Heuristic applied to the Euclidean Steiner tree problem with node-depth-degree encoding

Marcos Antônio Almeida de Oliveira[1], Telma Woerle de Lima[1], Les R. Foulds[1], Anderson Da Silva Soares[1], Alexandre Claudio Botazzo Delbem[2], and Clarimar Jose Coelho[3]

[1]Intitute of Informatics, Federal University of Goiás, Goiânia, GO, Brazil

[2]Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP, Brazil

[3]Departamento de Computação, Pontifícia Universidade Católica de Goiás, Goiânia, GO, Brazil

***Abstract*—In this paper a variation of the Beasley [1] algorithm for the Euclidean Steiner tree problem (ESTP) is presented. This variation uses the Node-Depth-Degree Encoding (NDDE), which requires an average time of $O(\sqrt{n})$ in operations to generate and manipulate spanning forests. For spanning tree problems, this representation has linear time complexity when applied to network design problems with evolutionary algorithms. Computational results are given for test cases available in [2] involving instances up to 50 vertices. These results indicate that NDDE can be used by other techniques besides evolutionary algorithms with good performance. Furthermore, it shows that the proposed algorithm has advantages in the solution found.**

Keywords: Euclidean Steiner Tree Problem, Heuristic Algorithm, Node-Depth-Degree Encoding

I. Introduction

Network design problems (NDPs) are present in several areas of science and engineering [3], [4]. A fundamental NDP is the Steiner tree problem [5]. This problem is one of the oldest in mathematical optimization [6]. Several NDPs and real applications can be represented as generalizations of the Steiner tree problem [3].

The problem of computing minimum Steiner trees is NP-complete [7] as well as the Euclidean Steiner tree problem (ESTP) [5], [8], which is one of its versions. In the case of NP-complete problems it is impossible to find an exact efficient algorithm due to its exponential complexity [5]. Exponential algorithms that can be used in practice are rare [5]. Although exact algorithms provide optimal solutions to the problem [9], [10], [11], it is appropriate to use heuristic algorithms[5]. Many heuristic algorithms have been developed for the problem, including the algorithm of Chang (1972)[12], the Thompson's method of minimum evolution (1973)[13], a hierarchical genetic algorithm of Barreiros (2003)[14], a recent randomized Delaunay triangulation heuristic of Laarhoven and Ohlmann (2010) [15], among others [16], [17], [1], [18], [19].

Most heuristic algorithms for ESTP perform a series of iterative improvements using the MST as a starting solution [20]. The first heuristic algorithms for ESTP proposed by Chang (1972) and Thompson (1973) are based on the iterative insertion of Steiner points in an MST[10], [13], [12]. Beasley (1992) [1] presents an heuristics for ESTP based on finding Steiner optimal solutions to the subgraphs of the Minimum Spanning Tree (MST) of the total number of vertices.

Besides the use of heuristic algorithms, another factor that can improve the performance of the algorithms is the representation used to encode solutions. Several optimization problems can be encoded by a variety of different representations, and the performance can vary greatly depending on the encoding used [23]. An adequate representation should at least be able to encode all possible solutions of an optimization problem [23]. One of the representations that have shown relevant results for encoding trees in graphs is the Node-Depth-Degree encoding (NDDE) proposed by Delbem et al. (2012) [22].

In general, the time complexity of operations for generating spanning trees for an effective representation is $O(n \log n)$[23]. NDDE requires an average time $O(\sqrt{n})$ in operations to generate and manipulate spanning forests. This efficiency boosts its use in heuristics developed for PPRs.

This paper proposes the use of a representation for spanning trees and forests. The possibility of applying the NDDE for the ESTP offers a contribution to several real-world applications that require more effective solutions. The NDDE is used to construct operators that implement the steps of the heuristic algorithm proposed in [1] for ESTP. Computational results are given for test cases available in [2] involving instances up to 50 vertices.

The next section gives a brief description of the Euclidean Steiner tree problem. Section III introduces the NDDE. Section IV presents the Beasley's heuristic algorithm and the algorithm proposed as a variation of this approach. Section V presents the obtained results. Final considerations are presented in section VI.

II. Euclidian Steiner tree problem (ESTP)

The Steiner tree problem consists in to find the minimum network that interconnects n points, by the addition of

auxiliary points to minimize the total cost [24], [1], [19]. An extra vertex that is added to a tree in order to reduce its cost is called Steiner point [24]. Details on the Steiner problem can be found in [9], [24], [10], [11], [6].

