Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 9, No. 4, November 2018), Pages: 115-127
www.jacr.iausari.ac.ir

A New Heuristic Algorithm for Drawing Binary Trees within Arbitrary Polygons Based on Center of Gravity

Jafar Mohamadian, Hossein Nematzadeh✉
Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran
jafarmohammadian91@gmail.com; nematzadeh@iausari.ac.ir

Received: 2018/05/28; Accepted: 2018/08/28

Abstract

Graphs have enormous usage in software engineering, network and electrical engineering. In fact graphs drawing is a geometrically representation of information. Among graphs, trees are concentrated because of their ability in hierarchal extension as well as processing VLSI circuit. Many algorithms have been proposed for drawing binary trees within polygons. However these algorithms generate binary trees with edge intersections and bends. Likewise, they have limitations in drawing binary trees in any arbitrary polygon with uniform distribution. The proposed algorithm which is based on calculating center of gravity in polygons draws a binary tree inside any kinds of polygon. The results which have been implemented with Microsoft Visual Basic reveal that the proposed heuristic algorithm had no edge intersection and no bend. In addition, not only the symmetry of drawing has been improved but also the computational complexity becomes less.

Keywords: *Binary Tree, Polygon, Edge Intersection, Bend*

1. Introduction

Designing integrated circuit manually is a complicated task. Therefore, many algorithms have been proposed for automatically designing circuits [1]. The problem of drawing binary trees inside polygon can be described as setting the tree vertices inside polygon. Many algorithms have been proposed for drawing binary trees within polygons [6, 7,9,10, 14, and 15]. However these algorithms generate binary trees with edge intersections and bends. Likewise, they have limitations in drawing binary trees in all kinds of polygon with uniform distribution. The proposed algorithm draws a binary tree inside any kinds of polygon. The proposed algorithm divides the input polygon into two segments using center of gravity. Then, for each segment a sub-tree is drawn and this process repeats until the number of centers of gravity equals to the number of tree vertices. The objectives in drawing binary trees were having no edge intersection, no bend, uniform distribution and increasing the symmetry of the result.

Edge intersections in any graph increases the complexity and confusion [13]. Thus, in circuit design, there should be no intersections among edges of graph. The existence of bends in a graph increases the overall cost of the result circuit. Uniform distribution leads to have a graph with vertices of not very close to each other and this increases readability of the generated graph. Moreover, because of the existence of some limitations, sometimes the design should be done on a certain surface. The center of

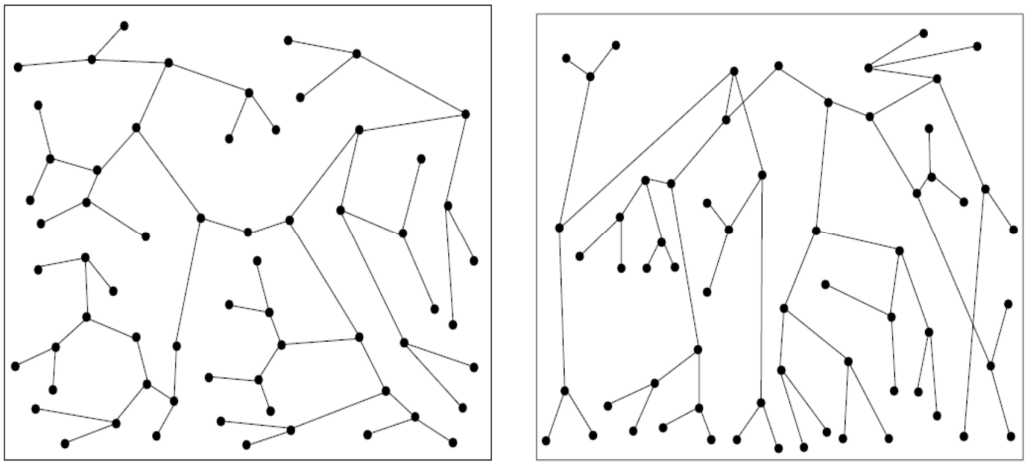
gravity method reduces the number of computations. In this paper in Section 2 the literature was clearly reviewed to highlight the relevant studies and their possible limitations. Section 3 described the proposed heuristic algorithm. Section 4 presented some examples of implementations and further illustrative evaluation of the proposed algorithm and its comparison with literature. Finally Section 5 provided the conclusion and possible future work.

2. Literature Review

In this section a critical review was done for identification and analysis of existing tree drawing algorithms.

