A Centralized Network Design Problem with Genetic Algorithm Approach

Gengui Zhou, Zhenyu Cao, Jian Cao, and Zhiqing Meng

College of Business and Administration
Zhejiang University of Technology, Zhejiang 310014, China
`ggzhou@zjut.edu.cn`

Abstract. A centralized network is a network where all communication is to and from a single site. In the combinatorial optimization literature, this problem is formulated as the capacitated minimum spanning tree problem (CMST). Up to now there are still no effective algorithms to solve this problem. In this paper, we present a completely new approach by using the genetic algorithms (GAs). For the adaptation to the evolutionary process, we developed a tree-based genetic representation to code the candidate solution of the CMST problem. Numerical analysis shows the effectiveness of the proposed GA approach on the CMST problem.

1 Introduction

A centralized network is a network where all communication is to and from a single site (Kershenbaum, 1993). In such networks, terminals are connected directly to the central site. Sometimes multipoint lines are used, where groups of terminals share a tree to the center and each multipoint line is linked to the central site by one link only. This means that optimal topology for this problem corresponds to a tree in a graph $G = (V, E)$ with all but one of nodes in V corresponding to the terminals. The remaining node refers to the central site, and edges in E correspond to the feasible telecommunication wiring. Each subtree rooted in the central site corresponds to a multipoint line. Usually, the central site can handle, at most, a given fixed amount of information in communication. This, in turn, corresponds to restricting the maximum amount of information flowing in any link adjacent to the central site (which we will refer as the root of the graph G) to that fixed amount. In the combinatorial optimization literature, this problem is known as the capacitated minimum spanning tree problem (CMST).

The CMST problem has been shown to be *NP*-hard by Papadimitriou (Papadimitriou, 1978). Much of the early works focused on heuristic approaches to find good feasible solutions. Among them are those by Chandy and Lo (Chandy, 1973), Kershenbaum (Kershenbaum, 1974), and Elias and Ferguson (Elias and Ferguson, 1974). The only full optimization algorithms that we are aware of are by Gavish (Gavish, 1982) and Kershenbaum et al. (Kershenbaum at al., 1983), but their use is limited to problems involving up to 20 nodes. Gavish (Gavish, 1983) also studied a new formulation and its several relaxation procedures for

Y. Wang, Y. Cheung, and H. Liu (Eds.): CIS 2006, LNAI 4456, pp. 123–132, 2007.

the capacitated minimum directed tree problem. Recently, this problem has even more aroused many researchers' interesting by using cutting plane algorithms by Gouveia (Gouveia, 1995) and Hall (Hall, 1996), branch-bound algorithm by Malik and Yu (Malik and Yu, 1993), neighborhood search technique by Ahuja et al. Ahuja, 2003, and ant colony optimization technique by Reimann and Laumanns Reimann, 2006.

In the studies that date back twenty years, it is not surprising to find that only very small instances were attempted in solving this problem. In this paper, we present a completely new approach by using the genetic algorithms (GAs), which have been demonstrating their powerful potential in dealing with such complicated combinatorial problem with tree topology (Zhou and Gen, 1998, 2003). For the adaptation to the evolutionary process, we developed a tree-based genetic representation to code the candidate solution of the CMST problem. Because the new genetic representation has the tree topology and is only encoded by a bigeminal string, it is easy to go on genetic operations. Also the tree-based genetic representation guarantees that the candidate solutions are always feasible solutions of the problem to be solved, and its locality property makes the evolutionary process more efficiency. Numerical analysis shows the effectiveness of the proposed GA approach on this CMST problem.

