A Bi-Level Hierarchical GA for Reliable Network Topology Design

Jong Ryul Kim, Mitsuo Gen, and Mitsuo Yamashiro
Dept. of Indust. & Inform. Systems Engg., Graduate School of Engg.,
Ashikaga Institute of Tech. Ashikaga 326-8558, Japan
Email: {jrkim gen yamasiro}@ashitech.ac.jp

Abstract

In this paper, we consider a bi-level hierarchical genetic algorithm for designing the reliable network topology with a structure that is comprised of the spanning tree of the centers and users using fiber-optic cables. Generally, anticipating future requirements of network, centers are set up by government, public enterprise, or some network providers. Then, users in each house connect these centers with some criteria such as connection cost, average message delay, and so on. Here, we can see the special structure that has two kinds of decision maker and has two hierarchical levels of decision makers. Therefore, network topology design problems can be represented by hierarchical optimization or bi-level programming problems. Thus, we develop a bi-level hierarchical genetic algorithm for solving the network design problems with network reliability constraints, which can be represented by hierarchical optimization problems. Finally, we get some experiments in order to investigate the efficiency of the proposed method, analyzing the results obtained by the proposed genetic algorithm (GA).

KEYWORDS: Network Topology Design, Network Reliability, Bi-level Hierarchical Genetic Algorithm

1 Introduction

Topology design problems for broad-band communication networks have been taken attentions by many related researchers, according to the scale of communication networks. The use of communication networks has also been rapidly increasing in order to exchange information with each other, share resources, get high reliability by having alternative sources of supply, save money, and provide a powerful communication medium among widely separated peoples [Tanenbaum 1989]. Therefore, in many real world application, it becomes important problems how we effectively design a network that certain constraints are met and objectives are optimized. Lastly, these communication network systems are well designed with fiber optic cable, because the requirements from users become increased, due to its potentially limitless capabilities: huge bandwidth (nearly 50 Tbps), low signal attenuation (as low as 0.2 dB/km), low signal distortion, and low power requirement, low material usage, and small space requirement.

In this paper, we propose a bi-level hierarchical genetic algorithm for designing the reliable network topology with a structure that is comprised of the spanning tree of the centers and users using fiber-optic cables. Especially, considering the high cost of the fiber optic cable that is used to connect the between centers and the between center and users, the network architecture is composed of a spanning tree. Also, as designing these computer network systems, an important step is to find the best layout of components to optimize the performance criteria, such as cost, message delay, traffic, reliability, and so on. The performance criteria of these systems are important and are largely determined by network topology. Generally, anticipating future requirements of network, centers are set up by government, public enterprise, or some network providers. Then, users in each house connect these centers with some criteria such as connection cost, average message delay, and so on. Here, we can see the special structure that has two kinds of decision maker, *i.e.*, government, public enterprise, or some network providers, and users, and has two hierarchical levels of decision makers. Therefore, network topology design problems can be represented by hierarchical optimization or bi-level programming problems, which are extensions of Stackelberg games and decentralized planning problems with multiple decision makers in a hierarchical organization [Niwa *et al.* 1998; Sakawa *et al.* 1999]. Thus, we develop a bi-level hierarchical genetic algorithm for solving the network design problems with network reliability constraints, which can be represented by hierarchical optimization problems. Finally, we get some experiments in order to investigate the efficiency of the proposed method, analyzing the results obtained by the proposed GA.

2 Mathematical Models

In order to formulate the problem, we consider a network that connect n service centers and m users. For example, Figure 1 shows network with 5 service centers and 18 users. The communication traffic demands between the users are given by an $m \times m$ matrix $\boldsymbol{U}$ which is called the users traffic matrix. An element u_{ij} of matrix $\boldsymbol{U}$ represents the traffic from user i to user j. We shall assume that the traffic characteristics are known and summarized in the users traffic matrix $\boldsymbol{U}$. Also we define the $n \times n$ service center topology matrix $\boldsymbol{X}$ which represent the connected appearance of service centers. An element x_{ij} represents whether the centers i and j are connected. We further assume that networks are partitioned into n segments (service centers). The users are distributed over those n service centers. The $n \times m$ clustering matrix $\boldsymbol{Y}$ specifies which user belongs to which center. An element y_{ij} means whether user j belongs to center i. We define the $n \times (n+m)$ matrix $\boldsymbol{S}$

called the spanning tree matrix ($[\boldsymbol{X}\ \ \boldsymbol{Y}]$) and define the $n \times n$ matrix $\boldsymbol{T}$ called the service center traffic matrix. An element t_{ij} of this matrix represents the traffic forwarded from users in center i to users in center j. We can calculate the service center traffic matrix t as follows: $\boldsymbol{T} = \boldsymbol{YUY}^T$.