2.1 Simulated Annealing Approach

Harel and Davidson (1996) proposed and algorithm using Simulated Annealing (SA) to draw a tree inside a rectangular polygon [8]. SA method is suitable for optimization problems in which the search space is both large and discrete. Basically, SA tries to optimize its objective function through comparison of a certain state with its neighbors. Therefore, the result may not be accurate and requires lots of time. In addition with SA based approach the distribution is not uniform and this could create edge intersections as shown in Figure 1a. Although, the hybrid skeleton and SA approach which was proposed by Bagheri and Razzazi (2004) has lower edge intersection and better distribution as shown in Figure 1b [1] but, the computational complexity is still high and the algorithm may not find the most accurate solution [2,3,5].



(a) ***(b)***

Figure 1. (a) Binary tree with SA and (b) binary tree with hybrid of SA and Skeleton

2.2 Orthogonal approach in Simple Rectangular Polygon

Bagheri and Razzazi (2010) proposed the orthogonal approach which is done by skeleton and runs on a rectangular polygon [4]. As shown in Figure 2 in this approach the centers of rectangles are connected to each other and this can cause bends which is a limitation itself. The skeleton procedure starts with one rectangle and moves on the skeleton until it reaches the half of the rectangle. Polygon division occurs. Edges are orthogonal in the rectangular drawing. In this way there is no possibility of drawing more than 4 degrees. Although this approach is free of edge intersection, but bends can

still be found. The order of implementation complexity is O (n^2 $m^{2.2}$). This method is only applicable on rectangular polygons.

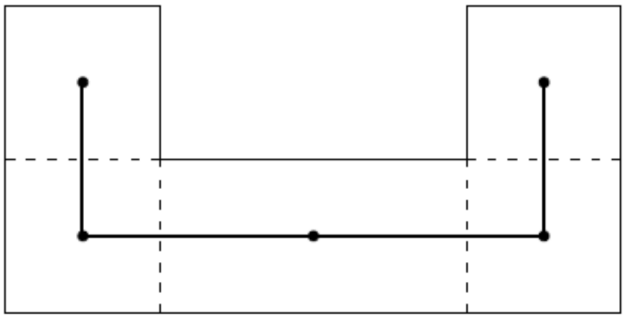
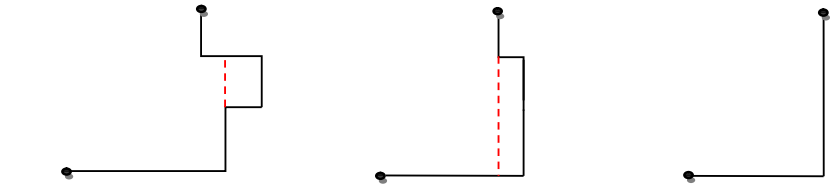


Figure2- Skeleton of orthogonal drawing

2.3 Orthogonal Approach in Rectangular Polygon with Hole

Jani (2011) proposed an algorithm using skeleton approach which is suitable with polygon with holes [16]. Although the proposed approach reduces the number of bends, there still exists bending as shown in Figure 3 in which five bends in Figure 3a reduced to one in Figure 3c. Garay and Johnson (1983) have already shown that decreasing edge intersection is NP-complete [11] Garg and Tamassia (1995) proved that decreasing bends is NP [12].



a) A sample with 5 bends *b) A sample with 5 bends* *c) A sample with 1 bend*

Figure 3. Decreasing number of bends

Previous works tried to optimize a certain behavior of drawing a binary tree. Likewise they tried to draw the binary tree on some certain kinds of polygon. The limitation of the background researches are as follow:

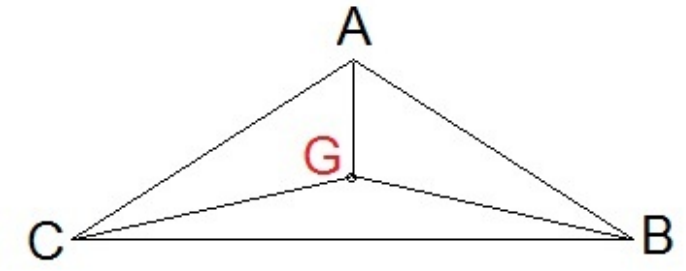
a) SA approach has edge intersections and bends. It is asymmetric and has high complexity. Its vertices distribution is not uniform.
b) Skeleton approach has bends and its complexity is high. The orthogonal approach that exploits skeleton is only suitable for rectangular polygons.