The minimum connection network is called Steiner minimal tree (SMT). SMT consists by set N with the n given points and set S with the s Steiner points added [24]. A T network connection is called Steiner tree if it satisfies three conditions: T is a tree; any two edges of T meet at an angle s of $120\,^{\circ}$; a Steiner point cannot be of degree 1 or 2 [10].

The topology of a tree defines the connection array of points of N and S. A minimum tree for a given topology is called relatively minimal. A topology is full if the number of Steiner points is $s=n-2$ [24]. A tree is named full Steiner tree (FST) when it's the relatively minimal tree for a given full topology [24].

The ESTP is the problem of finding the minimum length of an Euclidean tree, covering a set of fixed points in the plane, with the addition of Steiner points [19], [1]. This problem is defined to the Euclidean plane [11]. For two points $a(x_1, y_1)$ and $b(x_2, y_2)$ the Euclidian distance is:

$$d(a,b) = O(\sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2)}) \qquad (1)$$

Consider V a set of n points in the Euclidian plane. The solution to the ESTP is obtained through the minimum spanning tree (MST) of $V \cup S$ in which S is a set of Steiner points [1].

III. Node-Depth-Degree Encoding (NDDE)

The Node-Depth-Degree Encoding (NDDE) proposed in [22], is based on the concepts of vertex, depth and degree of a vertex in an undirected graph. The way of representing trees is composed of linear lists of triples (vertex, depth, degree). The order in which the triples are arranged in the list defines the edges that make up the tree. A vertex is always linked to the first preceding vertex with lower depth [22]. This order can be obtained by a depth search on the tree from a vertex chosen as the root [22].

Figure 1 illustrates an example of a spanning tree of a graph with 9 vertices. This spanning tree is represented by the NDDE presented in Table I.

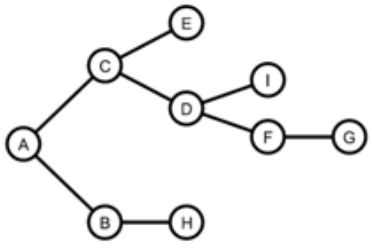

Figure 1: Example of a spanning tree with 9 vertex.

NDDE has two operators to generate new forests from an existing one. These operators have been applied in evolutionary algorithms with satisfactory results for network design problems [22]. This justifies the possibility of using NDDE in other techniques, such as heuristics for the ESTP.

Table I: Linear list containing in each column the NDDE triple representing the spanning tree presented in Figure 1.

Position	0	1	2	3	5	5	6	7	8
Vertex	A	B	H	C	D	F	G	I	E
Depth	0	1	2	1	2	3	4	3	2
Degree	2	2	1	3	3	2	1	1	1

IV. Proposed approach

Proposed approach consists in to develop the algorithm proposed in [1] with NDDE. In order to achieve this propose we need to construct operators to implement the steps of the heuristic algorithm of [1]. In the following paragraphs we present the heuristic algorithm of Beasley (1992), and modifications proposed in this paper.

A. Algorithm proposed by Beasley (1992)

Beasley (1992) [1] presents an heuristics for ESTP and the Rectilinear Steiner Tree Problem (RSTP). This heuristics is based on finding Steiner optimal solutions to the subgraphs of the Minimum Spanning Tree (MST) of the total number of vertices. The heuristic considers all subgraphs of the MST that contain four vertices, find an optimal Steiner tree for each subgraph. After it adds to set V the Steiner vertices obtained for these subgraphs [1].

This heuristic consists, basically, of the following steps. Consider V a set of vertices maintained by the algorithm. V is initialized from the given points and in the end of the algorithm is a set containing the initial points plus the Steiner points whose MST is the solution offered by the heuristic. Algorithm 1 presents the steps performed by the heuristic.

Algorithm 1: Algorithm of Beasley (1992)

1) Consider: $T(K)$ the cost of connecting K vertices through its MST; $T_S(K)$ the cost of connecting the same K vertices, through its Steiner minimal tree (SMT); V_0 the set of initial vertices and t the iteration counter. Configure $V = V_0$ and $t = 0$.
2) Find the MST of V. Attribute $t = t + 1$;
3) Enumerate the set L being all the subtrees of four vertices K in the MST.
4) For each $K \in L$ calculate the reduction of cost $R(K)$ that occurs when the vertices of K are connected via SMT.
5) If $max[R(K)|\ K \in L]=0$ stop.
6) Be M an empty set of vertices with each new iteration. Order in a decreasing way according to $R(K)$ the sets of vertices $K \in L$. In this ordered set, for each K, if $|M \cap K|=0$ and $R(K)>0$ do:
    a) Add the Steiner points associated with V.
    b) Set $M=M \cup K$

