Neighborhood Hybrid Structure for Minimum Spanning Tree Problem

Beatriz Martínez Bahena[1], Marco Antonio Cruz-Chávez[2], Ocotlán Díaz-Parra[3], Martín Gerardo Martínez Rangel[3], Martín H. Cruz Rosales[4], Jesús Del Carmen Peralta Abarca[5], Jazmín Yanel Juárez Chávez[1]

[1]Postgrado en Ingeniería y Ciencias Aplicadas, [2]CIICAp, [3]FCAeI, [4]FC,[5]FCQeI, Universidad Autónoma del Estado de Morelos, Avenida Universidad 1001. Col. Chamilpa, C.P. 62210. Cuernavaca, Morelos, México.

[2]mcruz@uaem.mx

Abstract

An experimental analysis of five neighborhoods is presented. The analysis includes a hybrid structure comprised of one random pair, two random pairs, three random pairs or four random pairs. The efficiency and effectiveness of each structure was tested using the minimum spanning tree problem. As proposed in this research paper, the hybrid structure approach applied to the minimum spanning tree problem demonstrates superior effectiveness and competitive efficiency as compared to other structures.

1. Introduction

The minimum spanning tree problem (MST) is a combinatorial optimization problem, and one of the most important problems in the field of distributed computing and communications networks [1]. It was formulated in 1926 by Otakar Borukva, in his attempt to find the cheapest way to distribute electric power in southern Moravia, Czech Republic [2]. The formulation of this problem has been useful for research in various fields, such as electrical and hydraulic systems, transportation, telecommunications network design, computer systems, telephone systems and other operations research problems where the goal is to optimize costs, distances, lengths or other measures between the points of consumption [3]. According to the complexity theory, the MST problem is classified as type P [4, 5].

This paper presents the analysis of five neighborhood structures with the minimum spanning tree problem. In this analysis, each neighborhood structure is implemented within an iterated local search algorithm (ILS). The algorithm is evaluated in efficiency and effectiveness for each neighborhood structure. Based on experimental testing, the best neighborhood structure is determined.

The proposed neighborhood hybrid structure is made up of the four structures that are analyzed, which are one random pair, two random pairs, three random pairs and four random pairs. The use of the neighborhood hybrid technique allows a better exploitation of the solution space, resulting in better solutions. Hybrid structures have been employed in different optimization problems [6, 7, 8], but there was not previous research found using hybrid structures for the minimum spanning tree problem.

This article is divided into the following sections. Section two defines the minimum spanning tree problem. Section three presents a neighborhood hybrid structure, as well as the performance of each neighborhood structure used. Section four explains the iterated local search. Section five details the experimental testing with ILS using the five neighborhood structures and section six presents the conclusions obtained in this research.

2. Minimum Spanning Tree Problem

The minimum spanning tree problem in the literature is represented by a graph [9, 10], which is defined as an undirected graph, connected and weighted G= (V, E), where $V= \{v_1, v_2 \ldots . v_n\}$ is a finite set of vertices and $E = \{e_{ij} \mid e_{ij} = (v_i, v_j), v_i, v_j \in V\}$ is a finite set of edges. It is said that a graph is weighted if each edge has an associated positive real number denoted by $W = \{w_{ij} \mid w_{ij} = w(v_i, v_j), w_{ij} > 0, v_i, v_j \in V\}$ representing distance, length, cost or another measure. The graph is undirected because the edges do not have a direction. A graph is connected if all edges are connected. Figure 1 shows an example of an undirected, connected and weighted graph.

