A MULTI-POPULATION GENETIC ALGORITHM FOR TREE-SHAPED NETWORK DESIGN PROBLEMS

Dalila B. M. M. Fontes and José Fernando Gonçalves
Faculdade de Economia and LIAAD-INESC Porto L.A., Universidade do Porto
Rua Dr. Roberto Frias, 4200-464 Porto, Portugal

Keywords: Multi-population genetic algorithms, Random keys, Network design.

Abstract: In this work we propose a multi-population genetic algorithm for tree-shaped network design problems using random keys. Recent literature on finding optimal spanning trees suggests the use of genetic algorithms. Furthermore, random keys encoding has been proved efficient at dealing with problems where the relative order of tasks is important. Here we propose to use random keys for encoding trees. The topology of these trees is restricted, since no path from the root vertex to any other vertex may have more than a pre-defined number of arcs. In addition, the problems under consideration also exhibit the characteristic of flows. Therefore, we want to find a minimum cost tree satisfying all demand vertices and the pre-defined number of arcs. The contributions of this paper are twofold: on one hand we address a new problem, which is an extension of the well known NP-*hard* hop-constrained MST problem since we also consider determining arc flows such that vertices requirements are met at minimum cost and the cost functions considered include a fixed cost component and a nonlinear flow routing component; on the other hand, we propose a new genetic algorithm to efficiently find solutions to this problem.

1 INTRODUCTION

There are many applications where it is necessary to find an optimal spanning tree. In the minimum spanning tree (MST) one wishes to find a least-cost tree spanning all vertices in a network. The MST problem arises in many applications in, e.g., communication networks. Many polynomial-time algorithms have been proposed for this problem by, for example, Dijkstra, Kurskal, Prim (Cormen et al., 2001). Although the basic MST problem can be solved in polynomial-time, the addition of one or more constraints often transforms it into problems, such as the MST with a pre-defined number of arcs in any path from the root vertex, which have been shown to be NP-hard, see e.g., (Dahl et al., 2006)). For these problems, a heuristic must be used, at least on larger problem instances. Genetic algorithms (GAs) are examples of search heuristics that have been successfully applied to such problems.

The MST with a pre-defined number of arcs has numerous practical applications and is frequently encountered in network design problems, for example in computer networks we can find the multicast-routing problem (see, e.g., (Deering et al., 1994)), where a number of clients and a server are connected by a common communication network. The server wishes to transmit identical information to all clients, and does so by transmitting the data to the vertices it directly connects to, and these latter vertices forward incoming data to their respective children in the tree. The limit on the number of arcs in each path is usually used to guarantee a certain quality of service with respect to availability and reliability constraints (LeBlanc and Reddoch, 1990; Woolston and Albin, 1988), as well as lower delays (Gouveia and Requejo, 2001), since they limit the number of arcs in each path from the central service provider. (Woolston and Albin, 1988) have shown that, by including this type of constraints it is possible to generate network designs with a much better quality of service and with only a marginal increase in the total cost.

Recently, Genetic Algorithms (GA) and other Evolutionary Algorithms (EAs) have been successfully applied to solve constrained spanning tree problems of the real life instances; and have also been used extensively in a wide variety of communication network design problems (Gen et al., 2001; Gen et al., 2005). For example, some authors have proposed GAs for the capacitated MST problem (Ahuja

B. M. M. Fontes D. and Fernando Gonçalves J. (2009).
A MULTI-POPULATION GENETIC ALGORITHM FOR TREE-SHAPED NETWORK DESIGN PROBLEMS.
In *Proceedings of the International Joint Conference on Computational Intelligence*, pages 177-182
DOI: 10.5220/0002281101770182