7) Find the MST of V and remove any vertex of V which represents a Steiner point with a degree less than or equal to two.
8) For each Steiner point with degree 3, move this vertex to its optimum location. Repeat this process until no cost reduction os possible.
9) Find the MST of V and for each subtree (V* of cost C*) that has the appropriate topology for FST:
    a) Apply the algorithm of Hwang [25] to find the FST of V*.
    b) If the cost of the FST is smaller than C* move the Steiner vertices to the locations given by the FST.
    c) If the cost of the FST is greater than C* try a cost reduction by removing Steiner vertices that are increasing the cost of V*.
10) Go back to step (2).

For more details on this algorithm see [1]. Beasley (1992) [1] shows computational results which indicates that this heuristic provides better quality solutions compared to other heuristics [12], [16]. The improvement of using this heuristic is typically 3 to 4% [1].

B. Algorithm with NDDE

We proposed a modification on Beasley algorithm [1] that consists in using the NDDE as a graph tree data structure. Another difference is that our algorithm does not implement step 9 from the original heuristic. The main purpose, of this implementation, is to demonstrate the use of NDDE to solve the ESTP without evolutionary algorithms. Another goal is to indicate the advantages of NDDE to implement some steps of the based heuristic.

NDDE is used to represent the tree topology. A graph structure represents the locations of the vertices in the set V defined for the Beasley heuristics. For this problem, we need to add three other informations to each node in NDDE. The Steiner value indicates if the node represents a Steiner point; the value parentIndex indicates the position of the parent node in the tree hierarchy; and childIndex indicates the position of the last child in the tree hierarchy.

The Steiner value contributes, in some operations applied only to Steiner points, in identifying these points. The parentIndex and childIndex columns help identifying the edges of a given node. One of the edges connects the node to parentIndex, the other should be between the position of that node and childIndex. For example, at one Steiner point, its degree should be 3 and the edges connecting this node should be in the positions parentIndex, node index + 1, and childIndex. Thus, these connections can be discovered without the need to iterate through the NDDE. We will describe below some operators implemented using the NDDE.

1) Sets query operator (SQO): The SQO described in the Algorithm 2 was used to implement step 3 of the Algorithm 1. The Algorithm 2 has as input the MST represented by NDDE, and returns the set L.

Algorithm 2: Sets query operator

1) Considering a, b, c and d, the four elements of the set K that will be added to L. For each MST node represented by the NDDE, starting from the last and going up to the first node of the tree, do:
    a) Consider a as the current node;
    b) If the degree of a is greater than or equal to three, get b, c and d by browsing the MST from the parentIndex column, add the new K to L;
    c) If the degree is greater than or equal to 1, configure b with the parentIndex of a, and try to find subtrees rooted at b such that c is the child of b and d is the child of c; Add all possible K to L;
    d) If the degree of a is greater than or equal to three, get all possible star type subtrees, in which b may be the parentIndex of a or any of its children, and c and d are children of a; Add all possible K to L;

2) Node Removal Operator (NRO): The NRO described in the Algorithm 3 was used to implement step 7 of the Algorithm 1. In this step the Steiner nodes added that has degree less than 3 must to be removed from the tree. The Algorithm 3 has as input the MST represented by the NDDE and the set V returns the new MST.

Algorithm 3: Node Removal Operator

1) While it is possible to remove nodes:
    a) for each node i of MST, if the node represents a Steiner point with degree lower than three:
        i) Remove the vertex of the set V;
        ii) Move the nodes of the subtree rooted at i to the immediately preceding position, decreasing the depth of each node by 1;
        iii) If i does not have a child, decrease 1 degree from the parent of i;
        iv) Move all subsequent nodes that are not subtree of i to the immediately preceding position on the list;
        v) Update the parentIndex and childIndex of the nodes after i.

One of the contributions to the algorithm efficiency attributed to the use of NDDE is the operator SQO. Subtrees of four nodes are found in three possible ways. The first easily finds the subtrees by browsing the parentIndex value. In the second and third cases the NDDE limits the search of subtrees on the list until the index indicated by the childIndex of the node. The NDDE is essential to keep indexes parentIndex and childIndex updated, and also to identify when it is possible to perform each of these operations.