2 Problem Formulation

Firstly, we formulate the centralized network design problem as a zero-one integer program. This particular formulation was first expressed by Gavish (Gavish, 1982). Considering a complete, undirected graph $G = (V, E)$, we let $V = 1, 2, ..., n$ be the set of nodes representing the terminals and denote the central site, or "root" node, as node 1, and $E = \{(i, j)|i, j \in V\}$ be the set of edges representing all possible telecommunication wiring. For a subset of nodes $S \subseteq V$ we define $E(S) = \{(i, j)|i, j \in S\}$ to be the edges whose endpoints are both in S. We define the following binary decision variables for all edges $(i, j) \in E$:

$$x_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ is selected;} \\ 0 & \text{otherwise.} \end{cases}$$

Let c_{ij} be the (fixed) cost of including edge (i, j) in the solution, and suppose that d_i represents the demand at each node $i \in V$, where by convention the demand of the root node $d_1 = 0$. We also use $d(S)$, $S \subseteq V$, to denote the sum of the demands of the nodes of S. The subtree capacity is denoted κ .

It is not hard to verify that the following formulation is a valid integer programming representation for the centralized network design problem:

$$\min \quad z = \sum_{i=1}^{n-1} \sum_{j=2}^{n} c_{ij} x_{ij} \tag{1}$$

$$s.t. \quad \sum_{i=1}^{n-1} \sum_{j=2}^{n} x_{ij} = n - 1 \tag{2}$$

$$\sum_{i \in S} \sum_{\substack{j \in S \\ j > 1}} x_{ij} \leq |S| - \lambda(S), S \subseteq V \backslash \{1\}, |S| \geq 2 \tag{3}$$

$$\sum_{i \in U} \sum_{\substack{j \in U \\ j > 1}} x_{ij} \leq |U| - 1, U \subset V, |U| \geq 2, \{1\} \in U \tag{4}$$

$$x_{ij} = 0 \, or \, 1, \, (i = 1, 2, \ldots, n-1, \, j = 2, 3, \ldots, n.) \tag{5}$$

Equality (2) is true of all spanning trees: a tree with n nodes must have $n-1$ edges. Inequalities (4) are some of the standard rank inequalities for spanning trees: if more than $|U|-1$ edges connect the nodes of a subset U, then that set of edges must contain a cycle. The parameter $\lambda(S)$ in (3) refers to the bin-packing number of the set S, namely, the number of bins of size needed to pack the nodes of items of size d_i for all $i \in S$. These constraints are similar to (4), except that they reflect the capacity constraint: if the set S does not contain the root node, then the nodes of $\lambda(S)$ must be contained in at least (S) different subtrees off of the root.

In the case that the demands of all non-root nodes are 1, inequalities (3) can be expressed more simply as follows as items of unit size can always be packed in $\lceil |S|/\kappa \rceil$ bins or subtrees.

$$\sum_{i \in S} \sum_{\substack{j \in S \\ j > 1}} x_{ij} \leq |S| - \left\lceil \frac{|S|}{\kappa} \right\rceil, S \subseteq V \backslash \{1\}, |S| \geq 2 \tag{6}$$

The above mathematical formulation is regarded as the capacitated minimum spanning tree problem in literature. Assuming that all the constraints in (3) or (6) can be explicitly represented, it is possible to compute a lower bound on the problem by replacing the binary variables with continuous variables in the range 0 to 1 and solving the resulting linear program. Unfortunately, there are $O(2^n)$ constraints in (3) or (6), leading to a very large linear program even for moderate values of n. In fact, the problem is *NP*-hard (Papadimitriou, 1978) and algorithms exist yielding exact solutions only for problems of modest size (Gavish, 1985).

Up to now, all heuristic algorithms for this problem are only focused on how to deal with the constraints to make the problem simpler to solve. On the approach of cutting plane algorithms (Gouveia, 1995, Hall, 1996) or branch-bound algorithm (Malik, 1993), the network topology of the CMST problem are usually neglected. As a result, it results in the exponential explosion of constraints. In the following section, we focus on a new approach on this problem by using genetic algorithms. In the evolutionary process, we make full use of its tree topology of the CMST problem and develop the algorithm to get the optimal or near-optimal solutions.