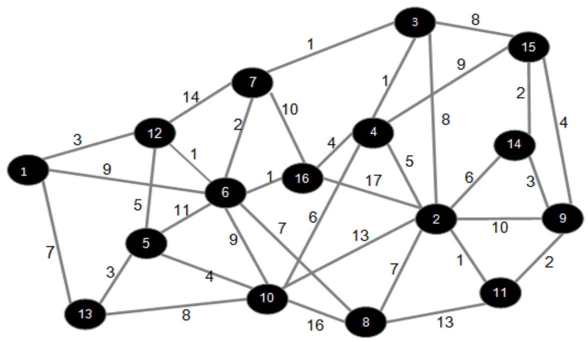

Figure 1. Undirected graph, connected with 35 vertices, 16 edges, with random cost in each edge

In terms of graphs, when finding a minimum spanning tree, certain conditions must be met [11]:

1. Being a subgraph of G with no cycles with n-1 edges, where *n* is the total number of vertices.
2. Being a subgraph of G where all vertices are connected.
3. The total of the costs of all edges associated with the subgraph is the minimum.

Figure 2 shows the minimum spanning tree for the graph presented in Figure 1.

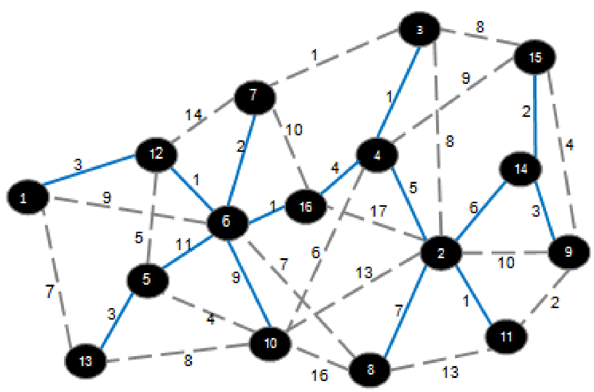

Figure 2. Example of a minimum spanning tree

The minimum spanning tree problem can be formulated by the following mathematical model [10]:

$$\min c = \sum_{e \in E} w_e x_e \quad (1)$$

$$s \quad t:$$

$$\sum_{e \in E} x_e = n - 1 \quad (2)$$

$$\sum_{e \in (S,S)} x_e \leq |S| - 1 \qquad \forall S \subseteq V \quad (3)$$

$$x_e \in \{0,1\} \qquad \forall e \in E \quad (4)$$

Equation (1) is the objective function of the problem, to minimize the total cost of all edges that form the minimum spanning tree. The set of constraints (2) requires that all vertices be connected, which implies that the total of all edges is equal to *n-1*, where *n* represents the total number of vertices, therefore no cycles are permitted. The set of constraints (3) prohibits the edges of E_T from forming cycles, where *(S, S)* notes that all the edges that go from a vertex in the set *S* must connect to another vertex in the set *S*. The set of constraints (4) indicates whether an edge connects to a pair of vertices or not, if x_e=1, then the edge connects a vertice *i* with a vertice *j*, otherwise x_e=0.

3. Neighborhood Hybrid Structure

The neighborhood structures are techniques used for the purpose of improving a solution, in which it is necessary to move step by step from an initial solution toward a neighboring solution that provides the minimum or maximum value of the objective function [12]. These techniques are used in optimization problems, which allow a better exploration of the solution space through its implementation within a local search algorithm.

A neighborhood is defined as a set of all those solutions which may be achieved from an initial solution *s*, through a movement σ during the exploitation of the solution space [4]. The movement σ can be a permutation, insertion or deletion between elements that form the solution *s*. The type of movement defines the kind of structure and size of the neighborhood [13].

A neighborhood structure is defined as a function *N(s)* presented in equation (5)

$$N(s) = \{s' \in S : s \xrightarrow{\sigma} s'\} \quad (5)$$

A neighborhood function *N(s)* specific for each solution $s \in S$ is a set $N(s) \in S$, called the neighborhood of *s*. This indicates that each solution *s'* is a neighborhood of *s* if $s' \in N(s)$. *S* represents the total set of possible solutions of an instance of the problem.

The movements made by each of the neighborhood simple structures are applied to the MST problem. These structures also form the neighborhood hybrid structure. The neighborhood structures are important because they allow a better exploitation of the solution space.