and Orlin, 2001; Lacerda and Medeiros, 2006), while others propose GAs for the degree-constrained MST (Han et al., 2005; Zeng and Wang, 2003) for a hybrid GA (with local search). Other researchers have investigated different encoding methods, see for example (Raidl and Julstrom, 2003; Thompson et al., 2007). GAs have also been proposed to other type of trees-shaped problems, for instance in (Fontes and Gonçalves, 2007) the authors address problems including flows and with general nonlinear cost functions. However, as far as the authors are aware of, there has not yet been proposed any methods for finding solutions to the MST a with maximum number of arcs on each path and flow characteristics. Here, a hybrid genetic algorithm to solve such problem is proposed. It should be noticed that in this problem we also have flow decisions to be made. Furthermore, the costs to be minimized include a general nonlinearly flow dependent cost component and fixed cost component. Nonlinear cost functions arise naturally in this type of problems as a consequence of taking into account economic considerations. Set up costs or fixed-charge costs arise, for example, due to the consideration of a new customer or a new route. Economies of scale often exist, and thus an output increase leads to a decrease in the marginal costs. On the other hand, further output increase may lead to an increase in marginal costs, e.g. by implying the need of extra resources. Therefore, discontinuities are observed. These may also arise due to price-discounting.

2 PROBLEM DESCRIPTION

We consider a problem which is an extension of the MST with a Limit on the number of Arcs on each path from the root vertex (MST-LA). For the MST-LA we wish to find a minimum cost tree spanning all vertices in a given network such that any path from the root vertex to any other vertex has no more than a pre-specified number of arcs. In the problem considered here, flows are also considered since all vertices, except for the root vertex, have a nonnegative flow requirement. Therefore, in addition to finding the arcs that are part of the MST-LA, we must also find the flows that are to be routed along these arcs.

Let $G = (W,A)$ denote a directed network with a set W of $n+1$ vertices (the source vertex and n demand vertices) and a set A of m directed arcs. Vertices 1 to n have associated a nonnegative integer demand r_i, which must be satisfied. The total cost to be minimized is given by the summation of all costs incurred by both using an arc (a setup cost) and routing flow through it (a flow cost). The cost of sending r units of flow through an arc, say (x,z) is given by any function $g_{xz}(r)$ satisfying $g_{xz}(0) = 0$. The flow that can be routed through each arc (x,z) may have upper u_{xz} and lower l_{xz} limits. Let H be the pre-defined maximum number of arcs in any path connecting the source vertex to any other vertex. Several formulations have been provided by Gouveia and his co-authors, see (Gouveia and Requejo, 2001; Gouveia et al., 2008) and the references therein.

3 METHODOLOGY

Our approach comprises a GA to evolve the population of solutions, a tree constructor to generate trees satisfying the vertex demands, a penalty cost function to change the trees so that the maximum number of arcs in any path H may be satisfied, and a local search procedure to refine the feasible trees, see Figure 1.

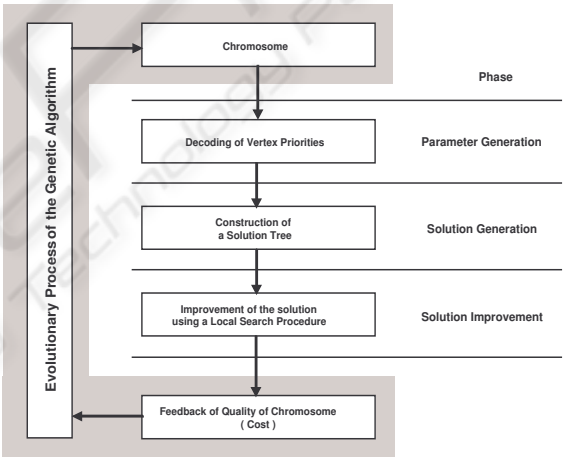

Figure 1: The solution approach.

Usually GAs do not perform well in fine tuning near local optimum solutions, since they tend to get trapped into a local optimum and also to have most of its population concentrated on a small part of the search space located around the local optimum, which is usually termed premature convergence. This problem can be obviated by using structured populations or by hybridization. In the methodology proposed here both are used. For the former case we use a multi-population strategy that evolves populations independently. On each population the dispersion of genetic material is slowed down, giving the algorithm more time to settle on the most promising part of the search-space. Interaction between the populations is provided by interchanging information on good chromosomes. In addition, we use a local search proce-

dure which tries to improve the current solutions by means of small perturbations.

3.1 The Genetic Algorithm

GAs often perform well approximating solutions to all types of problems because they, ideally, do not make any assumption about the underlying fitness landscape; this generality is shown by it successes in diverse fields and problems.

To build such an algorithm many choices have to be made.