The proposed heuristic algorithm can draw binary trees on any kinds of polygon. In addition, it improves the following criteria considerably:

a) Number of bends: Decreasing number of bends to zero
b) Number of edge intersection: Decreasing number of edge intersection to zero
c) Uniform distribution: Tree vertices are uniformly distributed
d) Symmetry: Tree drawing is symmetric
e) Complexity: Improves complexity

3. Proposed Algorithm

Center of gravity in polygons is the balance point as was shown in Figure 4 with a triangular polygon. Center of gravity is a unique point in the polygon which doesn't change through rotation.



$$\overrightarrow{GA} + \overrightarrow{GB} + \overrightarrow{GC} = 0$$

Figure 4 –Center of gravity of a triangle

The proposed heuristic algorithm uses a simple approach to calculate the center of gravity inside polygon, and the initial polygon is divided into two segments. Segmentation process will continue and in each segment one sub-tree would be drawn. The root of the tree would be placed in the center of gravity of polygon and the offspring are then put in the center of gravity of the new sub-polygons. Segmentation process will continue until the number of centers of gravity equals to the number of nodes in the tree. The algorithm was presented as follow:

Input: number of vertices of the polygon (n) and the number of nodes in a binary tree (m)

Output: binary trees inside polygons

1. i:=1;
2. Calculate center of gravity of P_i and call it G_i.
3. Calculate the distance from the farthest vertex of polygon to G_i and call it A.
4. Extend G_iA in order to intersect with one edge of polygon and call it C.
5. Draw the line AC to create two polygons. (AC should pass G_i)
6. Call the new polygons P_{2i} and P_{2i+1} and their respected center of gravities G_{2i} and G_{2i+1}.
7. Draw G_iG_{2i} and G_iG_{2i+1} and then extend to intersect the polygons and create new ones.
8. i:=i+1;
9. If ($i \leq \frac{m-1}{2}$) then goto step 6.
10. Finally connect G_i to G_{2i} and G_{2i+1} to create the tree.

In the following the proposed algorithm was applied on a typical pentagon (a polygon with 5 sides) to consider the major steps of the algorithm.

3.1 Center of gravity calculation

Assuming the vertices of the polygon are (x_1, y_1), (x_2, y_2), …(x_n, y_n) the center of gravity G would be calculated from Equations 1 and 2:

$$G_x = \frac{X_1 + X_2 + \ldots + X_n}{n} \quad (1)$$

$$G_y = \frac{y_1 + y_2 + \ldots + y_n}{n} \quad (2)$$

Center of gravity G in a convex polygon is always located inside the polygon. Assuming a pentagon with the following coordinates: $\begin{bmatrix}3\\1\end{bmatrix}, \begin{bmatrix}7\\2\end{bmatrix}, \begin{bmatrix}9\\4\end{bmatrix}, \begin{bmatrix}8\\6\end{bmatrix}, \begin{bmatrix}5\\5\end{bmatrix}$. The center of gravity has been calculated with Microsoft VB using Equations 1 and 2 and was shown in Figure 5.

$$G_x = \frac{3+7+9+8+5}{5} = 6.4 \quad , \quad G_y = \frac{1+2+4+6+5}{5} = 3.6$$

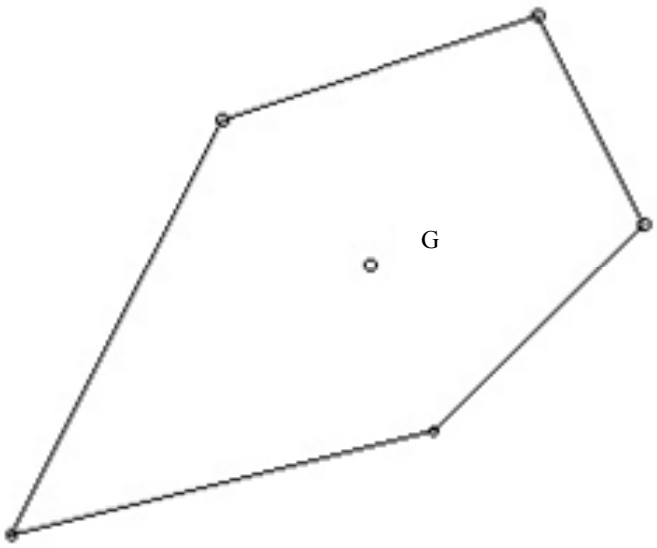


Figure 5 - locating center of gravity in a pentagon

3.2 Segmentation

The polygon should be divided into segments. Thus, the distance from the farthest vertex of polygon to G has to be calculated. Assuming the farthest vertex to G is A.

Then, G should be connected to A. Next GA is extended to intersect with one edge of polygon. The intersection point is called C. In this stage an edge is added to the polygon. The procedure is implemented by Microsoft VB and is shown in Figure 6.

