Representing Space: A Hybrid Genetic Algorithm for Aesthetic Graph Layout

M.H.W. Hobbs and P.J. Rodgers
University of Kent at Canterbury, UK.
email: M.H.W.Hobbs@ukc.ac.uk, P.J.Rodgers@ukc.ac.uk

Abstract

This paper describes a hybrid Genetic Algorithm (GA) that is used to improve the layout of a graph according to a number of aesthetic criteria. The GA incorporates spatial and topological information by operating directly with a graph based representation. Initial results show this to be a promising technique for positioning graph nodes on a surface and may form the basis of a more general approach for problems involving multi-criteria spatial optimisation.

1. Introduction and Background

Many spatial problems have a common need to locate interacting objects on a surface. Applications as diverse as geographical data analysis and molecular modelling deal with location as a special feature within a vector of attributes that describe a problem. Generic GAs have been used in applications such as electronic engineering design problems [6]; molecular conformational analysis [13,14,]; network and graph optimisation [15,17]. These systems deal with a variety of graph based representations but typically use a one dimensional encoding which means that either spatial information is lost, or extra, problem specific constraints have to be introduced.

Hybrid GAs which model the specific spatial needs of a problem have been developed [4,10], and other evolutionary techniques, using richer representation structures such as Genetic Programming [9] have been applied to spatial problems. This paper proposes that spatial problems need a specifically spatial representation which, combined with the more generic aspects of GAs can be used as a general solution technique.

The problem considered here is to locate the nodes of an arbitrarily connected graph so that it conforms to aesthetically pleasing principles of layout [1]. It is inherently difficult to optimise even one criteria such as minimising edge crossings [18]. Current research has focused on force models [7] and simulated annealing [5] but these are more likely to be trapped by local optima than a GA. The GA also has considerable potential to tackle large problems as the time complexity of the optimisation technique is not dependent on the size of the graph. Additionally, it is possible to reduce the time complexity for calculating the fitness function by using approximate methods and caching partial results.

The GA represents the spatial nature of the problem by using a graph based representation. The crossover operator is performed directly on the graph by splitting parent graphs and joining the resultant subgraphs into a child graph. This has the advantage of maintaining the geometric and graph theoretic association of the nodes negating the need for any separate constraint satisfaction.

The crossover operator also attempts to retain geometrically close subgraphs from each parent in an attempt to carry better subgraphs into the next generation. This contrasts with other GA graph drawing work that takes nodes (or node neighbourhoods) randomly from the two parent graphs [11] or which translates the graphs to a one dimensional representation and uses constraints on traditional crossover [12].

Section two of this paper briefly describes the main features of the genetic algorithm; section three presents some initial results and section four concludes by summarising our findings and discussing the current and future work for this project.

2. The Graphical Genetic Algorithm

The initial population is created by a randomly produced geometric node layout with the number of nodes and edges controlled by the user. Two graphs are selected for reproduction by choosing the fittest out of a four randomly selected from the population. This is a simple tournament selection mechanism with a low selection pressure.

2.1. Fitness Function

We use a multiobjective fitness function based on five well known measurable aesthetic criteria for graphs:

1. Total edge length. This is simply the sum of the lengths of each edge.
2. Graph area. This is a calculation of the area of the bounding rectangle of the nodes in the graph.
3. Node overlap. A weighted penalty is given to nodes that are too close or overlap each other.
4. Angular resolution. This is a calculation of the angle between edges connected to the same node and penalises connections that form acute angles.
5. Edge crossing. Applies a penalty function based on the number of edge crossings in the graph.

While these criteria are useful measures of aesthetic properties of graphs, this is not an exhaustive list and there are other measures that can be used [3,16].

The calculation of the fitness function follows the sum of weighted global ratios approach [1]. For each criteria i = 1..5, given above, the fitness ratio r_i is calculated from the criteria value, f_i, as follows:

$$r_i = 1 + (f_i - \text{Max Value}_i)/(\text{Min Value}_i - \text{Max Value}_i)$$

Where Max Value_i is the highest f_i found for that criteria in the operation of the GA, and Min Value_i is the lowest f_i found. This formula converts the criteria from an absolute range to a ratio that ranks it against the best found so far by the algorithm. This means that for any given generation we can tell which are the fitter individuals but that comparison across generations is dependent on the overall progress made in each of the fitness criteria. The fitness value, f, for a graph is then the weighted sum of these values:

$$f = w_1r_1 + w_2r_2 + w_3r_3 + w_4r_4 + w_5r_5$$

This fitness calculation is particularly suitable where there are a large number of criteria that need to be combined and the complex interrelations between them are not easily determined [8]. Weights are determined experimentally on a test graph to achieve the desired results and as they apply to ratios rather than absolute values these settings are suitable for graphs of different sizes and topologies.

2.2. Crossover and Mutation

We use a geometrically based gradient crossover function. This adds nodes to the child graph by taking nodes along a random gradient, one parent graph ascending the gradient and one descending the gradient. A node is taken from each parent graph in turn, unless the corresponding node is already in the child graph, in which case the next unused node is taken. This splits each parent graph into two subgraphs with equal number of nodes preserving spatial relationships and the main features of the topology.