With the NDDE the update operation of parentIndex and childIndex indexes can be easily done through an iteration that traverses the tree and gets these indexes from a dynamic

vector that stores the last indexes of each depth. Thus, for every node of the current iteration parentIndex can be found from the last index of the previous depth, at the same time this element, corresponding to parentIndex, will have the index of the current iteration node as childIndex.

In the NRO operator the node is easily removed, dragging the elements to the previous position from the first element subsequent to the removed one. The NDDE maintains its structure with only three exceptions: In case the element removed is a leaf, the parent node must have its degree reduced by one unit; the element that composes a subtree of the removed node should have their degrees decreased by one unit in order to become subtree of the parent of the node removed; The parentIndex and childIndex indexes must be updated. These two operators already indicate some advantages in using NDDE.

V. Results

The algorithm presented in this paper was implemented in C language. The tests were run on an Intel (R) Core (TM) i3-2120 CPU @ 3.30 GHz with 4GB of RAM and Operating System Windows 7.

Test cases were the same used by Beasley (1992) available at [2]. Each test case consists of 15 different instances of *n* points in the Euclidean plane with two dimensions. These *n* points, represented by the pair *(x, y)*, have values ranging from 0 to 1, accurate to 7 decimal places.

Follows a comparison between the results of the algorithm of Beasley (1992) [1] and the proposed approach. The results show the minimum, average, and maximum values of the different graphs for each number of nodes. Table II shows the number of iterations until the algorithm stops. We can observe that the algorithm with the NDDE expends a high number of iterations in average and maximum than Beasley algorithms. Probably, this result is a consequence of the removal of step 9.

Table II: Comparison of the number of iterations.

Number of iterations.						
	Beasley			Algorithm with NDDE		
N	Min.	Avg.	Max.	Min.	Avg.	Max.
10	2.000	3.000	4.000	2.000	6.933	40.000
20	3.000	3.600	5.000	3.000	14.200	35.000
30	3.000	3.533	5.000	3.000	10.000	21.000
40	3.000	5.133	21.000	4.000	12.600	35.000
50	3.000	4.067	7.000	3.000	12.333	23.000

Table III shows the percentage of reduction achieved given by

$$100(T(V_0)\text{-}T(V_F))/T(V_0) \quad (2)$$

, where $T(V_0)$ is the cost of the minimum spanning tree and $T(V_F)$ is the cost of the final solution. Table III also present the percentage of reduction between the MST and the optimum SMT. We can observe that the proposed algorithm provides solutions very close to the optimal solution and the solutions obtained by the Beasley algorithm.

Table III: Comparison of the percentage of reduction in relation to the MST.

Percentage of reduction.							
	Beasley			Algorithm with NDDE			Optimum SMT
N	Min.	Avg.	Max.	Min.	Avg.	Max.	
10	0.477	3.138	6.168	0.477	3.140	6.168	6.168
20	1.389	3.015	4.737	1.384	2.958	4.367	4.758
30	2.059	2.868	4.752	2.059	2.841	4.639	4.964
40	1.669	3.024	4.174	1.643	3.023	4.135	4.407
50	2.138	2.841	3.620	2.053	2.806	3.687	3.786

Table IV presents the number of Steiner points add by the algorithms. We can observe that both algorithms added the same number of Steiner points in most of the cases tested. It shows that the removal of step 9 did not effectively affect the number of Steiner points added.

Table IV: Comparison between the number of Steiner points.

Number of Steiner Vertices.						
	Beasley			Algorithm with NDDE		
N	Min.	Avg.	Max.	Min.	Avg.	Max.
10	2.000	3.400	6.000	2.000	3.400	6.000
20	6.000	7.667	10.000	6.000	7.933	10.000
30	9.000	11.667	16.000	9.000	11.800	15.000
40	13.000	15.733	18.000	13.000	15.733	18.000
50	16.000	19.333	22.000	16.000	19.533	22.000

VI. Conclusions

In this work we show that is possible to apply the NDDE to the Euclidean Steiner tree problem with favorable results. We want to highlight that the use of NDDE reduces the running time of the algorithm. In Beasley approach, the test were executed in a supercomputer and in our approach we use a personal computer and still can reach the results in some seconds. In addition to this new class of problems, it was also found that the NDDE can be applied to other heuristic techniques. Up until now, it had only been applied to Evolutionary Algorithms.

VII. Acknowledgment

Thanks the financial support from Fundação de Amparo à Pesquisa do Estado de Goiás (FAPEG).

