2005 International Symposium on Nonlinear
Theory and its Applications (NOLTA2005)
Bruges, Belgium, October 18-21, 2005

Dynamical Visualization Method for Sparse Matrix with Two Level Calculation

Kaname KUROKAWA† , Nao OHASHI† , Masayuki YAMAUCHI† and Mamoru TANAKA†

†Department of Electrical and Electronics Engineering, Sophia University,
7-1, Kioi-cho, Chiyoda-ku, Tokyo 102-8554, Japan
Email: kaname@mamoru.ee.sophia.ac.jp

Abstract—A directed graph is constructed by a lot of nodes and branches. The graph is used to analyze information on various fields. When a data set is large and complex, the data set is entailed to be visualized. We consider improvement of the visualization method by which a directed graph is arranged automatically. In this paper, we propose a new method which achieves shortening of computation time. In this method, the number of brunches of each node is considered and a directed graph is constructed twice.

1. Introduction

Recently, the situation in which huge information should be processed has increased. A graph is one of the means to prevent misunderstanding of information. This is used in representing a structure of website, a result of data mining, a work flow and so on. It can represent relation of items. When the sparse connection which shows the relevance among attributes is given, this connection is expressed by the corresponding sparse matrix and it is displayed as a directed graph which consists of nodes and edges. Nodes mean attributes and edges mean relations respectively. The number of branches of each node is called "degree." Figure 1 shows the transformation from a sparse connection matrix to the corresponding directed graph which is visualized.

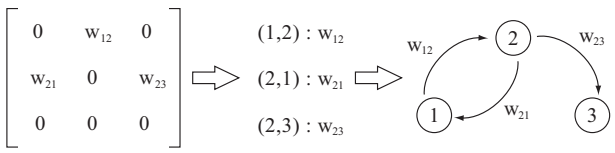

Figure 1: The transformation for visualization.

Though a graph is an effective method to represent relation of items, the graph only represents the structure. When it is large and complex, visualization is needed to understand easily. In general, dynamical model is used to visualize a graph. If two nodes have an edge, the spring's force works. If two nodes don't have an edge, Coulomb's force works. Appropriate positions of each node are given by solving equations of motion.

In this paper, we propose a new method which achieves shortening of computation time. Our conventional method executes wasteful calculations. Therefore, in this model, as the number of data increases, it takes much time to converge. In the proposed method, the value of degree is considered and a directed graph is constructed twice. This method provides a shorter computation time.

2. Dynamical Method

2.1. Dynamical Model

Our model which is to decrease intersections is sum of the spring's force and the Coulomb's force.

If two nodes have an edge then the spring's force works. In the spring model of the graph, a node and an edge are transposed to an iron ring and a spring which forms a dynamics system respectively. An attraction and a repulsive force of each spring works between the two connected nodes. The force is defined by

$$f_s = c_s log \frac{d}{d_0} \tag{1}$$

where c_s is the constant parameter, d is the length of the current spring, d_0 is the original length of the spring. The attraction works for $d > d_0$, the repulsive force works for $d < d_0$, and the force does not work for $d = d_0$

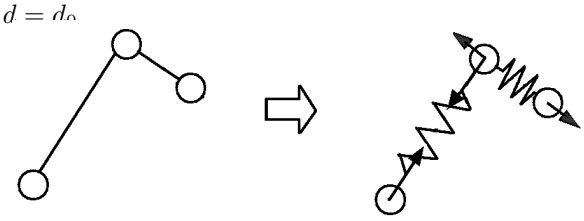

Figure 2: The model of spring's force.

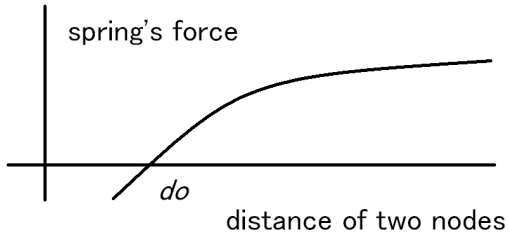

Figure 3: The graph of spring's force.

If two nodes don't have an edge, Coulomb's force works. Two nodes are transposed to be an electron with the same charge. The Coulomb's force works between unconnected nodes. This force is only repulsive one. The repulsive forces work for all pair of nodes which are not connected. The force is defined by

$$f_c = \frac{c_c}{d^2} \tag{2}$$

where c_c is the constant parameter, d is the distance of two nodes.