3.1.1 Representation Scheme

The GA described in this paper proposes a random-key alphabet, which is comprised of real-valued random numbers between 0 and 1, for encoding trees. Random keys have been used successfully for addressing problems where the relative order of tasks is important e.g., (Gonçalves, 2007; Gonçalves and Almeida, 2002; Gonçalves et al., 2005; Gonçalves and Resende, 2004). The evolutionary strategy used is similar to that of (Bean, 1994), the main difference occurring in the crossover process which is equal to the one proposed in (Gonçalves and Almeida, 2002).

3.1.2 Encoding and Decoding

A chromosome, which is made of n genes, is represented as a vector of random numbers, where n is the number of vertices. These numbers impose an order to the vertices, which we use as priorities to generate trees. Since in this encoding scheme the feasibility issue is moved into the objective function, if a random key vector can be interpreted as a feasible solution then any crossover vector is also feasible.

The values of the random key vector are used to obtain the vertex priorities with which the tree is constructed. The decoding is accomplished by sorting the genes is ascending order, see Figure 2.

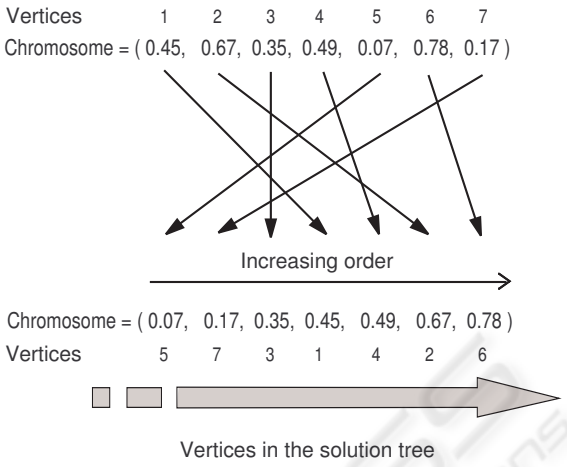

Figure 2: Decoding procedure.

3.2 Evolutionary Strategy

Given the current population and the fitness value of each chromosome, i.e. total cost incurred in constructing the tree and routing the flow, the population is divided into two sets: the elite solutions subset and the remaining solutions. The elite subset is a small subset containing only the very good solutions.

The population in the next generation is formed by all solutions in the elite set, a small number of new randomly generated solutions (mutant), and solutions obtained by mating solutions of the current population (offspring), see Figure 3.

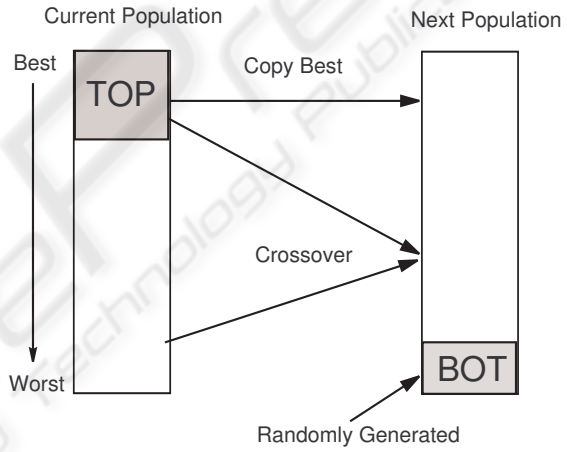

Figure 3: Evolutionary strategy.

By copying the best solutions, we guarantee that the best solution is monotonically improving. However, it may lead to excessive convergence to a local optimum. To overcome this problem we use two strategies. On one hand, we use mutation as described below, and on the other hand we use several populations, which are evolved separately.

3.2.1 Selection and Crossover

Two individuals are randomly chosen to act as parents. One of them is chosen amongst the elite solution set, while the other is chosen from the reminder. Genes are chosen by using a biased uniform crossover, that is, for each gene a biased coin is tossed to decide on which parent the gene is taken from. This way, the child inherits the genes from the elite parent with higher probability.

3.2.2 Mutation

As the mutation operator we use the so-called immigration operator. Immigration acts like a mutation operator, however, instead of performing gene-by-gene mutation with very small probability, at each generation some new individuals are introduced into the next generation. This new individuals are randomly generated from the same distribution as the original population and thus, no genetic material of the current population is brought in.