3 Genetic Algorithms Approach

A genetic algorithm (GA) can be understood as an "intelligent" probabilistic search algorithm which can be applied to a variety of combinatorial optimization problems (Gen and Cheng, 2000). The theoretical foundations of GAs were originally developed by Holland (Holland, 1975). The idea of GAs is based on the evolutionary process of biological organisms in nature. During the course of the evolution, natural populations evolve according to the principles of natural selection and "survival of the fittest". Individuals which are more successful in adapting to their environment will have a better chance of surviving and reproducing, whilst individuals which are less fit will be eliminated. This means that the genes from the highly fit individuals will spread to an increasing number of individuals in each successive generation. The combination of good characteristics from highly adapted ancestors may produce even more fit offspring. In this way, species evolve to become more and more well adapted to their environment.

3.1 Genetic Representation

For the CMST problem, two main factors should be taken into consideration if we want to keep its tree topology structure in the genetic representation: one is the connectivity among nodes; the other is the degree value (the number of edges connected on it) of each node. Therefore, the intuitive idea of encoding a tree solution is to use a two-dimension structure for its genetic representation. One dimension encodes the nodes of a spanning tree; another dimension encodes the degree value of each node. Thus it needs a $2 \times n$ matrix to represent a chromosome for an n-node tree. Obviously the genes in node dimension take the integers from 1 to n exclusively; the genes in degree dimension take the integers from 1 to b inclusively (b is the largest degree value for all nodes). We define this genetic representation as tree-based permutation.

For a rooted tree like the CMST solution, we can take one node (i.e. node 1) as the root node of it. All other nodes are regarded being connected to it hierarchically. For any node (current node), the node incident to it on the upper hierarchy is called as its predecessor node and the node incident to it on the lower hierarchy is called as its successor node. Obviously, the root node has no predecessor node and the leaf node has no successor node. Based on this observation, the tree-based permutation of such a tree can be encoded as the following procedure:

procedure: tree-based permutation encoding

step 1: Select node 1 (root node) as the current node in a labeled tree T, put it as the first digit in the node dimension of the permutation and its degree value as the first digit in the degree dimension.

step 2: Check all successor nodes of the current node from left branch to right branch. If there are successor nodes, let the leftmost successor node as the current node, then go to **step 3**. Otherwise, go to **step 4**.

step 3: Put the label digit of the current node to the permutation in the node dimension and its degree value to the permutation in the degree dimension (here we build the permutation by appending digits to the right), then go to **step 2**.

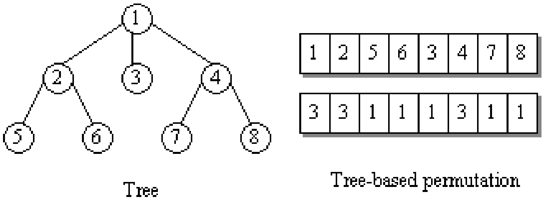

Fig. 1. A rooted tree and its tree-based permutation

```
step 4: Delete the current node and its adjacent edge from the tree, let its
    predecessor node as the current node.
step 5: If all nodes have been checked, stop; otherwise, go to step 2.
```

Figure 1 illustrates an example of this tree-based permutation. For the initial population, each chromosome can be generated randomly. However, in order to keep the connectivity between nodes, the genes in the degree dimension need to satisfy the following conditions: For an n-node tree, the total degree value for all nodes is $2(n-1)$. Suppose that d_{rest} is the total degree value of the nodes whose degree value in degree dimension have been assigned and d_{rest} is the total lower bound of the degree values for all those nodes whose degree value in degree dimension have not been assigned. Then the degree value of the current node in degree dimension should hold: no less than 1. The degree value of the current node together with that of the rest nodes should hold: no less than d_{rest} and no greater than $2(n-1) - d_{used}$. Especially, for the root node, its degree value should take no less the value of $[|V|/\kappa]$, which reflects the number of subtrees connected to the root node to satisfy the capacity constraint.