A Random Pair *[4, 6, 14, 15]*. This procedure starts with a feasible solution *s* from which is chosen a random number *num1*, which is considered the root. Then another random number *num2* is chosen, which is considered to be a neighbor vertex. A perturbation is performed; this movement generates a single cycle and removes an edge belonging to the cycle generated. If by removing that edge, vertices are not connected, the removed edge is reconnected and another edge is removed. The process continues until no disconnected vertices are left. Figure 3 shows the movement performed by the structure of a random pair.

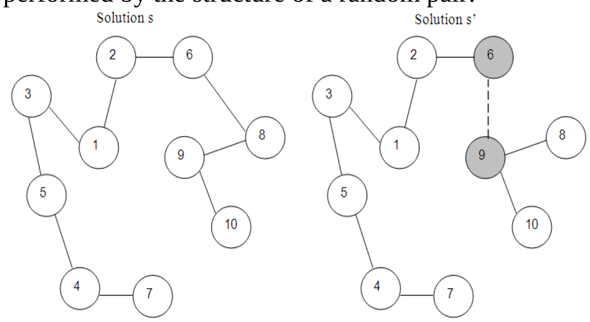

Figure 3. Neighborhood structure of one random pair

Two random pairs. The same procedure described for a random pair is carried out in the case of two random pairs [6, 15, 16, 17]. The only difference is that two random numbers are generated and considered a root, and two other random numbers are considered neighboring vertices. Therefore, two edges are removed corresponding to the numbers generated. Figure 4 shows the movement performed by the structure used.

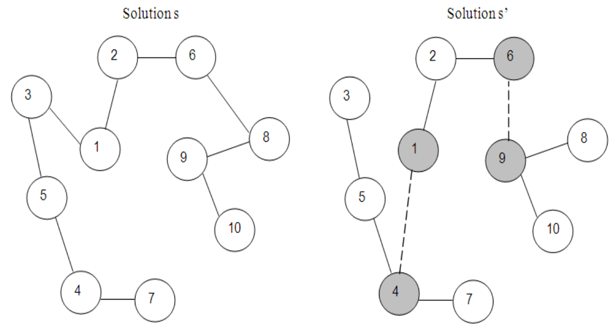

Figure 4. Neighborhood structure of two random pairs

Three random pairs. For a neighborhood structure with three random pairs [18], it is necessary to generate three random numbers which are considered root, and three other random numbers which are considered neighboring vertices. Three edges are removed, corresponding to the numbers generated. Figure 5 shows the movement performed by the neighborhood structure of three random pairs.

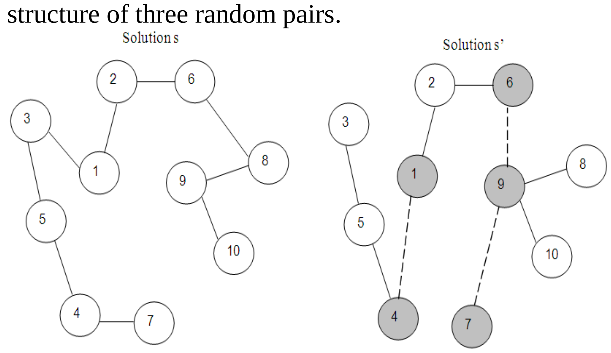

Figure 5. Neighborhood structure of three random pairs

Four random pairs. The same procedure described for the previous movements is used, with the generation of four random numbers [18] which are considered root and four other random numbers which are considered neighboring vertices. Four edges are removed belonging to the numbers generated.

Considering the roles of neighborhood structure presented above and their performance reported in the literature [6, 7, 8], the development of a **neighborhood hybrid structure** was proposed. Figure 6 shows the neighborhood hybrid structure in general. This structure is a combination of the individual structures already explained, in which the type of movement applied is determined at random during the execution of the algorithm.