3.3 Tree Constructor

We use the vertex priorities, given by the decoding procedure, to determine the order by which vertices are considered by the tree constructor. The algorithm repeatedly performs the following three steps in turn, until either a tree or an infeasibility has been obtained.

1. Find the highest priority vertex not yet supplied;
2. Search for the set of vertices that can act as a parent, for the vertex found in 1.
3. Chose as the parent, the highest priority vertex not creating a cycle, if one exists.

As said before, the arc limit H is handled implicitly and not considered within the constructor procedure. When computing the solution fitness, we also consider a penalty function, which is dependent on the constraints violation degree, that is we penalize vertices having more than H arcs in the path from the source vertex, as follows.

$$\sum_{for\ all\ vertices} Max\{0, patharcs_i - H\} \times M, \quad (1)$$

where M is a very large number.

3.4 The Local Search

The local search improves on a given solution by comparing it with adjacent extreme solutions. As no transshipment vertices exist, adjacent extreme solutions are obtained by replacing an arc currently in the solution by an arc not in the solution such that the new solution is still an extreme flow, i.e., a tree.

The order by which candidate arcs are considered is determined by the priorities associated with the outcoming vertex of each candidate arc.

3.5 The Multi Population Strategy

As mentioned before several populations are evolved, each being randomly generated. The populations are left to evolve independently and after a predetermined number of generations they interact by exchanging the best solutions. When evaluating possible interchange strategies we have noticed that exchanging too much information, i.e. too many chromosomes leads to the disruption of the evolutionary process. Also, if the populations exchange information very frequently they do not have enough time to produce good results since their evolutionary process is disrupted before good solutions can be achieved. Therefore, we chose a strategy that after a pre-determined number of generations (determined empirically) inserts only the best two chromosomes in all populations.

4 COMPUTATIONAL EXPERIMENTS

The efficiency and effectiveness of the multi population GA was tested on network flow problems involving nonlinear cost functions made of two components: a set-up cost and a routing cost. The problems data can be downloaded from the OR-Library and a detailed description is provided in (Fontes et al., 2003).

Three different cost function types are considered: types G1 and G2 are variations of the fixed-charge cost function where discontinuities other than at the origin are introduced; and type G3, where arc costs are initially concave and then convex, having a discontinuity at the break point.

$$g_{ij}(r) = \begin{cases} 0, & \text{if } r = 0, \\ -a_{ij}r^2 + b_{ij}r + c_{ij} & \text{if } r \leq \bar{R}, \\ a_{ij}r^2 + b_{ij}r + c_{ij} + k & \text{otherwise,} \end{cases}$$

where $a_{ij} = 0$ for G1 and G2, $k = b_{ij}$ for G1, $k = -b_{ij}$ for G2, and $k = 0$ for G3.

In tables 1 to 3 we summarize the results obtained for uncapacitated problems involving cost functions of types G1, G2, and G3, respectively, with the discontinuity point occurring at 50% of the root vertex outflow. Four different arc limit values have been considered $H = 3, 5, 7, 10$. For each size, cost function type and H value we have solve 30 problem instances. Thus, overall we have solved 150 problem instances for each H value and cost function type.

We report the average computational time, in seconds, required by the Multi Population GA (MPGA) and also on the average deviation from the optimal value. These values have been computed using the optimal solutions obtained by the dynamic programming methodology proposed in (Fontes, 2009).

As it can be seen, from the results reported in tables 1 to 3, the MPGA finds an optimal solution for most problems, except when H is very small. Furthermore, for $H = 3$ there are 19 problems for which

Table 1: Solution quality (% deviation from optimal) and time (s) for cost function type G1.

	Number of vertices				
H	10	12	15	17	19
3	0.00	0.13	0.05	0.082	0.782
5	0.00	0.02	0.00	0.000	0.193
7	0.00	0.00	0.01	0.000	0.011
10	0.00	0.00	0.00	0.000	0.000
Time	12.60	22.50	33.40	50.10	63.00
DP time	0.03	0.36	9.91	113.13	1972.33

Table 2: Solution quality (% deviation from optimal) and time (s) for cost function type G2.

