Advances and Applications in Discrete Mathematics

Published Online: April 2016
http://dx.doi.org/10.17654/DM017020201
Volume 17, Number 2, 2016, Pages 201-209 **ISSN: 0974-1658**

REINFORCING THE SPRING ALGORITHMS: THE TAGUCHI APPROACH

Ismo Lapinleimu and Erkki Mäkinen

UKK Institute
P.O. Box 30, FI-33501 Tampere
Finland
e-mail: ismo.lapinleimu@uta.fi

School of Information Sciences
University of Tampere
FI-33014, Finland
e-mail: erkki.makinen@uta.fi

Abstract

The spring algorithms are perhaps the most flexible method for drawing general graphs nicely. In principle, they are easy to implement and suitable for drawing different types of graphs. The price to be paid for their flexibility and wide suitability in various situations is the difficulty of choosing the parameters steering the drawing process. We present a method for making the choice of the parameters easier than the trail-and-error approach currently in use by applying the Taguchi method for designing of experiments.

Received: December 4, 2015; Revised: December 11, 2015; Accepted: December 19, 2015

2010 Mathematics Subject Classification: 68R10.

Keywords and phrases: algorithms, graph drawing, spring algorithms, designing of experiments.

Communicated by K. K. Azad

1. Introduction

Spring algorithms (also called force-directed algorithms) allow flexible drawing of various types of graphs. They are context-independent in the sense that their behaviour depends on the structure of the graph, but not on domain-specific information possibly available. Typically, a spring algorithm starts with initial positions of the nodes and allows a force to effect to the positions for some period of time, i.e., it repeatedly calculates the new positions under the assumption of spring forces, gravity, magnetic forces or other similar natural analogy. Kobourov [4] lists numerous variations of the spring algorithms. After over 30 years of active research, spring algorithms are still vigorously studied, see, e.g., [2, 3, 5].

Naturally, it is essential for the quality of the drawings which parameters (coefficients in the force calculations) are used when determining the positions. None of the existing spring algorithms have an exact method for determining the parameter values, but a trial-and-error approach is used when finding a proper combination of the parameters. This usually leads to a situation where the drawing is not as good as possible.

We consider graph drawing by using spring algorithms as a production line which produces layouts, and in order to solve the problem of choosing correct parameters in a given practical drawing task, we propose the use of the Taguchi method for designing of experiments.

The Taguchi method is a general quality control method which can be used both to optimize a new product or process, and to improve existing ones. For estimating the quality, it uses: (1) the quality loss function, and (2) signal to noise ratio. When drawing graphs, the quality means the fulfillment of so called aesthetics criteria [7].

The paper is organized as follows. In Section 2, we introduce the Taguchi method, and in Section 3, we recall some properties of typical spring algorithms, so that we can show in Section 4, how the Taguchi method can be applied when fixing the parameters. Finally, in Section 5, we draw conclusions.

2. The Taguchi Method

Taguchi assumed that each product passed to the user generates losses, and the size of a loss is inversely proportional to the quality of the product. In our case, the quality is measured by the aesthetics criteria of graph drawing, and losses are damages caused by the violations of the aesthetics criteria, such as misunderstandings or decreased transmission of information when watching the resulting layout. The form of the quality loss function depends on the target of the measure used.

There are three basic types of loss functions [8]:

1. The-nominal-the-best (*N* type).
2. The-smaller-the-better (*S* type).
3. The-larger-the-better (*L* type).

Several other types of loss functions are introduced in the literature, but the above three types are sufficient for the present paper.

A nominal size of an aesthetics criterion, and hence, an *N* type loss function, is required, e.g., when the distance of the vertices in a layout is considered. Although adjacent vertices should preferably lie near each others, they should not be too close, since that would decrease readability of the layout. *S* type loss function is needed with several aesthetics criteria, e.g., with the number of edge crossings or the variation of the edge lengths. An example of an *L* type loss function is the one corresponding to the aesthetics criterion of maximizing the minimum angle leaving a vertex (for other aesthetics criteria with maximizing target, see [7]).