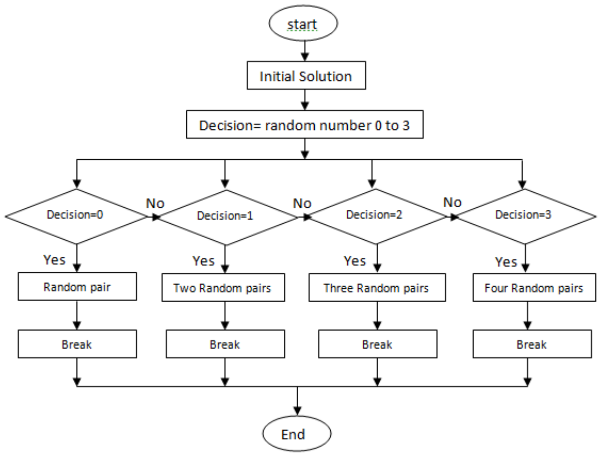

Figure 6. Flowchart of neighborhood hybrid structure

Input: data structure
CS_ILS=M; // *where M has a large value*
Do
Generates initial solution *s*
Do
Obtain s'= $N(s) = f(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$
If (*f (s')* ≤ *f (s)*) then
s'= the best solution so far
s = *s'*
End if
While stop criterion LS // *LS =Local Search*
If (*f(s')* ≤ *f(CS_ILS)*) then
CS_ILS= s'
End-if
While stop criterion ILS // *ILS=Iterated Local Search*
Output: solution of MST

Figure 7. Iterated local search general algorithm

4. Iterated Local Search with Neighborhood Hybrid Structure

The procedure implemented for analysis of the neighborhood structure is based on a local search method (*ILS*). ILS is a heuristic that iteratively applies a local search method [19]. Hoos and Stützle [20] claim that *ILS* is one of the simplest and most effective methods to avoid entrapment in local optima.

Figure 7 shows the iterated local search general algorithm. The iterated local search procedure requires a neighborhood structure and an objective function that maximizes or minimizes. The process starts with any solution *s* and the set of solutions in the neighborhood *N(s),* from which a solution is chosen *s'* through a movement σ. The type of movement σ applied to choose a neighbor defines the neighborhood structure. In this case there are four different types of movements in the neighborhood hybrid structure, $\sigma = \{\sigma_1, \sigma_2, \sigma_3\ \sigma_4\}$. This movement σ is performed by a stochastic process, which improves the objective function. If it is necessary to minimize, then *f (s')* ≤ *f (s)*. If this is true, then solution *s* is replaced by solution *s'* to improve it. This is repeated until the stop criterion of local search is met. New local searches are performed iteratively. Each time a local search terminates, it evaluates f (s') <= *f* (CS_ILS). If it is true, the local solution s' is replaced by the best solution CS_ILS that has been obtained so far. This procedure continues until the solution is not further improved. The stop criterion of ILS is the maximum number of executions of local searches. This work was also implemented in ILS separately for each neighborhood structure to conduct experimental testing.

5. Experimental Results

Experimental tests of the iterated local search were conducted in a PC with a processor of 3.17GHz, 2 GB RAM, Windows Vista Ultimate O.S and a compiler Visual C++ 2008. The test instances were generated randomly, from 100 and 200 vertices. Thirty executions were realized for each neighborhood structure. A stop criterion for ILS was set at a total of 100 iterations because after that number there were no significant improvements in the objective function (see Equation 1).

5.1. Effectiveness Tests

Table 1 shows that the structure with the least effectiveness is one random pair. Of the 30 tests, the best solution was found by the hybrid structure. The four random pairs structure found the worst solution of highest quality, and also showed better performance when comparing the average and standard deviation. The three pair structure has the best standard deviation. The hybrid structure takes second place of the five.

