms

In order to implement the graph's visualization, we study all kinds of visualization algorithms, and based on the boost library, we program and rewrite these algorithms. In the process of implement algorithms, all of them were designed and accomplished by visual C++ 6.0, further these algorithms were carried out on the Intel (R) Core (TM) 2 Duo, 2.93GHz CPU, memory 2G. As an operating system, Windows 7 was used in the experiments. Further, in all of these experiments, XML documents were chosen as the input and output graph files to finish all the test.

3.1. Genetic Algorithm(GA)

In order to make genetic algorithm apply in the layout of graph, we follow the next steps of genetic algorithm. In the process of real programming, we make the following choice:

(1) Determine the objective function: Here we choose the equation (1) in the literature **[9]** as the objective function, p_i and p_j are the coordinate values of node *i* and node *j* respectively. *L* is the ideal side length.

$$f = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \frac{L^2}{|p_i - p_j|} + \sum_{(v_i, v_j) \in E} \frac{|p_i - p_j|^2}{L} \tag{1}$$

(2) The coding of chromosome: For each possible solution (x_1, y_1) $(x_2, y_2)\ldots(x_n, y_n)$, we use a decimal string of length 2n $(x_1, y_1, x_2, y_2, x_3, y_3 \cdots\cdots x_n, y_n)$ to express it.

(3) Choose the fitness function: linear transformation, power function transformation, the exponential transform and so on. Here we adopt the objective function as the fitness function.

(4) Operating the genetic operators: includes selection, crossover and mutation.

(5) Settings of key parameters: the initial population size, the probability of crossover and mutation, and so on, in this test, the settings of key parameter are shown in Table 1.

Table 1 the settings of key parameters

Key Parameter	*Value*
size of chromosome population	30
iteration number of population evolutionary	60
the probability of crossover	0.75
the probability of mutation	0.0015

Based on these choose and settings, we get its test graph. From the test display, we can conclude that the nodes layout is evenly in the graph. The test graph is shows in **Fig.1** and **Fig.2**. Fig.1 is the GA test graph with few nodes, and Fig.2 is the GA test graph with many nodes.

3.2. Skeleton-based hybrid layout algorithm(SHLA)

In SHLA, there are two sub algorithms are included in this algorithm: the layout algorithm of the skeleton sub graph *KK_EX_*Layout and the layout algorithm of stub trees *Stub_Tree_Layout*. Based on the literature **[8]** and **[9]**, we program to implement these algorithms. In the process of programming, we found that there were some problems in the algorithms:

(1) In 'skeleton_identify(V)', it hasn't considered the solution in some special cases. For example, in this case the input graph is a graph without circle;

(2) In **[8]**, improved K_K algorithm is to adopt different algorithm for the layout of stub tree, but its skeleton sub graph is still the K_K algorithm;

(3) During the layout of stub trees, the layer of stub tree seriously underestimated. So when the number of nodes is more and more larger and the layer of stub tree is much more, the algorithm is not efficient;

(4) In **[9]**, it assumes that the radius of every layer in stub tree is same, which affects the beauty of layout.

For above-mentioned problems, we merge the layout algorithm of the skeleton sub graph and the layout algorithm of stub trees. The test graph is shows in **Fig.3** and **Fig.4**. Fig.3 shows the SHLA test graph with few test nodes, and Fig.4 is the SHLA test graph with many test nodes. From these test figures, we can know that this algorithm makes the overlap degree high between point and point in the graph.

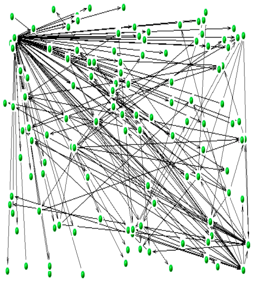


Fig.1. GA test graph (few）

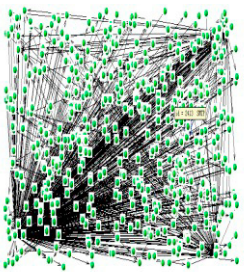


Fig.2. GA test graph (many）

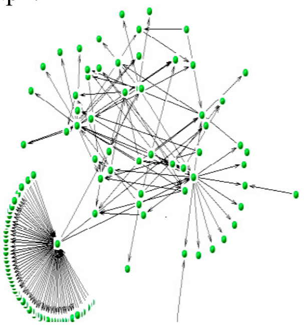


Fig.3. SHLA test graph (few）

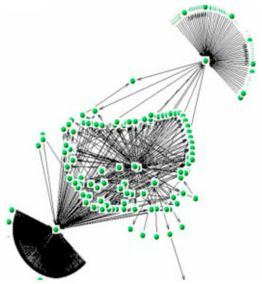


Fig.4. SHLA test graph (many）

4. FR Algorithm

In 1984, a 2D layouts algorithm was proposed by Eades **[4]**, which used a mechanical model to create 2D layouts, this algorithm can make the targeted graphs up to 30 vertices. In this algorithm, it uses the steel rings to replace the vertices and uses a spring to replace each edge to form a mechanical system. In the initial layout, many vertices are placed well and then let them go, as times go on, all the spring forces on the rings reach to a minimal energy state. In the process of it, two practical adjustments need to make:

Using logarithmic strength springs; that is, the force exerted by a spring is showed in formula (2). In this formula, parameter d represents the length of the spring, and parameter c1 and c2 are two constants.

$$c1 * \log(d / c2) \tag{2}$$

Making nonadjacent vertices repel each other. An inverse square law force formula (3) is showed as below:

$$c3 / d^2 \tag{3}$$

In formula (3), parameter c3 is a constant and parameter d represents the distance between the vertices. Experience shows that for most graphs the parameter values c1 = 2, c2 = 1 and c3 = 1 are appropriate in formula (2) and (3). When the simulation step runs about 100 times almost all the graphs can achieve a minimal energy state.

In 1989, a kind of graph layout model was proposed by Kamada and Kawai **[10]**. In this model, they regard the desirable geometric distance between two vertices as the graph theoretic distance in the corresponding graph.

In this model, the ‘perfect’ drawing of a graph would be one where the pair-wise geometric distances between the drawn vertices match the graph theoretic pair-wise distances, as computed by an All-Pairs-Shortest-Path computation; Secondly, if a pair of vertices is geometrically closer than their corresponding graph distance the vertices repel each other; Thirdly, by the equation (4), let parameter $d_{i,j}$ represent the shortest path distance between vertex *i* and vertex *j* in the graph. Then $l_{i,j} = L \times d_{i,j}$ is the ideal length of a spring between vertices *i* and *j*, where L is the desirable length of a single edge in the display. The strength of the spring between vertices *i* and *j* is defined as equation (4), where K is a constant.

$$k_{i,j} = K / d^{2}_{i,j} \tag{4}$$

Finally, furthermore treating the drawing problem as localizing |V | = n particles $p_1, p_2, \ldots, p_n$ in 2D Euclidean space, the coordinates of a particle p_i in the 2D Euclidean plane are given by x_i and y_i, which allows to rewrite the following overall energy equation as follows (5):

$$E = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{2} k_{i,j}(|p_i - p_j| - l_{i,j})^2 \tag{5}$$

4.1. The Physics Principle of FR Algorithm

In 1991, another graph-drawing algorithm named FR algorithm was proposed by T.M.J.Fruchterman and E.M.Reingold **[6]**, which is also based on the Spring-Embedded Model. In this model, each node corresponds to an atomic particle in the graph. Among all the different particles contain repulsive forces, and between the edges joining nodes contain attractive forces. From the beginning, all the atomic particles are in disordered state. After the interaction of attractive forces and repulsive forces, all the atomic particles will reach an equilibrium state. At this time, the atomic position also is the layout to each node in the graph.

The definition of attractive forces and repulsive forces are shown in formula (6) and (7). Here variable x is the distance between two nodes and k is the optimal distance between two nodes.

$$f_a(x) = x^2 / k; \tag{6}$$

$$f_r(x) = -k^2 / x; \tag{7}$$

The calculation formula about optimal distance k is shown in formula (8). Here variable area is the frame's area and *c* is a constant, we can obtain the constant c from the experiment data.

$$k = c\sqrt{\frac{area}{number\ \ of\ \ vertices}} \tag{8}$$

4.2. The Detailed Process of FR-boost Algorithm

In this algorithm, XML document is used to describe the input and output graph files **[11]**. The example about input file is shown in **Fig.5**. The Fruchterman_Reingold_layout (FR_layout) source programming is in the BOOST library. This algorithm follows the two principles: (1) any two vertices should not be too close. (2) there is an edge between vertices to be close to each other.

```
<?xml version="1.0" encoding="UTF-8"?>
<nexusml>
      <nexus rootnode="49805">
        <node id="49805" remark=""/>
        <node id="49806" remark=""/>
        ……
        <edge id="12347to50776" source="12347" target="50776" direction="1" remark=""edgetype=""/>
        <edge id="49805to12345" source="49805" target="12345" direction="1"  remark=""edgetype=""/>
        <edge id="12345to50444" source="12345" target="50444" direction="1"  remark=""edgetype=""/>   …
    </nexus>
</nexusml>
```

Fig.5. XML Document Format of input file

Using the test data, the graph with FR algorithm can be obtained. The graph is shown in **Fig.6** and **Fig. 7**. Fig.6 is the graph that the test nodes are few, and Fig.7 is the graph that its test nodes are more than Fig.6. From the test figures, we can conclude that the distribution of nodes is too uniformly, so the graph's appearance is poor.

