Simulated Annealing as a Pre-Processing Step for Force-Directed Graph Drawing

Farshad Ghassemi Toosi
Department of CSIS
University of Limerick
Limerick, Ireland
farshad.toosi@ul.ie

Nikola S. Nikolov
Department of CSIS
University of Limerick
Limerick, Ireland
nikola.nikolov@ul.ie

Malachy Eaton
Department of CSIS
University of Limerick
Limerick, Ireland
malachy.eaton@ul.ie

ABSTRACT

We report on our findings using Simulated Annealing (SA) as a preprocessing step for force-directed graph drawing. Our proposed SA algorithm finds a smart initial vertex placement (instead of a random initial vertex placement) in order to decrease the chance of having edge crossings (local minima) and also to decrease the number of required iterations from start placement to the final placement.

Keywords

Routing and layout; Simulated Annealing ; Fitness evaluation; Combinatorial optimization; Running time analysis

1. INTRODUCTION

Force-directed graph drawing is probably the most popular approach to finding aesthetically pleasing 2D layouts of general graphs [3, 6]. This is due to a number of factors which include the relative simplicity of implementation and the reasonable running time. Moreover, the final result typically emphasises the structure of the graph and exhibits a relatively low number of edge crossings, which is an indicator for the aesthetic qualities of the layout [8].

For reasons of efficiency, the initial layout required as an input to force-directed graph algorithms is typically a random one, i.e. a layout in which vertices are assigned random coordinates [1, 3–5, 10]. Nevertheless, as illustrated in Figs. 1(a) and 1(b), the choice of an initial layout can greatly influence the outcome of a force-directed graph drawing algorithm. The initial layout for Figure 1(a) is chosen by our algorithm and the initial layout in Figure 1(b) is chosen randomly. If a force-directed graph drawing is considered as a a solution to the problem of minimising the number of edge crossings in the final layout, then the choice of an initial layout can help a force-directed algorithm avoid local minima. Finding such smart initial layouts is a hard combinatorial optimization problem, thus a heuristic approach to it such as *genetic algorithms* or *simulated annealing* can potentially be very beneficial.

GECCO'16 Companion, July 20-24, 2016, Denver, CO, USA

DOI: http://dx.doi.org/10.1145/2908961.2931660

Simulated annealing is a heuristic technique that mimics the process undergone by misplaced atoms in metal when heated up and then slowly cooled down [12]. In previous work, simulated annealing has been used as a graph-drawing algorithm for finding straight-line layouts of general undirected graphs [2]. In our work we do not use simulated annealing for drawing a graph, we instead use it as a prepossessing step for well-known force-directed graph drawing algorithms.

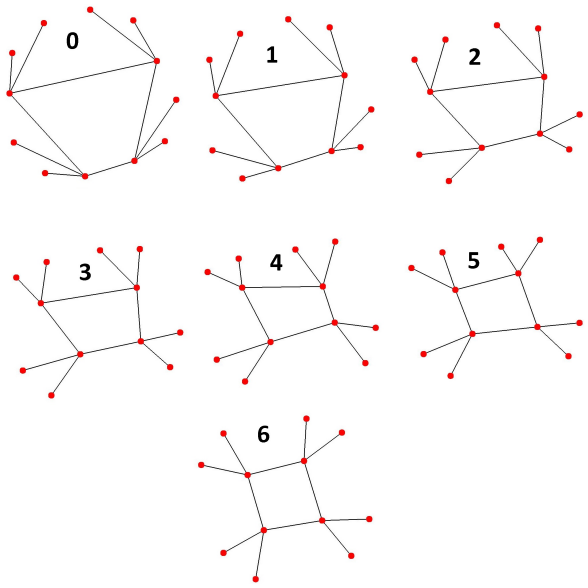
In this paper we report the utilisation of a simulated annealing algorithm for quickly choosing a good starting point for a classical force-directed algorithm. We report experimental results with our algorithm as a pre-processor to the Fruchterman and Reingold force-directed graph-drawing algorithm [4]. A similar idea is exploited in previous work, where a genetic algorithm is used instead of simulated annealing [9] .