Table 1. Results for 100 vertices with 30 executions of ILS for each structure

Structure	Best solution	Worst solution	Average	Std. Dev.
Random pair	3937	5008	4155.2	132.06
Two random pairs	3776	5157	4094.1	134.68
Three random pairs	3806	5144	4037.9	126.48
Four random pairs	3773	**4900**	**4013.7**	**109.16**
Hybrid	**3769**	4941	4056.9	122.66

Table 2 shows that the least effective structure was the one random pair structure. Of the 30 tests, the best solution was found by the hybrid structure, this structure also had the best average. The four random pairs structure performed best regarding the worst solution of best quality. The three random pairs structure showed the best standard deviation. In this case the hybrid structure is second of the five.

Table 2. Results for 200 vertices with 30 executions of ILS for each structure

Structure	Best solution	Worst solution	Average	Std. Dev.
Random pair	8178	10221	8630.7	187.47
Two random pairs	7995	9969	8493.4	200.68
Three random pairs	8162	9912	8515.4	**158.97**
Four random pairs	8088	**9776**	8490.8	183.05
Hybrid	**7987**	10095	**8468.5**	174.20

5.2. Efficiency Tests

Figure 8 shows the average run time of the MST problem for instances of 100 and 200 vertices with 30 tests for each instance, applied to each neighborhood structure. Figure 8 shows that in the instance of 200 vertices the execution time for each neighborhood structure increases significantly when compared to the instance of 100. This increase is due to the increase in the size of the solution space for the instance of 200.

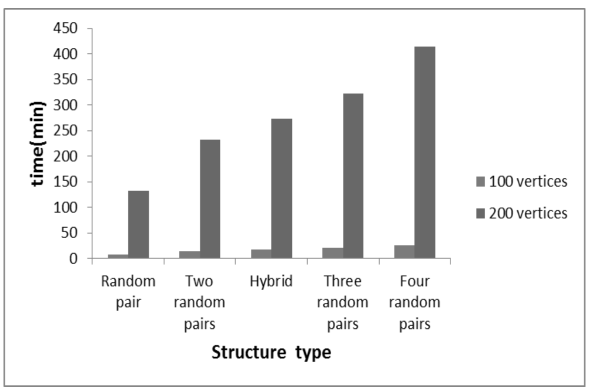

Figure 8. Run time with 100 and 200 vertices for each structure

Figure 8 shows that the random pair structure is most efficient for both the instances of 100 and 200 vertices, because it has the best average running time. The hybrid structure proponed in this research shows competitive behavior in both cases, although it does not show the best efficiency, it does not show the worst either. The hybrid structure is located at an intermediate point with respect to the other structures. The behavior of the hybrid structure is logical, due to the fact that it consists of the four other structures chosen randomly. The structure with the worst efficiency is that of four random pairs, which is required to perform a larger number of movements, obtaining the new solution s' requires more computational effort.

Future Work

In order to attempt to improve the performance of these algorithms, future work could involve implementation of the neighborhood hybrid structure to metaheuristics as simulated annealing, tabu search and others.

6. Conclusion

The neighborhood hybrid structure works effectively for the minimum spanning tree problem. This analysis is based on the results obtained in the experimental tests for the neighborhood with 100 and 200 instances. The efficiency of the hybrid structure is competitive with respect to the rest of the structures tested.

7. References

[1] Maleq, K. M. A., "Distributed Approximation Algorithms for Minimum Spanning Trees and other Related Problems with Applications to Wireless Ad Hoc Networks", Doctoral Thesis, 2007.

[2] Borůvka, O. "Příspěvek k řešení otázky ekonomické stavby elektrovodních sítí (Contribution to the solution of a problem of economical construction of electrical networks)" (in Czech). Elektronický Obzor 15: 153–154 (1926).

[3] Hillier, F. S., Lieberman, G. J., *Introducción a la Investigación de Operaciones*, Novena Edición, ISBN: 978-607-15-0308-4,2010.