Also, it is easy to decode the above tree-based permutation into a tree. Suppose that the node dimension for individual P is represented as $P_1(k)$, $k = 1, 2, ..., n$ and the degree dimension for individual P as $P_2(k)$, $k = 1, 2, ..., n$. The decoding procedure for each individual in the form of tree-based permutation can be operated as follows (for the convenience of the procedure operation, the first gene value in the degree dimension should be added by one):

```
procedure: tree-based permutation decoding
step 1: Set $k \leftarrow 1$ and $j \leftarrow 2$.
step 2: Select the node $r = P_1(k)$ and the node $s = P_1(j)$, add the edge from
    $r$ to s into a tree.
step 3: Let $P_2(k) \leftarrow P_2(k) - 1$, $P_2(j) \leftarrow P_2(j) - 1$.
step 4: If $P_2(k) = 0$, let $k \leftarrow k - 1$, otherwise, go to step 6.
step 5: If $j = n$, stop, otherwise, go to step 4.
step 6: If $P_2(j) \geq 1$, let $k \leftarrow j$, $j \leftarrow j + 1$, go to step 2, otherwise, $j \leftarrow j + 1$,
    go to step 2.
```

Obviously, any rooted spanning tree can be encoded by this representation scheme and any permutation encoded in this way represents a rooted spanning tree. However, the relation between the encoding and its spanning tree may not be one-to-one mapping because different chromosomes may represent the same spanning tree. But it is possible to represent all possible spanning trees on a complete graph. It is also easy to go back and forth between the encoded representation of a tree and the tree's representation for evaluating the fitness, which will be illustrated in Section 3.4.

It is important to point out that this encoding keeps the structure of a tree, so it possesses the locality in the sense that small changes in the representation (such as mutation operation) make small changes in the tree. Without this property, the GA search tends to drift rather than converge to a highly fit population. Therefore, this encoding is well adapted to the evolutionary process and thus adopted as the genetic representation for the CMST problem.

3.2 Genetic Operation

Genetic operation is used to alter the genetic composition of individuals or chromosomes. Usually it contains two kinds of operations: crossover and mutation. In order to keep all individuals being feasible after genetic operations on the tree-based permutation for the CMST problem, only three kinds of mutations are adopted in this paper.

Exchange mutation on nodes: Exchange mutation selects two genes (nodes) at random and then swaps the genes (nodes). This mutation is essentially a 2-opt exchange heuristic. The operation can be illustrated by Figure 2.

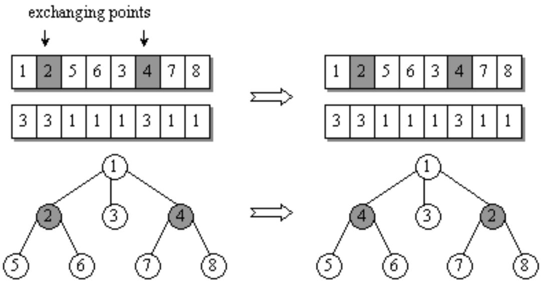

Fig. 2. Exchange mutation on nodes

Inversion mutation on nodes: Inversion mutation selects two genes (nodes) at random and then inverts the substring between these two genes (nodes). It is illustrated in Figure 3.

Insertion mutation: Insertion mutation selects a string of genes (branch) at random and inserts it in a random gene (node). When a string of genes are taken off from a gene, the gene value of that node should be decreased by one. When a string of genes are added on a gene, the gene value of that node should be increased by one. The operation can be illustrated by Figure 4. Obviously, this operation is indispensable for the evolutionary process to evolve to the fit tree structures.