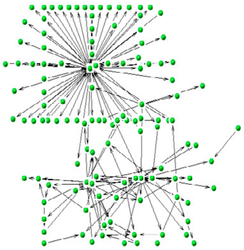
4.3. Improved FR Algorithm

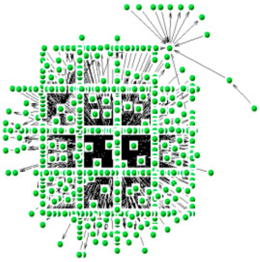
Based on above-mentioned FR algorithm, we modify the calculation formula (6) and (7). The core idea is to increase the attractive forces between nodes and decrease the repulsive forces between nodes. The detailed formula is shown in formula (9) and (10). Here variable x is the distance between two nodes, P_a and P_r are the setting parameters. We can get them according to the actual test data. In general, P_a is greater than 10, P_r is greater than 8.

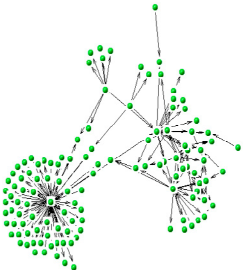
$$f_a(x) = x^2 / k * P_a \tag{9}$$

$$f_r(x) = -k^2 / x / p_r \tag{10}$$

Setting the vertex number of input graph is *n*, the edge number is *m* and the iteration number is *w*. Based on the FR code we can calculate the complexity of this algorithm is about: O (n^2). By applying the improved FR algorithm and using the same test data, we obtain the output graph in **Fig.8** and **Fig.9.** From Fig.8 and Fig.9, we can conclude that the improved FR algorithm has better clustering characters. The graph shows that it has high appearance and is neat and beautiful.







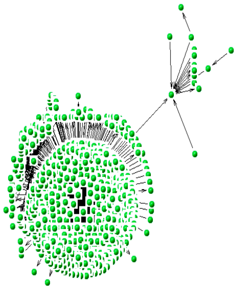


Fig.6 FR test graph (few) **Fig.7.** FR test graph (many) **Fig.8**. Improved FR test graph (few) **Fig.9**. Improved FR test graph (many)

5. Conclusions

In this paper, we first studied all kinds of algorithms on two-dimensional point-line graph drawing. By analyzing their pseudo-codes, we design, modify and implement these algorithms. At the same time, taking the XML document as the input and output files, which effectively save the information about the vertices and edges in the graph. Especially we focus on analyzing the FR algorithm in the BOOST library, from the experiment result, we can conclude this algorithm has good clustering characters. The study of graph's visualization has a long history, its intuition and simplicity make the researchers have great interest in different fields. Especially dynamic graphs' visualization has great application area, so graph's visualization has great prospects for development and be worth researching deeply **[12].**

Acknowledgements

This work was supported by the Scientific Research Funding of Nanjing Institute of Technology (No. ZKJ201612), the Innovation Funding of Nanjing Institute of Technology (No. CKJB201704), the nature science foundation of Jiangsu province (No. BK20160781) and national natural science foundation of China (No.61701221).

References

1. R. Leardi, Genetic Algorithms, *Comprehensive Chemometrics*, vol. 1, pp. 631-653, 2009.
2. Weiming Zhang, Kai Zhang,Qingxian Wang, Hybrid Layout Algorithm Based on Skeleton Sub graph, Journal of Computer Application, 2018, 28:378-381
3. Chun-Cheng Lin, Hsu-Chun Yen, A New Force-Directed Graph Drawing Method Based on Edge-edge Repulsion, Journal of Visual Languages & Computing, 2012,23:29-42
4. P. Eades, A Heuristic for Graph Drawing, In Congressus Numerantium, 1984,42:149-160

5. T. Kamada and S. Kawai, An Algorithm for Drawing General Undirected Graphs, In Information Proc. Letters, 1989,31:7-15
6. T. M. J. Fruchterman and E. M. Reingold, Graph Drawing by Force-directed Placement, Software Practice and Experience, 1991,21:1129-1164
7. Weiming Zhang, Kai Zhang, Qingxian Wang, Power Law Graph Clustering Algorithm based on Skeleton Subgraph, Application Research of Computers, 2008, 25:2610-2612
8. David Soen-Mun Chan1, Khim Shiong Chua, Christopher Leckie and Ajeet Parhar, "Visualization of Power-Law Network Topologies", IEEE International Conference on Networks - ICON , pp. 69-74, 2003.
9. Jingwei Huang, Lisan Tang, Yuping Chen, "A New Graph Drawing Algorithm for Undirected Graphs", Journal of Software, Vol. 11, pp. 138-196, 2000.
10. T. Kamada, S. Kawai, An Algorithm for Drawing General Undirected Graphs, Information Process Letter, 1989,31:7–15
11. W. Borutzky, BGML-A Novel XML Format for the Exchange and the Reuse of Bond Graph Models of Engineering Systems, Simulation Modeling Practice and Theory, 2006,14:787-808
12. Kasyanov, V. N., Kasyanova, E. V, Information visualisation based on graph models , Enterprise Information Systems, 2013,7:187-197