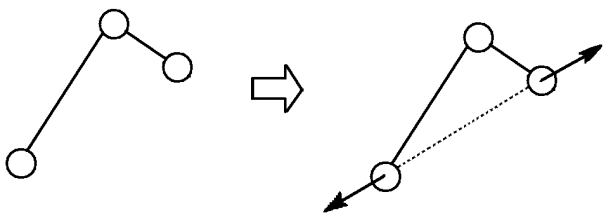

Figure 4: The model of Coulomb's force.

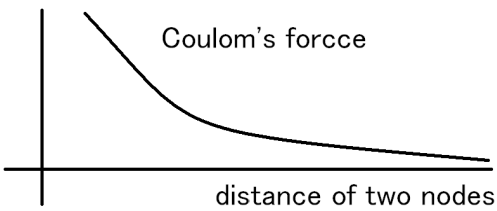

Figure 5: The graph of spring's force.

The efficient arrangement of the graph is obtained by finding the stable point.

2.2. Simulated Annealing

In a certain graph, when the dynamical solution has at least an intersection of two edges as shown in the left side of Figure 6, the intersection must be removed as shown in the right side of the desirable graph in Figure 6.

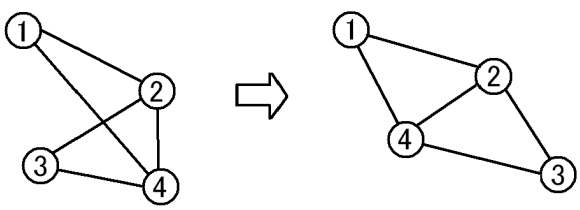

Figure 6: The removal of an intersection.

Since the local minimum solution is in a stable state, the trajectory must be slipped out of the local state. Then, it is important to adjust the energy for the required convergence. Therefore, we introduce a kind of annealing method. The simulated annealing is general optimization to slip out the local state. The simulated annealing parameter δ of the algorithm is used to determine the unit width of convergence. By enlarging δ, it is expected that the directed graph is slipped out of a local solution. However, since it is difficult to converge the algorithm while the parameter δ is large, the parameter δ must be small gradually. That is, the graph state converges by giving the simulated annealing.

2.3. Algorithm

The basic algorithm of the proposed dynamical method is given as follows:

Each node is on the initial state;
(It repeats until all nodes are stabilized)
{
  for v = 1 to (the number of all nodes)
  {
    for w = 1 to (the number of all nodes)
    {
      if(node v and w adjacent to edge e)
      then
        The force f_s into or from node v adjacent to the edge e is calculated;
        $f_v = f_v + f_s$;
      else
        The force f_c to the inverse direction from node v for the direction to the non-incident node w is calculated;
        $f_v = f_v + f_c$;
      }
    }
  The movement δf_v of node v is done;
  The simulated annealing parameter δ is decreased;
  $\delta = \delta - a$; // a×(the number of all loops) $< \delta$
}
Each node is moved on the display according to the force. The above algorithm converges from the initial state to the final stable state.

2.4. Setting of Parameter According to Number of Data

As the number of data changes, the value of the parameter should change. It is very important to set the parameter to an appropriate value. We have noticed values of appropriate parameters while repeating the experiment many times. The parameters are set as follows:

- In the spring model,

$$d_0 = d_1 - k_d N \ \ (spring's\ force : f_s = c_s log\frac{d}{d_0}) \tag{3}$$

where d_1, c_s and k_d are constant, N is number of nodes and d is the distance of two nodes. d_0 is the original length of the spring.

- In the Coulomb model,

$$c_c = \frac{c_n}{N} \quad (Coulomb's\ force : f_c = \frac{c_c}{d^2}) \qquad (4)$$

where c_n is constant, N is number of nodes and d is the distance of two nodes. c_c is a value that decides the ratio of force to work unconnected nodes. When c_c is small, the distance of unconnected nodes becomes small and all nodes are not distributed well. On the other hand, when c_c is large, nodes gather in the edge of the display.

- In the simulated annealing,

$$\delta = \delta_0 - k_\delta N \quad (the\ parameter\ \delta : constant) \qquad (5)$$

where δ_0 and k_δ is constant, N is number of nodes. δ is a parameter that adjusts force to give to an individual node.