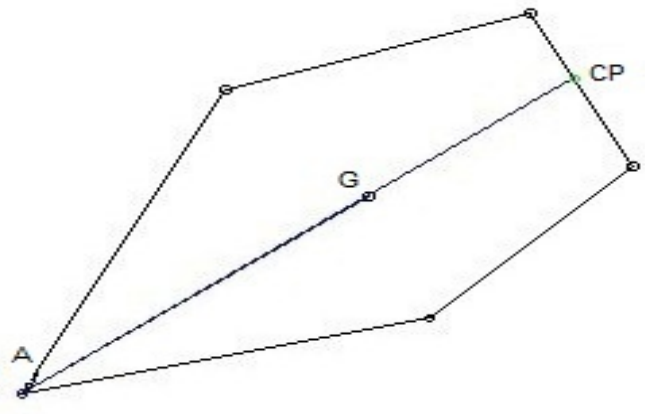


Figure 6 - Finding A and C in a pentagon

The segmentation continues and the respective centers of gravity should be calculated for new sub-polygons. Assuming G_1 and G_2 are new centers of gravity for new sub-polygons, GG_1 and GG_2 should be drawn and extended. This way the sub-polygons are iteratively segmentalized. This also guarantees to have no edge intersection. Furthermore, the computations are efficiently reduced. The root of the tree is G and the offspring are G1 and G2 as shown in Figure 7.

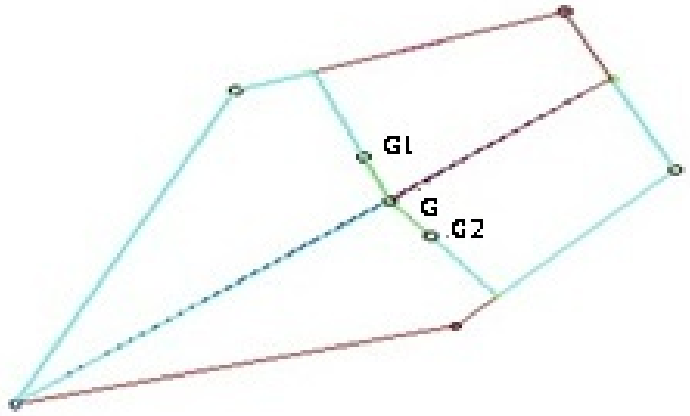


Figure 7- Calculation of G1 and G2 and continuing segmentalization

The process of segmentation and calculating the centers of gravity continue until the number of centers equals to the number of tree vertices. This way the number of segments inside polygon should equal to number of tree vertices. Figure 8 shows drawing a binary tree with 31 vertices inside a pentagon.

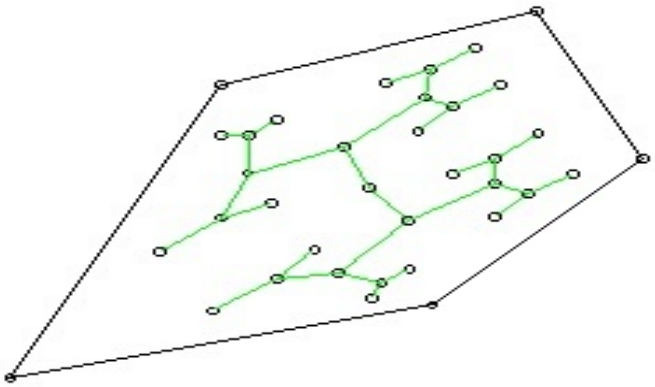


Figure 8 - Drawing a binary tree with 31 nodes inside a pentagon

3.3 Complexity

The complexity of calculating the center of gravity, calculating the intersection point, and segmentalizing the polygon is O (n) (only n edges are evaluated). The process iteratively repeats *m* times in which *m* is the number of nodes in a binary tree. Finally the complexity of the proposed algorithm is O (nm). This is much better in comparison with orthogonal approach with the complexity of O (n^2 $m^{2.2}$). Moreover, the proposed algorithm does not have any bends but in orthogonal approach bend has the complexity of O ($nm^{1.3}$) as shown in Table 1.