	Number of vertices				
H	10	12	15	17	19
3	0.14	0.13	0.05	0.08	0.78
5	0.00	0.02	0.00	0.00	0.20
7	0.00	0.00	0.01	0.00	0.01
10	0.00	0.00	0.00	0.00	0.00
Time	12.70	20.40	34.20	50.90	64.10
DP time	0.04	0.36	9.67	108.89	1864.30

Table 3: Solution quality (% deviation from optimal) and time (s) for cost function type G3.

	Number of vertices				
H	10	12	15	17	19
3	0.00	0.16	0.03	0.16	0.53
5	0.00	0.01	0.00	0.00	0.21
7	0.00	0.00	0.00	0.00	0.01
10	0.00	0.00	0.00	0.00	0.00
Time	15.10	25.30	44.60	64.50	85.00
DP time	0.04	0.34	9.74	107.89	2053.50

no feasible solution exists and the constraints are effective for almost all problems. For $H = 10$ the constraints are no longer effective.

In order to better understand the results obtained and the conclusions drawn we provide some graphical representations see figures 4 and 5. From the computational times reported it can be seen that the DP algorithm time requirements grow very rapidly, however, this trend is not observed in the proposed algorithm, see Figure 5.

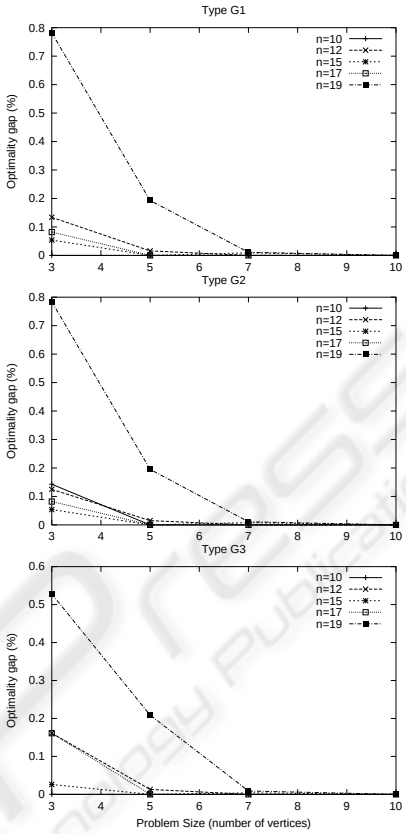

Figure 4: The effect of the value of H on solution quality, for type G1, G2, and G3, respectively.

5 CONCLUSIONS

We have presented a hybrid Multi Population GA (MPGA) that finds nearly optimal, actually optimal for most problems, Hop-Constrained Trees in Nonlinear Cost Flow Networks. Recall that a DP method was used to obtain an optimal solution for these problems. The proposed algorithm combines a local search algorithm with a MPGA, where several populations are evolved independently.

The results obtained have been compared with existing literature and the comparisons have shown the proposed algorithm to improve upon the efficiency of existing methods, since the computational time requirements are modest for all problem sizes (always bellow two minutes). We have solved 150 problem instances with four different H values and three cost function types. When the Hop-constraints are not very tight we were able to find an optimal for almost all problems. Nevertheless, when they are very tight we are still able to find very good solutions.

Thus, the Hybrid MPGA proposed here is capable of efficiently finding heuristic solutions, close to optimal, for the Minimum Nonlinear Cost Spanning Tree Problem with a limit on the maximum number of arcs in each path from the root vertex and routing flows, which is NP-hard.

ACKNOWLEDGEMENTS

This work has been supported by funds granted by project PTDC/GES/72244/2006.

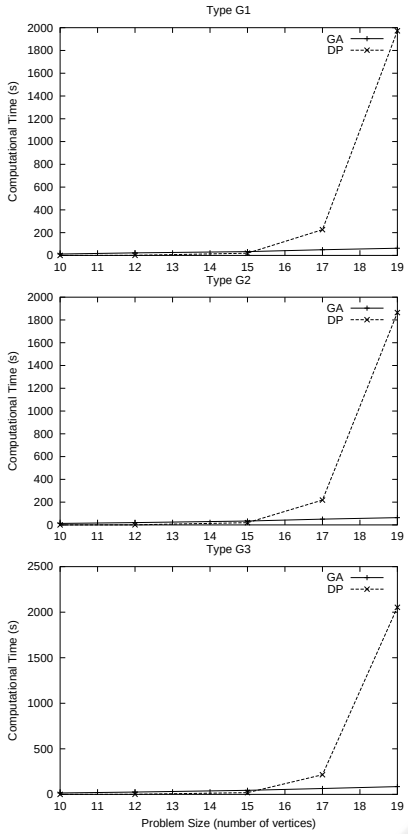

Figure 5: The effect of the value of H on computational time, for type G1, G2, and G3, respectively.