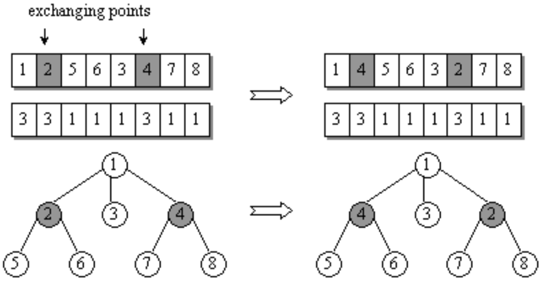

Fig. 3. Inversion mutation on nodes

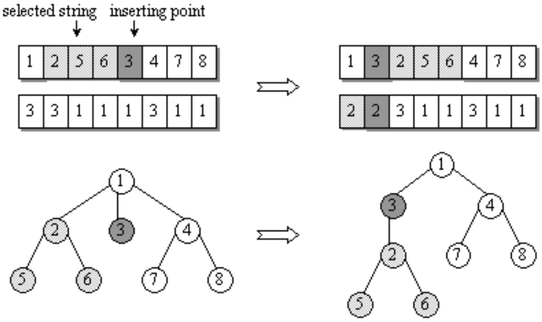

Fig. 4. Inversion mutation on nodes

3.3 Modification

For the CMST problem, there is the capacity constraint for each spanning tree. Especially, when the demands of all terminals are equal to one, the problem is finding a rooted spanning tree in which each of the subtree off of the root node contains at most κ nodes. Therefore, before evaluation, if there are such individuals whose subtrees violate the capacity constraint, we use the insertion mutation operation to insert the extra branch on a subtree into other subtree with less nodes.

3.4 Evaluation and Selection

Evaluation is to associate each individual with a fitness value which reflects how good it is. The higher fitness value of an individual, the higher its chances of survival and reproduction and the larger its representation in the subsequent generation. Obviously the evaluation together with selection provides the mechanism of evolving all individuals toward the optimal or near-optimal solutions. Simply, we take the objective value of Equation (1) for each individual's fitness value after its decoding from genotypic representation to phenotypic representation.

As to selection, we adopt the $(\mu + \lambda)$-selection strategy(Back, 1991). But in order to avoid the premature convergence of the evolutionary process, our selection strategy only selects μ different best individuals from μ parents and λ offspring. If there are no μ different individuals available, the vacant pool of population is filled with renewal individuals.

3.5 GA Procedure for the CMST

To summarize our GA approach on the CMST problem, the overall procedure can be outlined as follows:

```
procedure: GA for CMST
   begin
      $t \leftarrow 0$;
      initialize the population of parents $P(0)$;
      evaluate $P(0)$;
      while (not termination condition ) do
         reproduce $P(t)$ to yield the population of offspring $C(t)$;
         modify $P(t)$;
         evaluate $C(t)$;
         $t \leftarrow t + 1$;
      end
   end
```

Table 1. The cost matrix for the numerical example ($n = 16$, $\kappa = 5$)

i/j	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1616	1909	246	622	829	1006	2237	399	1717	632	1191	2116	824	1336	1519
2		2996	1419	2217	1213	2046	3753	1516	1180	1997	552	3622	2423	1367	862
3			1893	1543	1792	2785	1362	1667	3556	2332	2446	1248	1508	3233	3287
4				799	593	1188	2369	242	1670	857	962	2243	1004	1348	1425
5					1230	1253	1625	761	2301	801	1748	1509	206	1873	2119
6						1758	2597	480	1883	1449	663	2463	1420	1701	1573
7							2703	1399	1470	454	1849	2612	1350	960	1470
8								2238	3922	2304	3231	137	1442	3476	3743
9									1889	1029	1009	2108	959	1586	1628
10										1693	1437	3808	2480	511	340
11											1685	2206	909	1205	1603
12												3098	1952	1429	1100
13													1331	3368	3624
14														2038	2309
15															578

The parameters for the proposed GA approach are set as follows: population size pop_size = 200; mutation probabilities for three mutation operations are $p_m = 0.3$ respectively; maximum generation max_gen = 500; and run by 20 times.

4 Computational Experience

In order to illustrate the ideas that were presented in the previous section, we present a numerical example given out by Gavish (Gavish, 1985). The example consists of a CMST problem with 16 nodes, a unit traffic between each node and node 1, and a capacity restriction . The cost matrix for the example is presented in Table 1.

Gavish adopted an augmented lagrangean based algorithms to solve this problem and got the optimal solution 8526 (Gavish, 1985). By the proposed GA, we also got the optimal solution 8526 and its corresponding topology of a tree. Figure 5 illustrates the result.