3.4 Symmetry Evaluation

In proposed algorithm for any node in left sub-tree there exists a node in right sub-tree. In fact these nodes are symmetric in relation with line GA in Figure 6. This symmetry is far better than existing algorithms [1, 4, 8, and 16]. This is based on this fact that G_i and $G_{i+2^{\lfloor \log_2^i \rfloor}-1}$ are symmetric. Assuming GA has the standard equation of $ax^2 + bx + c=0$, the symmetry of each point like B= (x_1,y_1) in relation with GA is B′ = (x_1-ak, y_1-2bk) in which *k* is calculated through Equation 3.

$$k = \frac{ax_1 + by_1 + c}{a^2 + b^2} \tag{3}$$

Assuming the square polygon in Figure 9 with four coordinates of $A\begin{bmatrix}2\\2\end{bmatrix}, B\begin{bmatrix}8\\2\end{bmatrix}, C\begin{bmatrix}8\\8\end{bmatrix}, D\begin{bmatrix}2\\8\end{bmatrix}$ with center of gravities with coordinates of $G_1\begin{bmatrix}5\\5\end{bmatrix}, G_2\begin{bmatrix}5.75\\4.25\end{bmatrix}, G_3\begin{bmatrix}4.25\\5.75\end{bmatrix}, G_4\begin{bmatrix}5.1875\\3.3125\end{bmatrix} G_5\begin{bmatrix}6.6875\\4.18125\end{bmatrix}, G_6\begin{bmatrix}4.18125\\6.6875\end{bmatrix}, G_7\begin{bmatrix}3.3125\\5.1875\end{bmatrix}$.

According to Figure 9 G_1A has the linear equation of x-y=0. The nodes of binary tree are symmetric in relation with y=x. Using Equation 3 the symmetry of G_4 is G_7 for instance in which the coordinates of x and y have been exchanged which can be clearly seen in Figure 9.

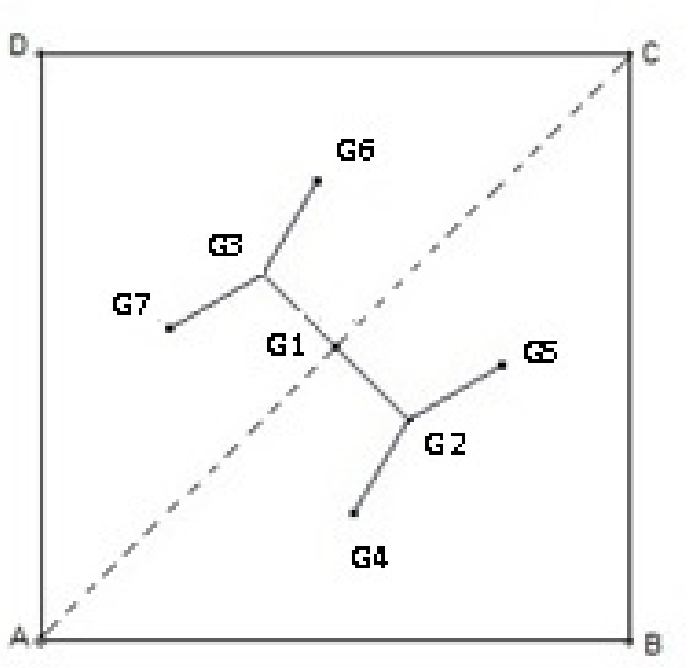


Figure 9 - Drawing a binary tree with 7 nodes inside a square polygon

3.5 Binary Nodes Distribution Evaluation

Figure 10 shows the optimum distribution of seven nodes inside a square polygon. Ideally, one node is assumed in the center of polygon. Then, the square polygon is divided into 6 rectangles. Therefore the coordinates of 7 nodes in Figure 10 will be $\begin{bmatrix}7\\6.5\end{bmatrix},\begin{bmatrix}5\\6.5\end{bmatrix},\begin{bmatrix}3\\6.5\end{bmatrix},\begin{bmatrix}5\\5\end{bmatrix},\begin{bmatrix}7\\3.5\end{bmatrix},\begin{bmatrix}5\\3.5\end{bmatrix},\begin{bmatrix}3\\3.5\end{bmatrix}$. With calculation of average points using Equations 1 and 2 and their variance using Equations 4 and 5 the dispersion of x and y coordinates would be calculated.

$$s^2{}_x = \frac{\sum_{i=1}^{n}(x_i - \overline{\overline{x}})^2}{n} \tag{4}$$

$$s^2{}_y = \frac{\sum_{i=1}^{n}(y_i - \overline{y})^2}{n} \tag{5}$$

Results between the ideal situation and the proposed algorithm are close. In one hand in ideal situation $\bar{x}$ and $\bar{y}$ equal to 5. In proposed algorithm $\bar{x}$ and $\bar{y}$ equal to 4.91 (based on Equation 1 and 2). Therefore the proposed algorithm has the mean accuracy of $\frac{4.91}{5}$ =0.98.