The uncontrollable factors which cause the functional characteristics of a product to deviate from their target values are called *noise factors*. In a typical production system, noise is caused by various manufacturing-process imperfections, like variations in machine settings [8].

In our case, noise is caused by unfavourable initial positions of the vertices and the possible inappropriateness of the algorithm for the present drawing problem (i.e., the force in the algorithm does not match the present drawing problem in the optimal way).

Taguchi uses signal-noise (S/N) ratio as the quality characteristic of choice. For the different loss function types, there are different characteristics:

1. *N* type: $10\log_{10}\frac{\mu^2}{\sigma^2}$,

2. *S* type: $-10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n} y_i^2\right)$,

3. *L* type: $-10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{y_i^2}\right)$,

where μ and σ stand for average and standard deviation, respectively, *n* is the number of observations and y_i is the observed data.

In our case, the controllable factors are the parameters of the spring algorithm. Their effects to the quality, i.e., to the aesthetics criteria of the resulting layouts are statistically tested in order to find out as good combinations of their values as possible.

3. The Eades' Spring Algorithm

Although there are numerous recent versions of the spring algorithm [2-5], the Eades' original spring algorithm [1] is used here as an example, because of its simplicity. The algorithm treats a graph as a mechanical system in which nodes are replaced by steel rings and edges by springs connected to the rings. All the springs have the same natural length *k*, and each spring has a current length *d*. Given a pair of rings connected by a spring, if $k > d$, then the spring attracts the rings. If, on the other hand, $k < d$, then the spring repulses the rings. If $k = d$, then the rings are stable. The spring forces will attract or repulse the rings until the system reaches the minimum energy (the balanced state). The strength of the forces is determined by *k* and *d*. Eades calculates the forces using the functions:

$f_a = C1 \times \log(k/C2)$ and $f_r = C3/d^2$, where f_a is the attractive force, f_r is the repulsive force, and $C1$, $C2$ and $C3$ are the coefficients. Fourth coefficient in Eades' algorithm is the scaling factor $C4$ which makes the drawing to fit in the area available. However, as $C4$ does not affect the quality of the drawing, we actually have three controllable factors.

4. Fixing the Parameters of the Eades' Algorithm

We use the Eades' algorithm as a running example in the rest of the paper, and test the coefficients $C1$, $C2$ and $C3$. We start by choosing three levels for each of the factors. These are shown in Table 1. There is no exact method of choosing the levels, but the user is expected to know the application area so that she can reasonably perform the choice.

Table 1. The chosen levels of the factors $C1$, $C2$ and $C3$

Factor	Low	Medium	High
$C1$	–100	–50	–5
$C2$	5	40	75
$C3$	10	100	200

Instead of all possible combinations of the parameters, the Taguchi approach uses orthogonal matrices which give the combinations of parameters to be tested. The most suitable orthogonal matrix in our case is L_9 [6].

Table 2 shows the matrix L_9 with one column deleted (because we have only three factors to be tested).

Table 2. The orthogonal matrix L_9 [6] (with ordinals replaced with the names of the levels)

Test	$C1$	$C2$	$C3$
1	Low	Low	Low
2	Low	Medium	Medium
3	Low	High	High
4	Medium	Low	Medium
5	Medium	Medium	High
6	Medium	High	Low
7	High	Low	High
8	High	Medium	Low
9	High	High	Medium

As an example, the parameter combinations are now tested with two quality measures, the number of edge crossing and the variation of the edge lengths. Since both the measures have quality loss function of type *S*, we use the function $\eta = -10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n} y_i^2\right)$ for the S/N ratio.

The test runs are performed with the termination criterion until the values of the quality measures do not change or the result alternates between a few values. This follows the original presentation of Eades [1]; naturally, different parameters would have been chosen with some other termination criterion. The test set of graphs contains 10 graphs of moderate size (up to 10 vertices and 12 edges), i.e., graphs that typically are drawn by the spring algorithms.