REFERENCES

Ahuja, R. and Orlin, J. (2001). Multi-exchange neighborhood structures for the capacitated minimum spanning tree problem. *Mathematical Programming*, 91:71–97.

Bean, J. (1994). Genetics and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6:154–160.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to algorithms*. MIT press Cambridge, MA, 2nd edition.

Dahl, G., Gouveia, L., and Requejo, C. (2006). On formulations and methods for the hop-constrained minimum spanning tree problem. In Pardalos, P. M. and Resende, M., editors, *Handbooks of Telecommunications*, pages 493–515. Springer.

Deering, S. E., D., D. E., and Farinacci (1994). An architecture for wide-area multicast routing. *Proceedings of SIGCOMM*.

Fontes, D. B. M. M. (2009). Optimal hop-constrained trees for nonlinear cost flow networks. *INFOR, to appear*.

Fontes, D. B. M. M. and Gonçalves, J. F. (2007). Heuristic solutions for general concave minimum cost network flow problems. *Networks*, 50:67–76.

Fontes, D. B. M. M., Hadjiconstantinou, E., and Christofides, N. (2003). Upper bounds for single source uncapacitated minimum concave-cost network flow problems. *Networks*, 41:221–228.

Gen, M., Cheng, R., and Oren, S. (2001). Network design techniques using adapted genetic algorithms. *Advances in Engineering Software*, 32:731–744.

Gen, M., Kumar, A., and Kim, R. (2005). Recent network design techniques using evolutionary algorithms. *International Journal of Production and Economics*, 98:251–261.

Gonçalves, J. (2007). A hybrid genetic algorithm-heuristic for a two-dimensional orthogonal packing problem. *European Journal of Operational Research*, 183:1212–1229.

Gonçalves, J. and Almeida, J. (2002). A hybrid genetic algorithm for assembly line balancing. *Journal of Heuristics*, 8:629–642.

Gonçalves, J., Mendes, J., and Resende, M. (2005). A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research*, 167:77–95.

Gonçalves, J. and Resende, M. (2004). An evolutionary algorithm for manufacturing cell formation. *Computers and Industrial Engineering*, 47:247–273.

Gouveia, L., Paias, A., and Sharma, D. (2008). Modeling and solving the rooted distance-constrained minimum spanning tree problem. *Computers & Operations Research*, 35:600–613.

Gouveia, L. and Requejo, C. (2001). A new lagrangean relaxation approach for the hop-constrained minimum spanning tree problem. *European Journal of Operational Research*, 132:539–552.

Han, L., Wang, Y., and Guo, F. (MAY 2005). A new genetic algorithm for the degree-constrained minimum spanning tree problem. *IEEE International Workshop on VlSI Design and Video Technology*, pages 125–128.

Lacerda, E. and Medeiros, M. (2006). A genetic algorithm for the capacitated minimum spanning tree problem. *IEEE Congress on Evolutionary Computation*, 1-6:725–729.

LeBlanc, L. and Reddoch, R. (1990). Reliable link topology/capacity design and routing in backbone telecommunication networks. *First ORSA telecommunications SIG conference*.

Raidl, G. and Julstrom, B. (2003). Edge sets: An effective evolutionary coding of spanning trees. *IEEE Transactions on Evolutionary Computation*, 7:225–239.

Thompson, E., Paulden, T., and Smith, D. (2007). The dandelion code: A new coding of spanning trees for genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 11:91–100.

Woolston, K. and Albin, S. (1988). Design of centralized networks with reliability and availability constraints. *Computers & Operations Research*, 15:207–217.

Zeng, Y. and Wang, Y. (2003). A new genetic algorithm with local search method for degree-constrained minimum spanning tree problems. *Proceedings of ICCIMA*, pages 218–222.