In other hand, in ideal situation S_x = 1.5 and S_y = 1.38. However for the proposed algorithm S_x = Sy =0.994. Standard deviation of *x* and *y* are $\frac{0.994}{1.5}$ = 0.67 and $\frac{0.994}{1.38}$ = 0.72 respectively. The pessimistic average of two mentioned characteristics are (mean and standard deviation) is $\frac{0.98+0.67}{2} = 0.82.5$. Thus it can be concluded that the proposed algorithm is 0.85 of ideal situation.

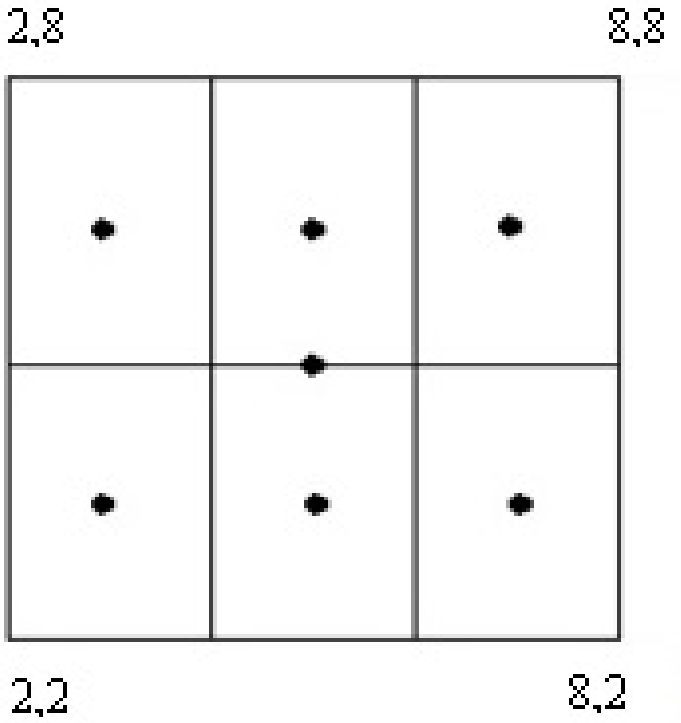


Figure 10 - Distribution of 7 nodes ideally inside a square polygon

Table 1. Formal comparison of proposed algorithm and existing works

Criteria / Algorithm	Complexity	Symmetric	Edge Intersection	Bend	Uniform Distribution
Simulated Annealing	$O(n^2m^2)$	NO	YES	YES	NO
Hybrid of SA & Skeleton	$O(n^2m^2)$	NO (better than SA)	YES (fewer than SA)	YES (fewer than SA)	NO (better than SA)
Orthogonal which uses Skeleton	$O(n^2 m^{2.2})$	YES	NO	$O(nm^{1.3})$	YES
Proposed Algorithm	O(nm)	YES	NO	NO	YES

4. Illustrative Examples

Using a proposed algorithm, a binary tree with 127 nodes was drawn inside a triangular polygon in Figure 11. Likewise Figure 12, 13, and 14 show the applicability

of the proposed algorithm in concave, convex, and L shaped polygons respectively. In Figures 11 to 14 all binary trees are drawn symmetrically, there is no any edge intersection, there is no any bend, and distribution of nodes is uniform.