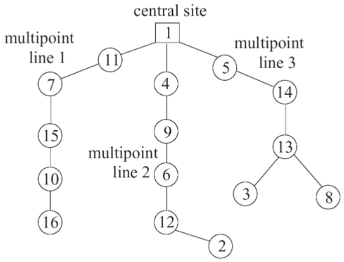

Fig. 5. Inversion mutation on nodes

5 Conclusion and Further Work

The centralized network design problem can be formulated as a capacitated minimum spanning tree problem. In this paper we developed a new approach to deal with this problem by using genetic algorithms. In order to code the corresponding rooted tree topology for the genetic representation on the CMST problem, we presented a tree-based permutation which is able to represent all possible rooted trees. Small numerical example shows the effectiveness of the proposed GA approach on the CMST problem.

Further works are needed to demonstrate the effectiveness of the proposed GA approach on this problem, which including the test on the problems with larger scale, the comparison with its lower bound since it is difficult to give out the optimal solution of the problem on larger scale. However, the research work gives out an novel approach on such complicated combinatorial optimization problems.

Acknowledgements

This research work was partially supported by grant No.70671095 from National Nature Science Foundation of China.

References

1. Ahuja, R.K., Orlin, J.B., Sharma, D.: A positive very large-scale neighborhood structure for the capacitated minimum spanning tree problem. Operations Research letters 31, 185–194 (2003)
2. Bäck, T., Hoffmeister, F., Schwefel, H.: A survey of evolution strategy. In: Belew, R., Booker, L. (eds.) Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 2–9. Morgan Kaufmann Publishers, San Mateo,CA (1991)
3. Chandy, K.M., Lo, T.: The capacitated minimum spanning tree. Networks 3, 173–182 (1973)
4. Elias, D., Ferguson, M.J.: Topological design of multipoint teleprocessing networks. IEEE Trans. Commun. 22, 1753–1762 (1974)
5. Garey, M., Johnson, D. (eds.): Computers and Intractability: A Guide to the Theory of NP-completeness. W.H. Freeman and Co, San Francisco (1979)
6. Gavish, B.: Topological design of centralized computer networks–formulation and algorithms. Networks 12, 355–377 (1982)
7. Gavish, B.: Formulation and algorithms for the capacitated minimal directed tree problem. J. Assoc. Comput. Machinery 30, 118–132 (1983)
8. Gavish, B.: Augmented lagrangean based algorithms for centralized network design. IEEE transaction on Commun. 33, 1247–1257 (1985)
9. Gen, M., Cheng, R.: Genetic Algorithms and Engineering Optimization. John Wiley & Sons, New York (2000)
10. Gouveia, L.: A 2^n constraint formulation for the capacitated minimal spanning tree problem. Operations Research 43, 130–141 (1995)
11. Hall, L.: Experience with a cutting plane algorithm for the capacitated spanning tree problem. INFORMS Journal on Computing 8, 219–234 (1996)
12. Holland, J.H.: Adaptation in natural and Artificial Systems. MIT Press, Cambridge, MA (1975)
13. Kershenbaum, A.: Computing capacitated minimal spanning trees efficiently. Networks 4, 299–310 (1974)
14. Kershenbaum, A., Boorstyn, R.R., Oppenheim, R.: Centralized teleprocessing network design. Networks 13, 279–293 (1983)
15. Kershenbaum, A.: Telecommunication Network Design Algorithms. McGraw-Hill, Inc, Singapore (1993)
16. Malik, K., Yu, G.: A branch and bound algorithm for the capacitated minimum spanning tree problem. Networks 23, 525–532 (1993)
17. Papadimitriou, C.H.: The complexity of the capacitated tree problem. Networks 8, 217–230 (1978)
18. Reimann, M., Laumanns, M.: Savings based ant colony optimization for the capacitated minimum spanning tree problem. Computers & Operations Research 33, 1794–1822 (2006)
19. Zhou, G., Gen, M.: An effective genetic algorithm approach to the quadratic minimum spanning tree problem. Computer & Operations Research 25, 229–237 (1998)
20. Zhou, G., Gen, M.: A genetic algorithm approach on tree-like telecommunication network design problem. Journal of The Operational Research Society 54, 248–254 (2003)