3. Essential of proposed method

Our conventional method has a useless calculation. In this model, one node influences the other nodes and is influenced from the other nodes. All nodes are calculated at the same time. However, it is clear that the node whose degree is one approaches another side. In the proposed method, a directed graph is constructed twice. The algorithm is as follows:

1. The degree of all nodes is counted.
2. Nodes whose degree is one are disregarded.
3. By the dynamical model, a graph is constructed from a data set without the nodes.
4. The nodes are given very strong spring and added to the data set of the result of the number 3.
5. A conclusive graph is constructed.

4. Experiments

We verify whether the theory is right or not and use sparse matrices generated at random from 10×10 matrix to 100×100 matrix by the difference of 10 matrices between them. The number of data is 50 in all. The resolution of the display is 1280×1024 pixel. We compare the conventional method and the proposed method, measured response time, counted the number of intersections and measured length of edges by using three parameter sets in Pentium4 Processor at 3.00GHz.

Parameters which are constant according to number of data are as follows:

$c_s : 3.0 \times 10,\ d_0 : 6.0 \times 10$
$c_c : 6.0 \times 10^5$
$\delta_0 : 9.0 \times 10^{-1},\ a : 5.0 \times 10^{-3}$

Parameters which are variable according to number of data are as follows:

$c_s : 3.0 \times 10,\ d_1 : 7.0 \times 10,\ k_d : 5.0 \times 10^{-1}$
$c_n : 1.2 \times 10^7$
$\delta_0 : 1.0,\ k_\delta : 4.0 \times 10^{-3},\ a : 5.0 \times 10^{-3}$

Parameters of the proposed method are as follows:

$c_s : 5.0 \times 10,\ d_1 : 1.0 \times 10^2,\ k_d : 5.0 \times 10^{-1}$
$c_n : 1.6 \times 10^7$
$\delta_0 : 5.0 \times 10^{-1},\ k_\delta : 1.0 \times 10^{-3},\ a : 5.0 \times 10^{-3}$

4.1. Results

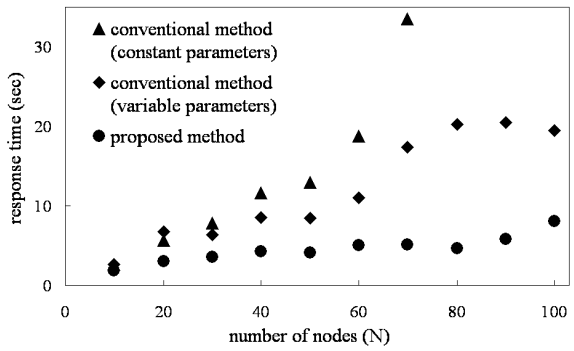

Figure 7: Response time according to change in number of nodes.

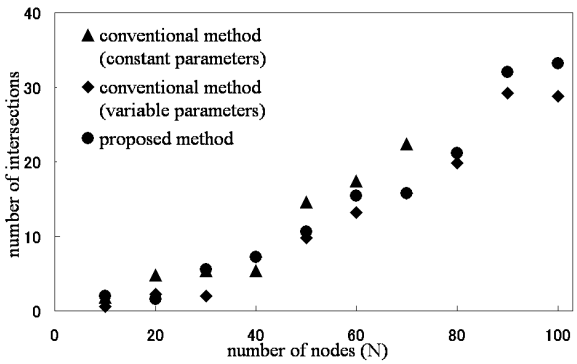

Figure 8: Number of intersections according to change in number of nodes.

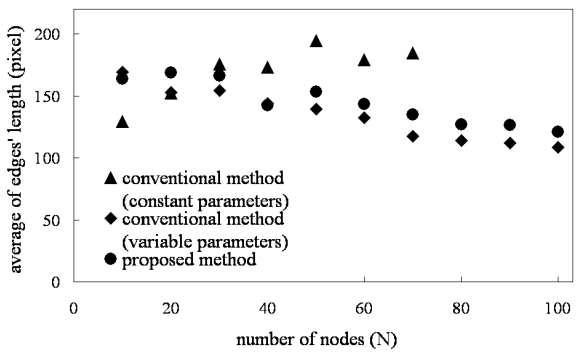

Figure 9: Average of length of edges according to change in number of nodes.