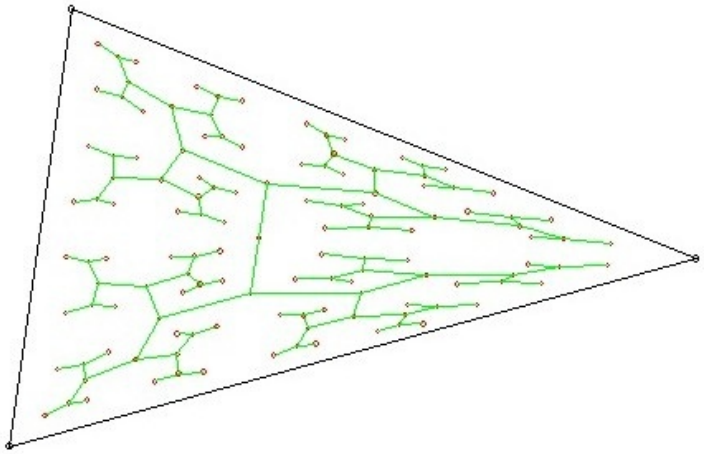


Figure 11 – drawing binary tree inside a triangle

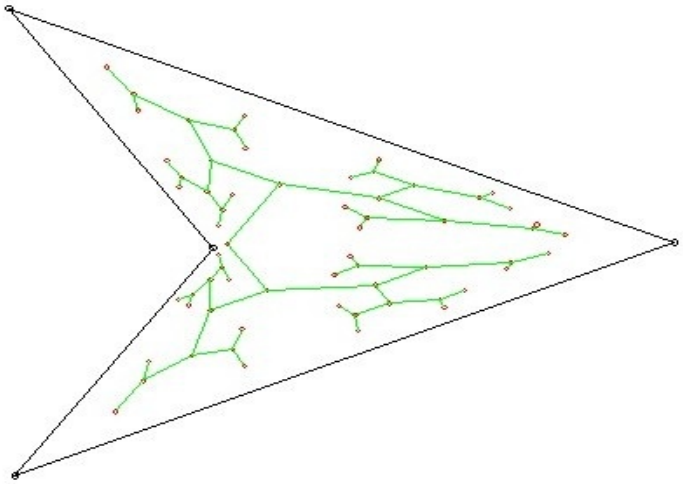


Figure 12 - drawing binary tree inside a concave polygon

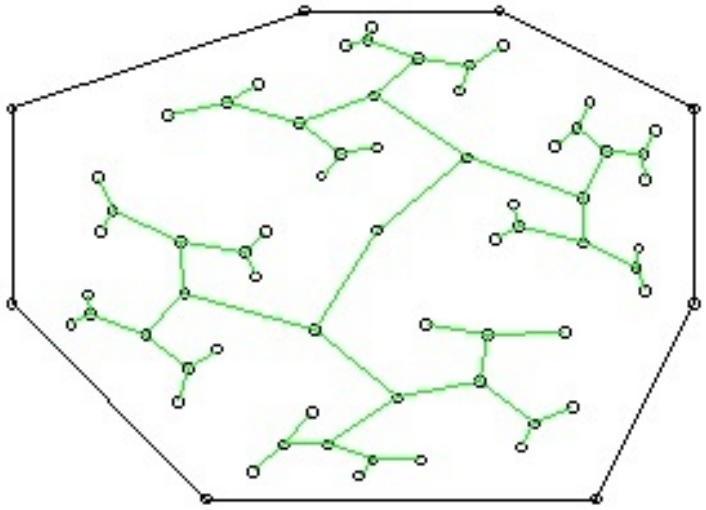


Figure 13 - drawing binary tree inside a convex polygon

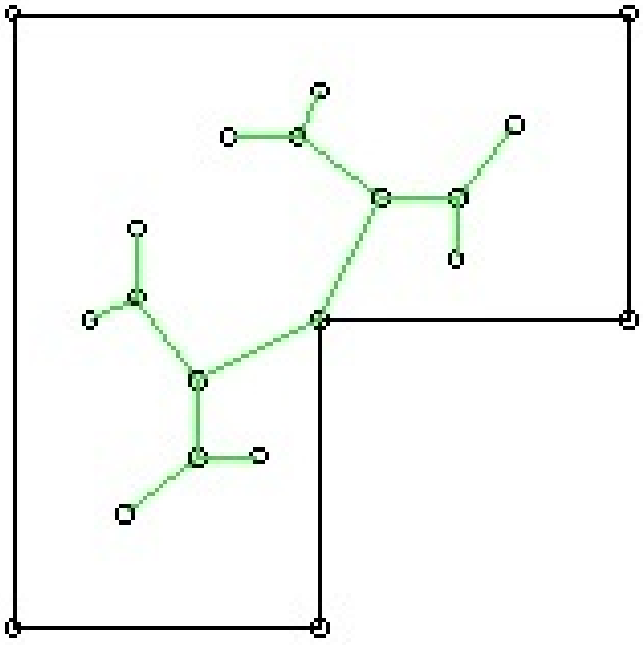
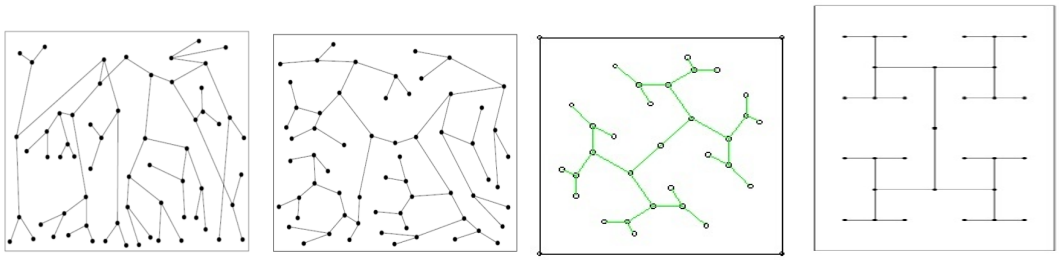


Figure 14 - drawing binary tree inside an L shaped polygon

Figure 15 provides an illustrative comparison between four approaches. Hybrid of SA & skeleton is better than SA, however the proposed algorithm is superior to SA and hybrid of SA & skeleton since not only it does not have any bends or edge intersections but also the nodes of binary tree have been distributed uniformly and also the symmetry in binary tree is obvious. The orthogonal approach which exploits skeleton acts acceptably in comparison with SA or hybrid of SA & skeleton however it suffers from three main limitations: 1- it is only suitable for rectangular polygons, 2- it has bends, 3- its complexity is more than the proposed algorithm.