2. SIMULATED ANNEALING ALGORITHM

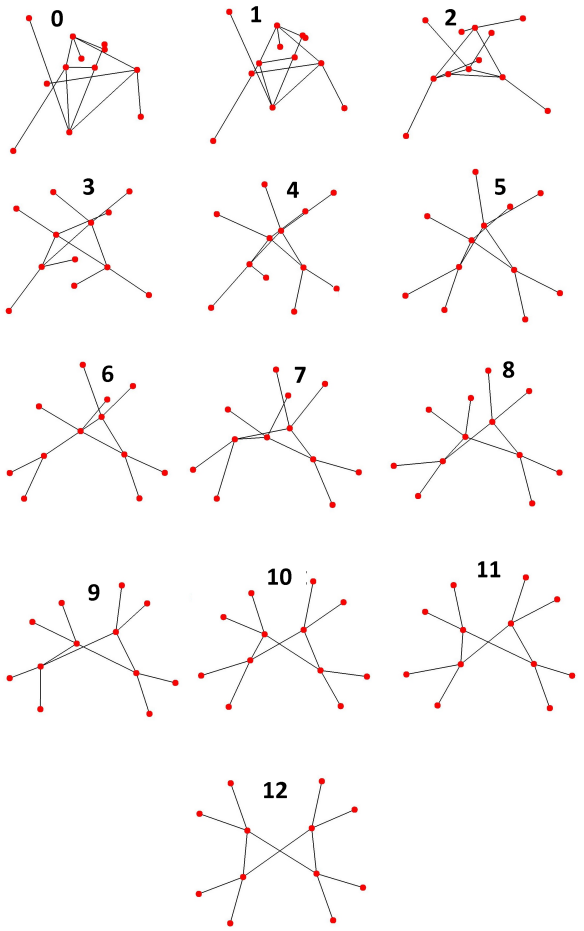
Simulated Annealing usually starts with a randomly generated candidate as a solution. The temperature of the system is initially high therefore the candidate is allowed to modify itself by a high rate (proportional to the temperature) with the hope of finding a better solution. As the temperature gets lower (system is getting cooler) the candidate modifies itself with lower rate until the system is, so called, frozen, and no more modification is allowed. The operation which does the modification on the candidate is called *Mutation*. In section 2.2 we introduce our SA algorithm by describing its fitness function and the required operators. A candidate (ordering) in our algorithm represents a sequence of vertex indices. We consider a discrete drawing space and represent a layout by a circle which can be easily converted to a linear environment for the sake of computational simplicity. The considered circle has a radius equal to 1 and its border is divided by $|N|$ (number of vertices in the graph). Two different orderings for the same graph with 9 vertices are shown in Figure 2. Figures 2(a) and 2(b) show how two different orderings **[5-3-2-1-4-7-8-9-6]**, generated by simulated annealing, and **[2-1-5-7-9-6-8-4-3]**, generated randomly, on a circle create two different initial vertex placements according to their orderings. Figure 2(c) represents the final placement (layout) by Fruchterman and Reingold algorithm [4].

2.1 Fitness

Our fitness function aims at minimizing the total length between certain pairs of vertices in the graph. Normally any

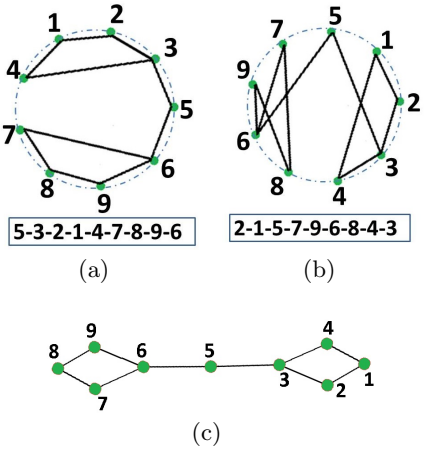


(a)



(b)

Figure 1: Two different initial layouts of the same graph leading to two different outcomes, using 6 and 12 iterations respectively of the Fruchterman-Reingold force-directed graph drawing algorithm.