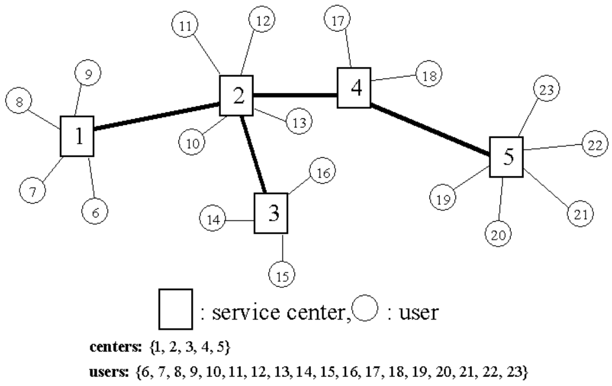


Figure 1: A sample LAN structure

Also the total offered traffic Γ is represented as follows:

$$\Gamma = \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij}.$$

The total traffic at center k, $d_k(\boldsymbol{S})$, is represented as follows:

$$d_k(\boldsymbol{S}) = \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} \cdot a_{ij}^{k}(\boldsymbol{X}), \quad k = 1, \cdots, n.$$

where $a_{ij}^{k}(\boldsymbol{X})$ means whether traffic from center i to center j through center k exists. The total traffic through link (k, l), $f_{kl}(\boldsymbol{X})$, is represented as follows:

$$f_{kl}(\boldsymbol{S}) = \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij} \cdot b_{ij}^{(k,l)}(\boldsymbol{X}),$$
$$k = 1, \cdots, n \quad l = 1, \cdots, n.$$

where $b_{ij}^{(k,l)}(\boldsymbol{X})$ means whether traffic from center i to center j passes through existing link connecting centers k and l exists.

And M/M/1 model [Bertsekas and Gallager 1992] is used in this paper to describe a single cluster (service center) behavior. Then we can formulate the reliable network design problems represented by bi-level hierarchical optimization problems as follows:

$$\min_{\boldsymbol{X}} \quad \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} c_{ij}^{X} \cdot x_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}^{Y} \cdot y_{ij} \tag{1}$$

where $\boldsymbol{Y}$ solves:

$$\min_{\boldsymbol{Y}} \quad \frac{1}{\Gamma}\left[\sum_{i=1}^{n} \frac{d_i(\boldsymbol{S})}{C_i - d_i(\boldsymbol{S})} + \sum_{i=1}^{n}\sum_{j=1}^{n} \beta_{ij} \cdot f_{ij}(\boldsymbol{S})\right] \tag{2}$$

$$\text{s. t.} \quad R(\boldsymbol{S}) > R_{lim} \tag{3}$$

$$\sum_{i=1}^{n} y_{ij} = 1, \ j = 1, 2, \cdots, m \tag{4}$$

$$d_i(\boldsymbol{S}) < C_i, \ i = 1, 2, \cdots, n \tag{5}$$

where $R(\boldsymbol{S})$ is the network reliability, R_{lim} means the requirement of network reliability, C_i is the traffic capacity of center i, β_{ij} is the delay per bit due to the link between centers i and j, c_{ij}^{X} is the cost of the link between centers i and j, and c_{ij}^{Y} is the cost of the link between center i and user j.

The above mathematical model is formulated as bi-level programming problem that is a simple example of multi-level programming problems to solve the decentralized planning problems with multiple decision makers in a hierarchical organization. In these cases, the Stackelberg strategy (solution) has been usually employed as solution concept, based on Stackelberg game theory [Lai 19 96; Niwa *et al.* 1998; Sakawa *et al.* 1999]. A hierarchical organization has the following common features: interactive decision-making units exist within a predominantly hierarchical structure; the execution of decisions is sequential, from higher (leader or upper) to lower (follower) levels; each unit independently maximizes its own benefits, but is affected by the actions of other units through externalities; the external effect on a decision maker's problem can be reflected in both his/her objective function and the set of feasible decisions. In the basic concept of hierarchical optimization techniques, the leader sets his/her goal and/or decisions and then asks followers for their optima calculated in isolation; the follower's decisions are submitted to and then modified by the leader with consideration of the overall benefit for the organization; and the process is continued until a solution is reached [Lai 1996].