First, the total averages of the quality measures in all test runs are calculated. The resulting averages are shown in Table 3.

Table 3. Averages of edge crossings and variations of edge lengths

Test	Edge crossings	Variations of edge lengths
1	3,4	21,84
2	4,0	6,3
3	6,4	37,63
4	9,11	152,97
5	5,56	19,22
6	109,67	3,96
7	8,11	1282,20
8	53,89	10,09
9	14,67	117,24

Second, the effects of the parameters are compared with an analysis of variance. Tables 4 and 5 show the results for the numbers of edge crossings and for the variations of edge lengths, respectively.

Table 4. Analysis of variance for the S/N ratios of edge crossings

	Averages at levels							
	Low	Medium	High	df	SS	MS	F	p
$C1$	–24,67	–29,63	–29,97	2	53	26	175	0,006
$C2$	–26,86	–25,85	–31,57	2	56	28	185	0,005
$C3$	–39,30	–23,71	–21,26	2	575	287	1904	0,001

Table 5. Analysis of variance for the S/N ratios of variations of edge lengths

	Averages at levels							
	Low	Medium	High	df	SS	MS	F	p
$C1$	–27,84	–29,25	–45,72	2	593	296	48	0.021
$C2$	–47,89	–22,33	–32,59	2	993	496	80	0.012
$C3$	–20,13	–37,84	–44,85	2	974	487	78	0.013

All the parameters in both cases are statistically significant $(\alpha = 0.05)$. Table 4 suggests the parameter values $C1 = 100$ (low), $C2 = 40$ (medium) and $C3 = 200$ (high), while Table 5 suggests the values $C1 = 100$ (low), $C2 = 40$ (medium) and $C3 = 10$ (low). The combinations (low, medium, high) from Table 4 and (low, medium, low) from Table 5 are not among the combinations of Table 2. The S/N ratio for the values suggested by Table 4 is 15,6 and the S/N ratio for the values suggested by Table 5 is 1,8, and the corresponding variation of edge lengths is 2,25. Hence, the combinations found with the help of Tables 4 and 5 are better than those in Table 2.

5. Concluding Remarks

We have proposed the use of Taguchi approach in fixing the parameters of spring algorithms. The proposed method can naturally be used with other spring algorithms that the Eades algorithm which was used here because of its simplicity. As can be expected, different quality measures prefer different parameters. The final parameters can be obtained as weighted average of the optimal parameters for individual measures.

We have demonstrated the use of the approach in sample case and performed the calculations by hand. Naturally, all these can be automated in single software package with a well-organized user interface.

Acknowledgement

The authors thank the anonymous referees for their valuable suggestions which let to the improvement of the manuscript.

References

[1] P. A. Eades, A heuristic for graph drawing, Congressus Numerantium 42 (1984), 149-160.

[2] J. Hua, M. L. Huang and Q. V. Nguyen, Drawing large weighted graphs using clustered force-directed algorithm, Proc. of the 18th International Conference on Information Visualisation (IV), 2014, pp. 13-17.

[3] M. Huang, W. Huang and C.-C. Lin, Evaluating force-directed algorithms with a new framework, Proc. of the 27th Annual ACM Symposium on Applied Computing, 2012, pp. 1030-1032.

[4] S. G. Kobourov, Spring embedders and force directed graph drawing algorithms, arXiv:1201.3011v1.

[5] C.-C. Lin and H.-C. Yen, A new force-directed graph drawing method based on edge-edge repulsion, J. Visual Lang. Comput. 23(1) (2012), 29-42.

[6] M. S. Phadke, Quality Engineering using Robust Design, Prentice-Hall, Englewood Cliffs, NY, 1989.

[7] H. C. Purchase, Metrics for graph drawing aesthetics, J. Visual Lang. Comput. 13(5) (2001), 501-516.

[8] G. Taguchi, E. A. Elsayed and T. Hsing, Quality Engineering in Production Systems, McGraw-Hill, New York, NY, 1989.