Applying Topology-Shape-Metric and FUZZY Genetic Algorithm for Automatic Planar Hierarchical and Orthogonal Graphs

Nahla F. Omran  
Department of mathematics  
Faculty of science, South Valley University  
Qena, Egypt

Sara F. Abd-el ghany  
Qena, Egypt

Abstract—The graphs appear in many applications such as computer networks, data networks, and PERT networks, when the network includes a small number of devices, it can be drawn easily by hand, as the number of devices increases, drawing becomes a very difficult task. For this problem we will develop a new method for automatic graph drawing based on two steps, the first is applying the topology –shape –metric that is approaching to orthogonal drawings for the grid and the second step is applying the fuzzy genetic algorithm that is directed, in the topology –shape –metric the final drawing is achieved through three sequential steps: planarization, orthogonalization, and compaction. Each of these steps is responsible for the quality of the final drawing. Then the genetic algorithm applied at the planarization step of the topology-shape-metric to find the geometric position of each vertex to minimize bending in the graph. The developed technique generates a greater number of planar embedding by varying the order of edges’ insertion. This is achieved clearly in the Results given in the paper.

Keywords—graph drawing; hierarchical graphs; topology-shape-metric; fuzzy genetic algorithms

I. INTRODUCTION

The expanding use of computers into business, science and the home, making scientists tend to draw diagrams to understand computer software. Graph drawing is a visualization of objects and relations between those objects. The effectiveness of the visualization of a graph is dependent on how efficiently the associated diagram conveys information to the users. The Specific requirements in this application are: Initially we will apply the topology-shape-metric that is divided into three main steps. The first step is the planarization step in this step reduces the number of edge crossings as much as possible.

The second step is the Orthogonalization: The goal of this step is to minimize the number of bends without changing the topology. The third step is the compaction in this step the goal is to minimize the drawing area. The problem of drawing graph was first studied in 1983 by M.R Garey, and D.S. Johnson [1] studied the problem of minimizing the number of edge-crossing. Then, in 1994, Di Battista [2] presented study to produce esthetically pleasing drawings of graphs based on main-cost-flow for both vertical and horizontal edge groups. Later, in 1999 Klau, Petra Mutzel [3] presented an approach based on a branch – and – cut algorithm which computes optimally labeled orthogonal drawings for compaction and labeling problem. In 2001, Maurizio Patrignani [4] presented study for the complexity of orthogonal compaction based on three problems consist of providing an orthogonal grid drawing, while minimizing the area, the total edge length, or the maximum edge length. In 2002 Markus Eiglsperger, Michael Kaufman [5] present a new compaction algorithm for orthogonal graph drawing with vertices of prescribed size.

Finally, applied the fuzzy genetic algorithm at the planarization step of the TSM. The use of fuzzy logic based techniques for either improving genetic algorithm behavior and modeling genetic algorithm components, the results obtained have been called a Fuzzy genetic algorithm. A FGA may be defined as an ordering sequence of instruction in which some of the instructions or algorithm components may be designed with fuzzy logic- based tools. In 1960[6], the Genetic Algorithms were first described by John Holland and further developed by Holland and his students and colleagues at the University of Michigan in the 1960 and 1970.In 1997, J.Branke, F.Bucher, H.Schmeck [7] use genetic algorithm for undirected graphs. In 1999, D.K.Pratihar, K.Deb, A.Ghosh [8] uses a fuzzy logic and a fuzzy genetic algorithm for the problems with mobile robots. In 2001, I.G. Damousis, K.J. Satsio’s, D.P. Labrdis, P.S. Dokopoulos [9], combined fuzzy logic and genetic algorithm techniques—application to an electromagnetic field problem. In 2006, P.Kuntz, B.Pinaud, A.Ghosh [10] used a hybrid genetic algorithm to minimizing crossing in hierarchical graphs. In 2007, D.Vrajitoru [11] applied a hybrid genetic algorithm to solve graph drawing problems. In 2011, Bernadette M.M, Gustavo H.D, Frederico G. Guimar, Renato C. M, Petr Ya. E [12] using a fuzzy genetic algorithm for automatic orthogonal graph drawing. In this chapter we will use TSM to minimize the crossing in the graph and then apply the FGA in the second step of TSM to find the geometric position of each vertex to minimize the bends in the graph to produce a graph with good esthetic criteria.

II. THE TOPOLOGY – SHAPE – METRIC TECHNIQUE

The topology-shape-metric was widely discussed and improved in 1998 by I.G. Toll’s, G. Di Battista, P. Eades, and R. Tamassia [13]. When applying the topology-shape-metric, the final drawing is produced by applying three consecutive steps: planarization, orthogonalization, and compaction.
A. Planarization

this step determine the topology of the graph, which test if the graph is planar or no. therefore, the goal of this step is to minimize the number of edge crossing as much as possible because the number of crossing affects the understanding of the graph. In 1998, I.G. Toll's, G. Di Battista, P. Eades, R. Tamassia [13] present algorithms that are used to build planar graph.