References

[1] J. E. Beasley, "A heuristic for Euclidean and rectilinear Steiner problems," *Eur. J. Oper. Res.*, vol. 58, pp. 284-292, 1992.

[2] J. E. Beasley, "OR-Library: distributing test problems by electronic mail," *Eur. J. Oper. Res.*, vol. 41, pp. 1069-1072, 1990.

[3] J. Smith, *Generalized Steiner network problems in engineering design*. In: Design optimization., Ed. Orlando, United States of America: Academic Press, Inc, 1985.

[4] A. Delbem, A. de Carvalho, and N. Bretas, "Main chain representation for evolutionary algorithms applied to distribution system reconfiguration," *Power Syst. IEEE Trans.*, vol. 20, pp. 425-436, 2005.

[5] Michael R. Garey, and David S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness* Ed. New York, USA: W. H. Freeman & Co., 1990.

[6] E. N. Gordeev, and O. G. Tarastsov, "The Steiner problem: A survey," *Discrete Math. Appl.*, vol. 3, pp. 339-364, 1993.

[7] M. R. Garey, R. L. Graham, and D. S. Johnson, "The Complexity of Computing Steiner Minimal Trees," *SIAM Journal on Applied Mathematics*, vol. 32, pp. 835-859, 1977.

[8] M. Garey, and D. Johnson, "The rectilinear Steiner tree problem is NP-complete," *SIAM Journal on Applied Mathematics*, vol. 32, pp. 826-834, 1977.

[9] Z. A. Melzak, "On the problem of Steiner," *Canadian Mathematical Bulletin*, vol. 4, no. 2, pp. 143-148, 1961.

[10] F. Hwang, and D. Richards, "Steiner tree problems," *Networks*, vol. 22, pp. 55-89, 1992.

[11] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner Tree Problem* Ed. Netherlands: Elsevier Science Publishers B. V., 1992.

[12] S. Chang, "The generation of minimal trees with a Steiner topology," *Journal of the ACM JACM*, vol. 19, no. 4, pp. 699-711, 1972.

[13] E. A. Thompson, "The method of minimum evolution," *Annals of human genetics*, vol. 36, no. 3, pp. 333-40, 1973.

[14] J. Barreiros, "An hierarchic genetic algorithm for computing (near) optimal Euclidean Steiner trees," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2003, p. 8.

[15] J. W. Laarhoven, and J. W. Ohlmann, "A randomized Delaunay triangulation heuristic for the Euclidean Steiner tree problem in Rd," *Journal of Heuristics*, vol. 17, no. 4, pp. 353-372, 2010.

[16] J. M. Smith, D. T. Lee, and J. S. Liebman, "An O(n log n) heuristic for steiner minimal tree problems on the euclidean metric," *Networks*, vol. 11, no. 1, pp. 23-39, 1981.

[17] D. Du, and Y. Zhang, "On better heuristics for Steiner minimum trees," *Mathematical Programming*, vol. 57, no. 1-3, pp. 193-202, 1992.

[18] J. E. Beasley, and F. Goffinet, "A delaunay triangulation-based heuristic for the euclidean steiner problem," *Networks*, vol. 24, no. 4, pp. 215-224, 1994.

[19] D. Dreyer, and M. Overton, "Two heuristics for the Euclidean Steiner tree problem," *Journal of Global Optimization*, pp. 1-14, 1998.

[20] M. Zachariasen, "Algorithms for Plane Steiner Tree Problems," PHD Thesis, University of Copenhagen, Copenhagen, Denmark, March 1998.

[21] S. Fortune, *Voronoi diagrams and Delaunay triangulations*. In: Handbook of discrete and computational geometry., Ed. United States of America: CRC Press, Inc., 1977.

[22] Alexandre C.B. Delbem, Telma Woerle de Lima, and Guilherme P. Telles, "Efficient forest data structure for evolutionary algorithms applied to network design," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 6, pp. 829-846, 2012.

[23] Franz Rothlauf, *Representations for genetic and evolutionary algorithms*, 2nd ed., Ed. Netherlands: Springer-Verlag, 2006.

[24] E. N. Gilbert, and H. O. Pollak, "Steiner Minimal Trees," *SIAM Journal on Applied Mathematics*, vol. 16, no. 1, pp. 1-29, 1968.

[25] F. Hwang, "A linear time algorithm for full Steiner trees," *Operations Research Letters*, vol. 4, no. 5, pp. 235-237, 1986.