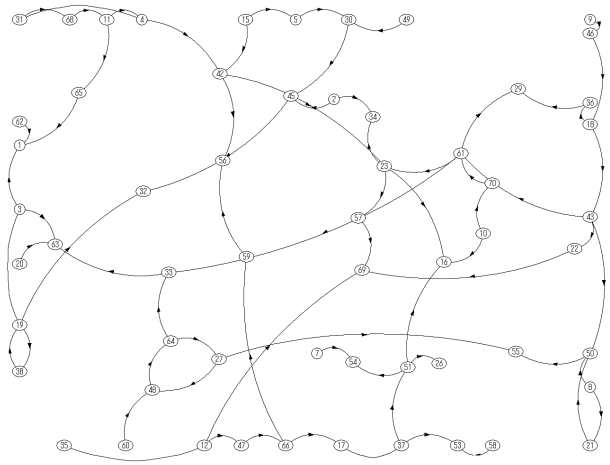

Figure 10: Visualization of the 70×70 matrix in the conventional method with constant parameters (30.3 seconds to converge).

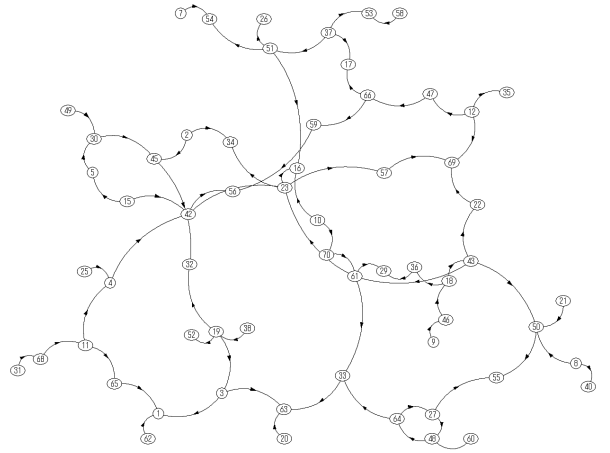

Figure 11: Visualization of the 70×70 matrix in the conventional method with variable parameters (10.7 seconds to converge).

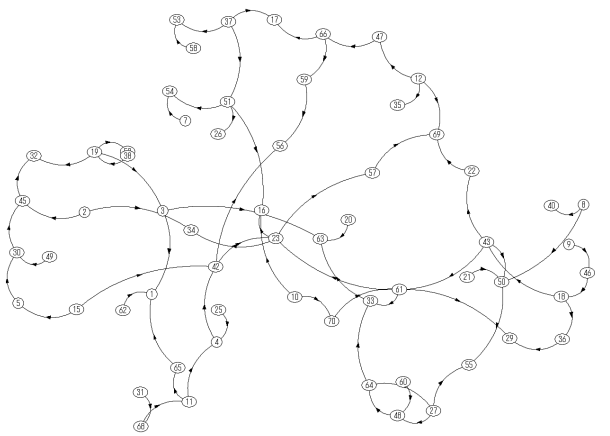

Figure 12: Visualization of the 70×70 matrix in the proposed method (3.31 seconds to converge).

The difference of response time has extended as number of nodes increases. In the conventional method with constant parameters, when number of nodes is over 80 it doesn't converge. Though the number of intersections is almost the same, this method's nodes come to gather in the edge of the display as the number of nodes increase. As an example, we apply the same initial graph to three methods. In addition, though there is hardly a difference between the graph of the proposed method and the conventional method with variable parameters, calculating time of the proposed method is about a half of the conventional method with variable parameters.

The number of intersections has increased as the number of nodes increases. Reasons with intersection are as follows.

- It is not possible to slip out local minimum solutions.
- In this model, some nodes have always the intersection. Figure 13 shows this case.

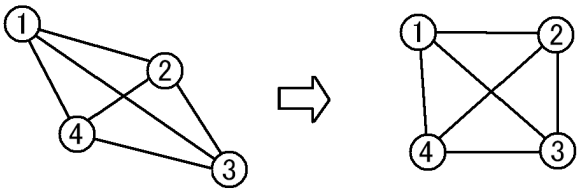

Figure 13: The model always having intersection.

5. Conclusion

In this paper, we proposed a new method whose calculation time is less than conventional method. The calculating time of the proposed method was about a half of the conventional method with variable parameters.

Future works are as follows:

- Correspondence to further large-scale data
- Improvement of algorithm
- Improvement of drawing about edges
- Making the method of evaluating the graph

Acknowledgments

This research is supported by the fund of Open Research Center Project from MEXT of Japanese Government (2002-2006).

References

[1] S. Morishita, et al., "Discovery science and data mining," *Kyouritushuppan, 2001(in Japanese).*

[2] Y. Hattori, et al., "Graph theory," *Shokoudou, 1984(in Japanese).*

[3] Michael. J.A. Berry, et al., "The data-mining technique," *Kaibundou, 1999(in Japanese).*