(a) (b)

(c)

Figure 2: A graph with 9 vertices with two different orderings with their corresponding initial vertex placements (2(a) and 2(b)) respectively. The actual layout using the Fruchterman and Reingold algorithm is shown in Figure 2(c).

pair of nodes is either *adjacent* or *non-adjacent*. Finding the theoretical distance between *non-adjacent* pairs is an expensive process especially if the graph is large. With a less expensive process one can find only those *non-adjacent pairs* with length equal to 2. The idea of the employed function for the fitness function is similar to the one proposed by the Kamada and Kawai algorithm [5]; meaning that our proposed SA tries to minimize the length between adjacent pairs and those pairs in which their theoretical distance is equal to 2. Starting from the first vertex in the ordering, each vertex is given a polar coordinate starting from 0 to 2π. See Figure 3. The fitness of layout L of graph $G = (V, E)$ with a set of vertices $V = \{v_1, v_2, \ldots, v_n\}$ and a set of edges $E \subseteq V \times V$ is expressed in Equation (1) where $e.v_P$ and $t.v_P$ are the polar coordinate of the vertex v and $|e.v_P - e.u_P|$ is the distance between $e.v_P$ and $e.u_P$. T is the set of vertex pairs with theoretical distance equal to 2.

$$f(L) = \sum_{e \in E} |e.v_P - e.u_P| + \sum_{t \in T} |t.v_P - t.u_P| \qquad (1)$$

Using this fitness function, connected groups of vertices tend to be placed closer to each other therefore it makes a better starting point for the actual force-directed algorithm.

2.2 Mutation

Mutation is an operation which makes changes on candidate solutions. There are several different mutation operations [7]. Exchange Mutation (EM) [7] is the mutation operation that we have employed in this work. This operation selects two random individuals and swaps their position in the ordering, see Figure 4.

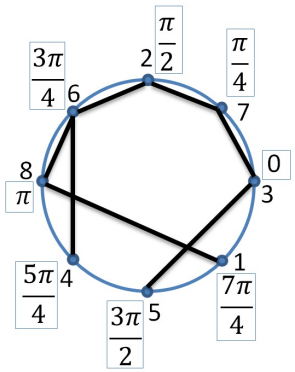
Figure 4 shows an example of mutation on only one pair of individuals. The number of pairs which are applied for mutation is proportional to the temperature of the system, meaning that, at the beginning of the SA process a larger number of pairs are applied for mutation. As the system gets cooler (temperature decreases), the number of pairs undergoing mutation gets smaller as well.

Table 1: Results of applying SA as a pre-processor for Fruchterman-Reingold.

Graph	n	m	SA-planar	R-planar	SA-cross	R-cross	SA-run (s)	R-run (s)
1	7	7	100	75	0	0.25	0.0022	0.002
2	39	45	35	17	0.86	1.11	0.3	0.2
3	51	63	90	80	0.31	0.72	0.57	0.46

Table 2: Results from experiments with 512 graphs from Rome repository.