Figure 1. shows two parent graphs, where the highlighted nodes form the child graph. Figure 2. shows the resultant child. The nodes are sorted on the indicated gradient (shown as a directed line next to each graph), with nodes from the first graph taken from top left to bottom right and nodes from

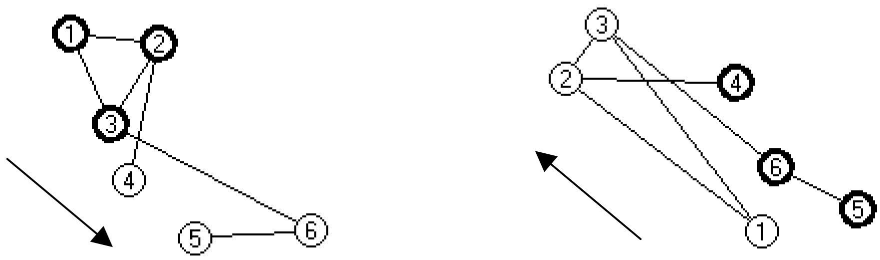

Figure 1. The two parent graphs undergoing crossover

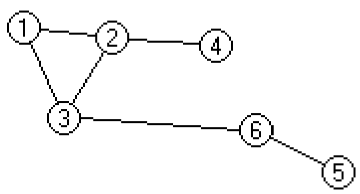

Figure 2. The Child graph

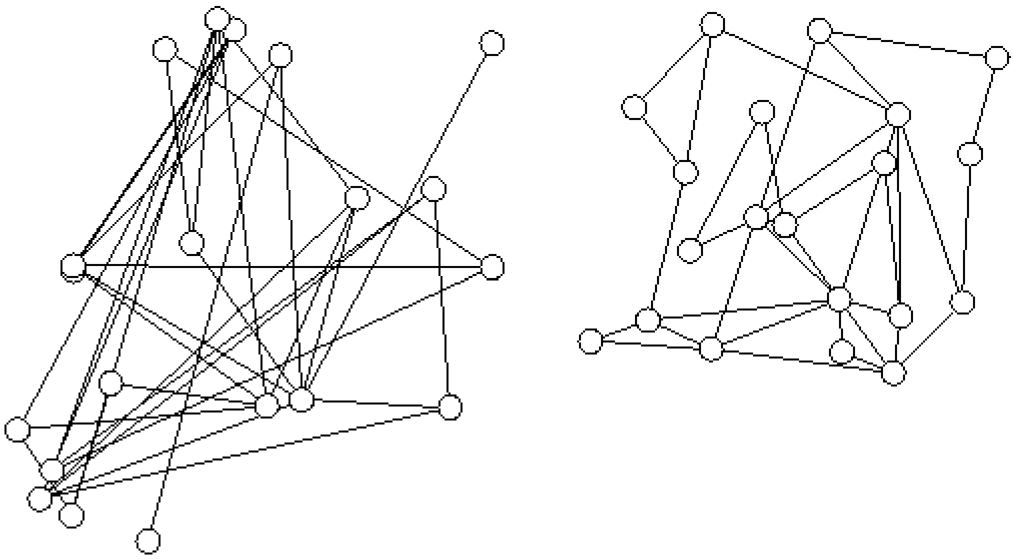

Figure 3. An initial graph and the aesthetically optimised result

the second graph taken from bottom right to top left.

We mutate a graph by picking nodes from the graph randomly, and moving them a random distance along the X and Y axis from their current location. The likelihood of a mutation and the size of the movement are controlled by the user.

3. Results

Figure 3. shows a randomly generated graph with 159 edge crossings and a graph from the final population which has four edge crossings produced by running the GA with a population of 20 for 1000 generations. Typically, most of the improvements were seen by around 250 generations with small improvements due to minor changes caused by the mutation operator.

A number of small test graphs were generated which could be optimised by hand. The results from the GA showed that in 3 out of 10 cases the optimum number was found and on average the GA produced graphs with just over 2 extra edge crossings.

4. Conclusions and Further Work

We have introduced a new geometrically based crossover technique for graph drawing GAs. This has been used within a hybrid GA that implicitly processes spatial information via the main operators of crossover and mutation. This has demonstrated that it is possible to hybridise the GA to implicitly process spatial information via the main operators of crossover and mutation without resort to special optimisation or constraint enforcement techniques. Much work remains even in this limited application area but this is a start towards a more general goal of providing a generic framework for a wide range of spatial problems.

There are two types of application that are appropriate to graph based GAs. The first are applications that use graphs where the graph theoretic structure is maintained throughout the GA process, as with graph drawing described here. An interesting area of this type is integrated circuit design, where the connections between circuits are constant, and the aim is to minimise the physical layout of circuits to improve the performance of the final chip. The second type of application is where the graph theoretic structure of the graph may alter during the process of the GA. One example is network routing, where in order to improve communication between pairs of nodes short paths are sought through a graph representing a communication network. For this second type, new graph crossover strategies may be required in order to produce graphs that are different, but still appropriate to the application area.