We will apply the planarization step on the graph in Figure2, we will get different planar graph by varying edge insertion to minimize edge crossing.

![Fig. 1. Block diagram for the topology-shape-metric algorithm [14, 15]](image1)

**Algorithm (Planarize).**

Input: graph G;
Output: planarization G_ of G;

1) Compute a maximal planar subgraph S of the input graph G, and partition the edges into “planar” and “non-planar”, as follows:

   a) Start with subgraph G_ consisting only of the vertices of G, but no edges;

   b) For each edge e of G, if the graph obtained by adding e to G_ is planar, then add e to G_ and classify e as “planar”, else reject e and classify it as “non planar”.

2) Construct a planar embedding of the planar subgraph G_ and the dual graph of S.

3) Add to G_ the non planar edges, one at a time, each time minimizing the number of crossings. This is done as follows for a non planar edge (u, v):

   a) Find a shortest (least number of edges) path in the dual graph of the current embedding G_ from the faces incident to u to the faces incident to v;

   b) Add the nonplanar edge and update G_ as well as its dual graph.

![Fig. 2. Example of nonplanar hierarchical graph.](image2)

![Fig. 3. Nonplanar hierarchical graph with one bend.](image3)

![Fig. 4. Planar hierarchical graph with three bends.](image4)
The graph in Figure 4 is planar but contains three bends, to solve this problem we will apply fuzzy genetic algorithm.

B. Orthogonalization

This step is performed to reshape the drawing to cancel the bends from the graph and the edge become straight line. We will use the Tamassia algorithm [14] computes an orthogonal shape of a planar graph with respect to an input embedding with a minimal number of bends the result of the orthogonalization on the Figure 2 is shown in Figure 5.

Algorithm. ORTHOGONAL

Input. A biconnected graph G.

Output. An orthogonal drawing of G.

1) Compute a st-numbering of G
2) Produce a reduced graph G I and modify the st-numbering so that there are no gaps in the st sequence.
3) Run Form _pairs on the reduced graph G.
4) Place vertices v and v2 in the same row, if v2 does not belong to a pair in which it shares a row with another vertex. If v1 and/or v2 have degree less than 4, then the placement of v1 and v2 might require one or two rows.
5) REPEAT
   a) Consider the next vertex vi according to Unmarked. The st-numbering of G.
   b) If v has already been placed, then go to Step 6.
   c) If vertex vi is unassigned, then place v, i in a new row. Connect vi with each vertex vj (j < i) such that (vj, vi) is a directed edge of G. Add as many uncompleted edges as required, depending on vi's out degree.
   d) If vertex vi is assigned to a pair, then place vi together with the other vertex in the same pair following the placement rules described above for the specific type of pair.
6) UNTIL the only remaining vertex is vn.
7) Insert vn, in a new row. If vn, is of degree 4, then there is an incoming edge that enters vn, from the top and bends twice. This edge is chosen to be the one that connects to vn, _l.
8) Restore the degree 2 vertices of G that were absorbed in Step 2.
9) End.

C. Compaction

In this section we will minimize the area of the given orthogonal drawing. The result of this step is shown in Figure 6.

Algorithm. planar graph compaction.

Input. √n x. √n bitmap planar graph layout.

Output. √n x. √n bitmap planar graph layout compacted one point to the east.

1) identify all points on layout that may possible move to the east. Mark these points to be movable.
2) unmark movable points that can cause connectivity violation to be stationary.
3) repeat step 2 until no further points is unmarked.
4) compact movable points to one point to the east maintaining connectivity.

III. PROPOSED FUZZY GENETIC ALGORITHM

We will solve the problem of bending in the graphs by using the FGA to find the geometric position of each vertex. We will applying the FGA at the planarization step to obtain a lot of planar graphs then submit this to orthogonalization and compaction step .The aesthetic criteria takes into consideration:

1) The number of crossings FX in the graph.
2) The number of bends fB in the graph.
3) The total sum of the edges’ length fL.

By minimizing all of them we will obtain the optimal graph.

The fitness function φ(st,i) = α1FX + α2fB + α3fL where i ∈ {0, 1}

We develop the diagram of TSM by adding FGA [15].

Algorithm (TSM-Fuzzy-GA).