No. L	Avg. *Total*	Avg. n	No. m	$SA > R$ *Cross*%	$R > SA$ *Cross*%	$SA > R$ *No.Graphs*	$R > SA$ *No.Graphs*	SA time(s)	R time(s)
L_1	512	9	11.55	11%	-3%	186	82	0.0061	0.0048
L_2	125	26.2	29.56	7%	-1%	39	7	0.09	0.074
L_3	213	26.62	36.86	19%	-0.6%	175	13	0.097	0.08



[3 - 7 – 2 – 6 – 8 – 4 – 5 – 1]

Figure 3: One example of an ordering on a circle with their poplar coordinates, e.g. the distance between the nodes 3 and 5 is the same as the distance between the nodes 2 and 8 and it is equal to 1.41**.**

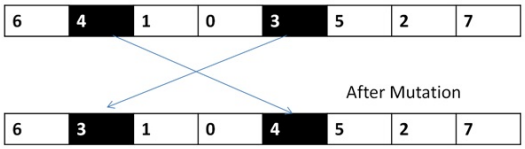


Figure 4: An example of Exchange Mutation on an ordering.

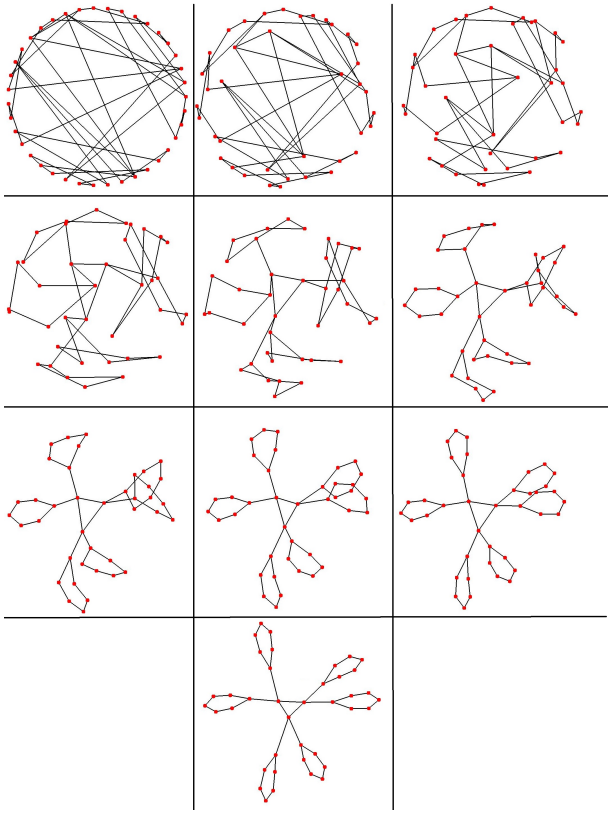
3. EXPERIMENTAL RESULTS

In order to evaluate our system we ran the force-directed algorithm of Fruchterman and Reingold [4] both with a randomly generated initial layout and with our SA algorithm for the same graphs and compared the number of edge crossings in the final force-directed layouts. The best individual found by our algorithm is the initial vertex placement for start point of the actual force-directed algorithm. We compared the force-directed algorithm using two different initial vertex placements one randomly generated and another one SA created. We measured the running time to create the SA initial vertex placement, the number of edge crossings are given in Table 1.

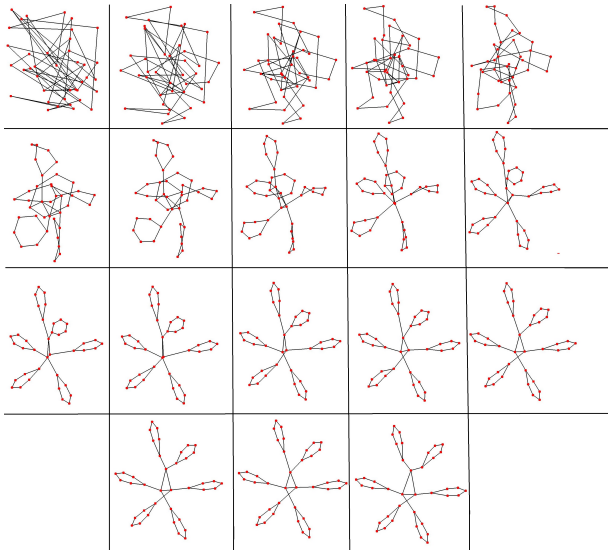
We have also run Fruchterman and Reingold on a few bigger sets of graphs with both randomly generated initial layout and also with our SA algorithm. The first dataset L_1 we used in our experimental study consists of 512 planar graphs from the Rome Graphs dataset [1] with vertex count 9 the results are listed in Table 2. The second dataset L_2 is comprised 125 randomly generated *cactus* graphs. The last dataset L_3 is a list of 213 star-shaped graphs which have been generated randomly based on the number of branches (see Table 2).