There are a number of areas that require work in the specific domain of graph drawing. The selection of aesthetic criteria needs to incorporate ideas of symmetry and even spacing between nodes before the results can compete with traditional methods for small graphs. However, even now the GA provides an excellent pre-processor to these methods where larger graph sizes make an exhaustive, deterministic approach slow or prone to trapping in local optima. The GA is adaptable because of the general way in which the fitness function is calculated and scalable due to the inherent efficiencies of the technique.

The overall speed of the fitness calculation can be improved by combining the calculation of the various criteria into one pass across the graph. Additionally, results for particular sub-graphs can be cached and used where these patterns re-appear. To speed up the GA, a less general layout technique could be employed, using a grid where nodes are restricted to appearing on certain regularly spaced points. The resolution of this grid will be sufficient so that location of a node on either of two adjacent points to will have a minimal effect on any of the fitness criteria. It would also allow the automatic enforcement of node overlap and minimum inter-node distance.

Before any conclusions can be drawn about the wider context of general spatial problem solving this technique will be tested on a number of different applications. However, it can be expected that a system that can dispense with ad-hoc optimisations and constraints will be easier to generalise and will be more sympathetic to the underlying processes of the GA technique.

References

1. Di Battista, P. Eades, R. Tamassa and I. Tollis, Algorithms for drawing graphs: an annotated bibliography. *Computing Geom. Theory Appl*. 4: pp 235-282. 1994.
2. P.J. Bentley and J.P. Wakefield. Finding Acceptable Solutions in the Pareto-Optimal Range using Multiobjective Genetic Algorithms. *WSC2 Second Online World Conference on Soft Computing*. June 1997.
3. F.J. Brandenburg, M. Himslot and C. Rohrer. An Experimental Comparison of Force-Directed and Randomised Graph Drawing Algorithms. *Symposium on Graph Drawing, GD'95. LNCS 1027.* Springer-Verlag. 1995.
4. R. Cooley, M. Hobbs, D. Pack. Genetic Algorithms and the Analysis of Spatially Referenced Data. *Applied Artificial Intelligence* 2:11 pp. 151-170. 1997.
5. R. Davidson and D. Harel, Drawing Graphs Nicely Using Simulated Annealing. *ACM Transactions on Graphics 15(4):301-331*. October 1996.
6. L. Davis, D. Orvosh, A. Cox and Y. Qiu, A Genetic Algorithm for Survivable Network Design. In *Proceedings of the fifth International Conference on Genetic Algorithms*, ed. S. Forrest, 408-415. Morgan Kaufmann. 1993.
7. P. Eades. A Heuristic for Graph Drawing. *Congressus Numerantium, 22*. pp. 149-160. 1984.
8. C. Fonseca, and P. Flemming. An Overview of Evoutionary Algorithms in Multiobjective Optimization, *Evolutionary Computation*, vol. 3 pp. 1-16. 1995
9. A. Ghozeil and David B. Fogel. Discovering Patterns in Spatial Data using Evolutionary Programming Genetic Programming 1996: Proceedings of the First Annual Conference, pp. 521-527, MIT Press,28-31 July 1996.
10. M. Hobbs. Spatial Clustering with a Genetic Algorithm. In D. *Parker, editor, Innovations in GIS 3*, pages 85-94 Taylor & Francis. 1996
11. C. Kosak, J. Marks, S. Shieber. Automating the Layout of Network Diagrams with Specified Visual Organization. IEEE Transactions on Systems, Man and Cybernetics. vol 24, no. 3 March 1994. pp. 440-454.
12. S.R. Mangano. Algorithms for Directed Graphs: A unique approach using genetic algorithms. Dr. Dobb's Journal. April 1994. pp. 92-97.
13. C.M. Oshiro, I.D. Kuntz and J. Scott Dixon. Flexible ligand docking using a genetic algorithm. *Journal of Computer-Aided Molecular Design*, 9:113-130. 1993.
14. A.L. Patton, W.F. Punch and E.D. Goodman. A standard GA Approach to Native Protein Conformation Prediction. *Proceedings of the sixth international conference on Genetic Algorithms*, 574-581. 1995.
15. P. Piggot and F. Suaraweera Encoding Graphs for Genetic Algorithms: An Investigation using the Minimum Spanning Tree Problem. In *Progress in Evolutionary Computation VIII*; 305-313 Springer. 1994.
16. H.C. Purchase, RF. Cohen and M. James. Validating Graph Drawing Aesthetics. *Symposium on Graph Drawing, GD'95. LNCS 1027.* Springer-Verlag. 1995.
17. F. Schweitzer, W. Ebeling, H. Rose and O. Weiss. Network Optimization Using Evolutionary Strategies. In *Parallel Problem Solving from Nature - PPSN IV*; Eds. Voigt, H., Ebeling, W., Rechenberg, I., Schwefel, H.; 940-949 Springer. 1996.
18. F. Shahrokhi, L.A. Székely and I. Vtr'o. Crossing Numbers of Graphs, Lower Bound Techniques and Algorithms. *International Workshop on Graph Drawing, GD'94. LNCS 894.* Springer-Verlag. 1995.