[4] Papadimitriou, C. H., Steiglitz, K., *Combinatorial Optmization. Algorithms and Complexity.* ISBN: 0-486-40258-4. USA, 1998.

[5] Talbi, El-Ghazali., "Metaheuristics: from design to implementation", ISBN 978-0-470-27858-1, 2009.

[6] Cruz-Chávez, M.A, Martínez-Oropeza, A., Serna Barquera, S. A, "Neighborhood Hybrid Structure for Discrete Optimization Problems", *Electronics, Robotics and Automotive Mechanics Conference, CERMA2010, IEEE*

Computer Society, ISBN 978-0-7695-4204-1, pp. 108-113, 2010.

[7] Hansen, P., Mladenovic, N., "Variable Neighborhood Search: Principles and Applications", *European Journal of Operational Research 130*, Montréal, Canada, pp 449-467, 2001.

[8] Arajy, Y. Z., Abdullah, S., "Hybrid Variable Neighbourhood Search Algorithm for Attribute Reduction in Rough Set Theory", Data Mining and Optimisation Research Group (DMO) Centre for Artificial Intelligence Technology Universiti Kebangsaan Malaysia, 43600 Bangi Selangor, Malaysia, ISSN: 978-1-4244-8136-1, 2010.

[9] S.P.V., S.J. P., "A Novel Minimum Spanning Tree Based Clustering Algorithm for Image Mining", *European Journal of Scientific Research* © EuroJournals Publishing, Inc. ISSN 1450-216X Vol.40 No.4 pp.540-546 , 2010.

[10] Hochbaum, D. S., Moreno-Centeno, E., *Network Flows and Graphs*. Berkeley IEOR 266. Aug 28 - Sep 11, 2008.

[11] Cotta, C., Cowling P., "Evolutionary Computation in Combinatorial Optimization", *9th European Conference, EvoCOP 2009 Tubingen*, Germany, 2009.

[12] Cruz-Chávez, M. A., Rivera-López, R., "A Local Search Algorithm for a SAT Representation of Scheduling Problems", *Lecture Notes in Computer Science, Springer-Verlag Pub*., Berlin Heidelberg, ISSN: 0302-9743, Vol.4707, No. 3, pp. 697-709, 2007.

[13] Martinez, M. A. A., "Algorithm based in Tabu Search for Calculating the Index of a Graph Transmission" (in Spanish), Vol. 1, No. 1, pp. 31-39, 2006.

[14] Kenneth D. B., Cost Versus Distance in the Traveling Salesman Problem. UCLA Computer Science Dept., Los Angeles. CA 90024 1596 USA, https://eprints.kfupm.edu.sa/ 32861/ 1/32861.pdf

[15] González-Velázquez Rogelio, Bandala Garcés M. A. Hibrid Algorithm: Scaling Hill and Simulated Annealing to Solve the Cuadratic Allowance Problem. 3er Taller Latino Iberoamericano de Investigación de Operaciones. Acapulco, Guerrero, México, 2009.

[16] Martin O., Otto S. W. and Felten E. W., " Large Step Markok Chains for the Traveling Salesman Problem" Complex Systems 5(3). pp 299-326, 1991.

[17] Martin O., Otto S. W. and Felten E. W., "Large Step Markov Chains for the TSP Incoporating Local Search Heuristics". Operations Res. Letters 11, pp 219-224, 1992.

[18] Martinez, B. B., "Solution to Problem Minimum Spanning Tree applying Simulated Annealing with Tabu Search" (in Spanish), Master's Thesis, Center of Engineering and Applied Science-UAEM, 2011.

[19] Micheli, M., "Iterated local search algorithm for the scheduling of parts in a hybrid flow shop system" (in Spanish), Polytechnic University of Catalonia, Department of Management, 2009.

[20] Hoos, H.H.; Stützle, T., *Stochastic local search: foundations and applications*, Morgan Kaufmann Publishers, San Francisco, CA, 2005.