Input: graph G;

Output: an optimized planar drawing;

1) Generation of the initial population:
   N = number of individuals;
   a) Generate at random the ordering of edge’s insertion from G (represented by an integer permutation);
2) Fitness computation: 
i = 0;
While (i < N) do
   a) Submit solution $s_i$ to the planarization step to obtain
      a planar embedding ($\Gamma_i$) and the number of crossings $FX(s_i)$;
   b) Submit the planar embedding ($\Gamma_i$) to the
      orthogonalization step to obtain the orthogonal representation
      $H$, and the number of bends $f_B(s_i)$;
   c) Submit the orthogonal representation $H$ to the
      compaction step to obtain the final drawing and the total sum
      of the edges length $f_L(s_i)$;
   d) Calculate the value of the fuzzy membership's $\mu_{FX}$,
      $\mu_{FB}$, and $\mu_{FL}$;
   e) Calculate the fuzzy-max-min aggregation $\mu_D$;
   f) $i = i + 1$;
3) Record the best individual according to the fitness
   function;
4) Application of the genetic operators for generating the
   new population. Each crossover operator (PMX or OX) for
   producing each offspring is selected with equal probability
   (0.50). The mutation operator to be used (scramble, swap,
   insert, and invert) for each offspring is also selected at
   random with equal chance (0.25);
5) Application of generational survival selection;
6) Go to step 2 until the stop criterion is met;
The results of applying FGA on the Figure2.

First step: in Figure7 we replaced v2 by v3 to cancel the
bend between v2 and v7.

Second step: in Figure8 we replaced v8 by v9 to cancel
the bend between v5 and v8.

IV. A GENETIC OPERATOR

in this paper we attempt to solve the problem of bends in
the graph by using FGA, the operator used in genetic
algorithms to maintain genetic diversity, known
as mutation and to combine existing solutions into
others, crossover. The main difference between them is that
the mutation operators operate on one chromosome, that is,
they are unary, while the crossover operators are binary
operators. Genetic variation is a necessity for the process
evolution. Genetic operators used in genetic algorithms are
analogous to those in the natural world: survival of the fittest,
or selection, reproduction (crossover, also called
recombination), and mutation.

A. Selection

The Selection operator decides which of the individuals in
the population will go into the next generation. This is decided
by the fitness value of an individual as calculated by the Fuzzy
Fitness Function. At this point the assumption is that a fitness
value pertaining to each individual is available.

B. Crossover

Crossover is the most widely used recombination operator.
Uniform 1-point crossover has been used. In general, 1-point
 crossover selects a random cut point and combines the first
portion of one parent with the second portion of the other and
vice versa to produce two offspring. The individual here
consists of an array of cluster numbers. Hence, the main issue
in recombination is the renumbering of clusters in the
resulting offspring.
C. Mutation

Mutation is needed to counteract the loss of some potentially useful genetic material during selection and crossover. In an artificial chromosome, this is affected by an occasional random alteration of the value of a string position. In a binary implementation, a bit value is toggled. In an integer or floating point implementation, a value is changed within an allowed range. This definition cannot be directly applied to the present scenario. A mutation operator that works at the boundaries of clusters has been worked out. For this, the pair-wise fitness between two consecutive data points is found for the whole data set. This has been calculated using the Fuzzy Fitness Function.

1) In each, select randomly, points in the range \((1, mi)\) where \(mi\) is the number of clusters in the \(i\)th individual. The number of clusters to be selected is equal to \(n\) mutation as worked out in the previous paragraph. Mutation is applied on the left as well as on the right border of each selected cluster which amounts to \(2*n\) mutation borders.

2) for a border point

Two cut-off values (in %) are to be fixed, one below which the pair-wise fitness will be classified as insignificant (lower value \((lval)\)) and the other above which the pair-wise fitness will be significant (high value \((hval)\)). The actual values will be implementation dependent.

a) Mutation Rule 1: if it's pair-wise fitness with the data point in the neighboring cluster is greater than \(hval\) and its pair-wise fitness with the neighboring data point within the same cluster is less than \(lval\), and then reallocates the data item to the neighboring cluster.

b) Mutation Rule 2: if it's pair-wise fitness with the data point in the neighboring cluster is greater than \(hval\) and also its pair-wise fitness with the neighboring data point within the same cluster is greater than \(hval\), then the two clusters can be merged.

c) Mutation Rule 3: if it's pair-wise fitness with the data point in the neighboring cluster is less than \(lval\) and its pair-wise fitness with the neighboring data point within the same cluster is also less than \(lval\), then the border point can be made a single-point cluster to which zero fitness value is assigned.

d) Mutation Rule 4: if none of the above conditions apply, then the border is left undisturbed.

V. RESULTS

In this paper, we tested many values for the number of vertices \(v\) in the graph, to generate an optimal graph by following the procedure shown in the diagram [16]:

1) We generated graphs by varying the number of vertices \(V\), from 10 to 600 vertices.

2) For each graph in the test set, when we applying the TSM on the graph in Figure 2, at the planarization step the final graph contain three bends shown in Figure 5, but when we applying FGA at the planarization step on the Figure 2, the final graph contains one bend shown in Figure 7.