(a) (b) (c) (d)

Figure 15- (a) SA (b) Hybrid of SA & Skeleton (c) Proposed Algorithm (d) Orthogonal which exploits skeleton

5. Conclusion and Future Work

A heuristic algorithm has been proposed to draw a binary tree inside an arbitrary polygon. The heuristic was based on calculating the center of gravity of an arbitrary polygon and the segmentation process. The center of gravity approach decreased the computations and kept the symmetry of the binary tree. The proposed algorithm does not have any bends and edge intersections. The proposed algorithm does not work well on concave polygons with hole and this is the main direction of future work of this research. Another trend is to compose center of gravity approach with skeleton.

Reference

[1] Bagheri A, Razzazi M., Drawing Free Tree inside Simple Polygons Using Polygon Skeleton, computing and informatics, vol.23, 239-254, 2004.

[2] Bagheri A, Razzazi M.,Minimum Height Path Partitioning of Trees, Computer Science of Engineering and Electrical Engineering, vol.17.no.2,pp.99-104.2010.

[3] Bagheri, A., Drawing Planar Graphs inside Simple Polygons, Ph.D Thesis, Amirkabir University of Technology, 2006.

[4] Bagheri A, Razzazi M., Drawing Complete Binary Trees inside Rectilinear Polygons, International Journal of Computer Mathematics, Vol. 87, No. 14, pp. 3138, November 2010.

[5] Bagheri A, Razzazi, Drawing Free Trees on 2D Grids Which are Bounded by Simple Polygons, Scietia Iranica, vol.13, no.4, pp 387-394,2014.

[6] Chiba, N., Onoguchi, K., and Nishizeki, T, Drawing Planar Graphs Nicely, Acta Informatica, 22, 1985, pp. 187-2010.

[7] Chistian A.Duncan, David Eppstein, michad T.Goad rich, stepher G.kobourouMartin nullenbury , lombovdi, Drawing of Graphs, Journal of Graph Algorithms and Applications,vol.16, no 1, pp.37-83, 2012.

[8] Davidson, R., and Harel, D., Drawing Graphs Nicely using Simulated Annealing, ACM Transactions on Graphics, Vol. 15, No.4, pp. 301-331. October 1996.

[9] Di Battista, G., Eades, P., Tamassia, R. and Tollis, I.~G, Algorithms for Drawing Graphs: an Annotated Bibliography, Computational Geometry: Theory and Applications, 4(5), pp. 235-282, 1994.

[10] Eppsteine D.,arrangement, Graphs in Small Grids or How to Play Planarity, Journal of Graph Algorithms and Applications , vol 8,no.2,pp.211-232.2014.

[11] Garay, M.~R., and Johnson, D.~S., Crossing Number is NP-Complete, SIAM J. Alg. Disc. Methods, 4(3), pp. 312-316, 1983.

[12] Garg, A., and Tamassia, R., On the Computational Complexity of Upward and Rectilinear Planarity Testing, Proc. 2nd Int. Symp. Graph Drawing (GD'94), Lect. Notes in Computer Science 894, pp. 286-297, 1995.

[13] Khosravinejad v., Bagheri a., A New Heuristic Algorithm for Constructing Steiner Trees inside Simple Polygons in Presence of Obstacles, ACSIJ Advances in Computer Science: AN international journal, vol.2, issue 3, no.4, 2013.

[14] kumar s., A Simple Algorithm for Steiner Tree Problem in Networks International journal of Applications or Innovation in IJAIEM, vol 3, issue 7, 2014.

[15] Norouzi tallalb m., Bagheri A. and Jahansire M., Heuristic Algorithms for Geometric Embedding of Complete Binary Trees on to a Point-set, Proceedings of the International Multi Conference of Engineers and Computer Scientists , vol 2,Hong kong, IMECS 2014.

[16] Jani, F., Drawing Flat Graphs Inside Rectangular Polygons With Hole, Master Thesis, Amirkabir University of Technology, February 2011.