The ordering in Figure 2(a) creates more clear initial vertex placement for the actual drawing algorithm than the ordering in Figure 2(b). We have found that a more clear initial vertex placement (like the one in Figure 1(a)) not only causes a lower number of edge crossings (see Figure 1) but it also reduces the number of iterations in the actual algorithm; Figure 5 shows two drawings for the same graph with two different initial vertex placements; the drawing in Figure 5(a) gets to the final layout using less number of iterations than the drawing in Figure 5(b). This can be also realized in Figure 1 which Figure 1(a) needs less movement to get into the optimal layout compared with the one in Figure 1(b). Figure 5 shows a force-directed algorithm with two different start points (one starts with random initial vertex placement Figure 5(b) and another one starts with SA initial vertex placement Figure 5(a)).

[1] http://www.graphdrawing.org/data/



(a)



(b)

Figure 5: Force-Directed algorithm with two different initial vertex placement; Random initial vertex placement 5(b) and SA initial vertex placement 5(a).

4. CONCLUSIONS

In this work we propose to us simulated annealing as a preprocessing stage for force-directed graph drawing algorithms. Our proposed algorithm finds a graph layout that can be subsequently used as a smart initial layout by any force-directed algorithm. Our experimental study with the Fruchterman-Reingold on different graphs gives evidence that our algorithm improves the chances of finding a planar layouts of planar graphs and leads to a decreased number of edge crossings in the layouts of non-planar graph. This idea potentially can be used for node-placement process for the class of multilevel force-directed algorithms [11].

5. REFERENCES

[1] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[2] R. Davidson and D. Harel. Drawing graphs nicely using simulated annealing. *ACM Trans. Graph.*, 15(4):301–331, October 1996.

[3] P. Eades. On the future of graph drawing: Invited talk at the 18th International Symposium on Graph Drawing. http://graphdrawing.org/gd2010/invited.html, Accessed: 2015-06-08.

[4] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, 1991.

[5] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Inf. Process. Lett.*, 31(1):7–15, April 1989.

[6] S. G. Kobourov. Force-directed drawing algorithms. In Roberto Tamassia, editor, *Handbook of Graph Drawing and Visualization*, volume 81 of *Discrete Mathematics and Its Applications*, chapter 12, pages 383–408. Chapman and Hall/CRC, 2013.

[7] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artif. Intell. Rev.*, 13(2):129–170, April 1999.

[8] H. Purchase. Which aesthetic has the greatest effect on human understanding? In Giuseppe DiBattista, editor, *Graph Drawing*, volume 1353 of *Lecture Notes in Computer Science*, pages 248–261. Springer Berlin Heidelberg, 1997.

[9] F. Ghassemi Toosi, N. S. Nikolov, and M. Eaton. Evolving smart initial layouts for force-directed graph drawing. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, GECCO Companion '15, pages 1397–1398, New York, NY, USA, 2015. ACM.

[10] W. T. Tutte. How to draw a graph. *Proc. London Math. Society*, 13(52):743–768, 1963.

[11] C. Walshaw. A multilevel algorithm for force-directed graph drawing. In Joe Marks, editor, *Graph Drawing*, volume 1984 of *Lecture Notes in Computer Science*, pages 171–182. Springer Berlin Heidelberg, 2001.

[12] E. Weisstein. Simulated annealing. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/SimulatedAnnealing.html, Accessed: 2016-03-21.