Table 1 shows the results obtained with the classical topology shape-metric approach.

Table 2 presents the results obtained by the fuzzy genetic algorithm.

<table>
<thead>
<tr>
<th>(F)</th>
<th>(f_1)</th>
<th>(f_6)</th>
<th>(f_9)</th>
<th>(v)</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>16</td>
<td>3</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>87</td>
<td>38</td>
<td>2</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>148</td>
<td>64</td>
<td>5</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>232</td>
<td>92</td>
<td>6</td>
<td>6</td>
<td>40</td>
</tr>
<tr>
<td>441</td>
<td>132</td>
<td>19</td>
<td>24</td>
<td>50</td>
</tr>
<tr>
<td>1398</td>
<td>328</td>
<td>49</td>
<td>119</td>
<td>100</td>
</tr>
<tr>
<td>2126</td>
<td>632</td>
<td>59</td>
<td>137</td>
<td>150</td>
</tr>
<tr>
<td>2757</td>
<td>781</td>
<td>75</td>
<td>194</td>
<td>180</td>
</tr>
<tr>
<td>4769</td>
<td>854</td>
<td>172</td>
<td>509</td>
<td>200</td>
</tr>
<tr>
<td>5969</td>
<td>1144</td>
<td>197</td>
<td>618</td>
<td>250</td>
</tr>
<tr>
<td>9055</td>
<td>2748</td>
<td>173</td>
<td>608</td>
<td>500</td>
</tr>
<tr>
<td>9500</td>
<td>2800</td>
<td>198</td>
<td>620</td>
<td>520</td>
</tr>
<tr>
<td>10000</td>
<td>3000</td>
<td>200</td>
<td>630</td>
<td>550</td>
</tr>
<tr>
<td>10100</td>
<td>3200</td>
<td>195</td>
<td>616</td>
<td>600</td>
</tr>
</tbody>
</table>
Fig. 11. Block diagram for the TSM-Fuzzy-GA algorithm

<table>
<thead>
<tr>
<th>V-N stats</th>
<th>f_I</th>
<th>f_2</th>
<th>f_3</th>
<th>μ_D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>14</td>
<td>3</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>1</td>
<td>0.675</td>
<td></td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>0.95</td>
<td>0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>29</td>
<td>1</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>1</td>
<td>1</td>
<td>0.714</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>0.52</td>
<td>0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>48</td>
<td>5</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>1</td>
<td>1</td>
<td>0.26</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>1.69</td>
<td>0.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>51</td>
<td>5</td>
<td>0.857</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>5</td>
<td>6</td>
<td>0.433</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>4.37</td>
<td>0.71</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>12</td>
<td>10</td>
<td>0.769</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>14</td>
<td>12</td>
<td>0.446</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>1.69</td>
<td>1.96</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>62</td>
<td>31</td>
<td>0.917</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>70</td>
<td>36</td>
<td>0.330</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>7.89</td>
<td>3.30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>85</td>
<td>36</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>89</td>
<td>38</td>
<td>0.237</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>5.56</td>
<td>3.08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>95</td>
<td>51</td>
<td>0.696</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>110</td>
<td>52</td>
<td>0.321</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>10.27</td>
<td>2.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>334</td>
<td>120</td>
<td>0.698</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>342</td>
<td>130</td>
<td>0.198</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>18.57</td>
<td>5.91</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>427</td>
<td>145</td>
<td>0.848</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>429</td>
<td>152</td>
<td>0.279</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>12.77</td>
<td>4.47</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>319</td>
<td>103</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>521</td>
<td>108</td>
<td>0.697</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>5.67</td>
<td>6.98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>310</td>
<td>95</td>
<td>1.100</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>303</td>
<td>87</td>
<td>0.699</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>1.90</td>
<td>5.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>367</td>
<td>163</td>
<td>0.900</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>332</td>
<td>145</td>
<td>0.770</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>20.23</td>
<td>7.10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>358</td>
<td>150</td>
<td>1.900</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>290</td>
<td>110</td>
<td>0.800</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>2.29</td>
<td>7.80</td>
<td>0.56</td>
<td></td>
</tr>
</tbody>
</table>
VI. CONCLUSIONS

In this paper, we present a new method for automatic orthogonal graph drawing by using fuzzy genetic algorithm at the planarization step of the topology-shape-metric to find the geometric position of each vertex to obtain optimal graphs without crossing and bends.

Fig. 12. \( V = 100 \): (a) drawing obtained with topology-shape-metric (b) drawing obtained with the fuzzy genetic algorithm

ACKNOWLEDGMENT

I cannot express my feelings to my family who supported me all over through my way and backed me up during my studies wishing me good luck throughout the preparation of this thesis.

REFERENCES