3 A Genetic Algorithm

3.1 Representations and Initialization

The genetic representation is a kind of data structure which represents the candidate solutions of the problem in coding space. Usually different problems have different data structures or genetic representations. There are two kinds of decision variables in order to design active network configurations. Therefore, we employ two kinds genotypes,*i.e.*, one is for the connection between service centers, and the others is for the connection between centers and users. In this paper, we employ the encoding method that the connection between service centers is represented Prüfer number and the connection between centers and users is depicted by clustering string which describes distribution of users into the service centers.

One of the classical theorems in graphical enumeration is Calyley's theorem that there are $k^{(k-2)}$ distinct labeled trees on a complete graph with k nodes. Prüfer provided a constructive proof of Cayley's theorem by establishing an one-to-one correspondence between such trees and the set of all strings of $k-2$ digits [Zhou and Gen 1997]. This means that we can use only $k-2$ digits permutation to uniquely represent a tree where each digit is an integer between 1 and k inclusive. This permutation is usually known as the *Prüfer number.* For any tree there are always at least two leaf nodes [Skiena 1990]. Prüer number encoding is not only capable of equally and uniquely representing all possible spanning tree, but also explicitly contains the information of node degree that any node with degree d will appear exactly $d-1$ times in the encoding, *i.e.*, when a node appear d times in Prüer number, the node exactly have $d+1$ connections with other node.

Prüer number is suitable for encoding a spanning tree, especially in some research fields, such as transportation problems, minimum spanning problems, and so on. Also, the verification for the excellence of Prüer number is addressed by the references [Gen and Cheng 1997; Gen and Cheng 1999; Zhou and Gen 1997].

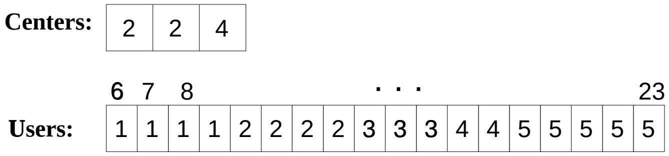


Figure 2: Representation for Figure 1

We randomly generate the chromosome in the initialization process as shown in Figure 2, *i.e.*, service centers are composed of $n-2$ digits (Prüfer number) randomly generated in the range $[1,n]$ and users are make up of m digits (clustering string) randomly generated in the range $[1,n]$, which mean how to allocate the users to service centers so that each user belongs to a specific service center. Note that the our representation have the $n+m-2$ size of chromosome.

3.2 Evaluation

For upper-level, we evaluate as follows:

$$Up\_eval(\boldsymbol{X},\hat{\boldsymbol{Y}}) = \begin{cases} \frac{1}{\text{Equation(1)}}, & \text{feasible case,} \\ BIG, & \text{otherwise.} \end{cases}$$

where $\hat{\boldsymbol{Y}}$ is the best result of lower-level for given $\boldsymbol{X}$ and BIG means a very large number. For lower-level, we evaluate as follows:

$$Lo\_eval(\boldsymbol{X},\boldsymbol{Y}) = \begin{cases} \frac{1}{\text{Equation(2)}}, & \text{feasible case,} \\ BIG, & \text{otherwise.} \end{cases}$$

3.3 Tree-based Reliability Calculation

Because only spanning tree topologies can be used as active network configurations, the evaluation of the reliability of network can often be accomplished by breaking down the problem into several problems on trees, and a single portion [Kershenbaum 1993].

We consider, as reliability measure, the probability of all operative nodes (centers and users) being connected. Now we want to calculate the reliability of a spanning tree network assuming the reliability of its elements, nodes and links, are known. Considering the tree to be a rooted tree, we associate a state vector with the root of each of subtrees. The state vector associated with a root node contains all information about that node relevant to our calculation. We then define a set of recursion relations which yield the state vector of a rooted tree given the state of its subtrees. For subtrees considering of single nodes the state is obvious. Then we join the rooted subtrees into larger and larger rooted subtrees using the recursion relations until the state of the entire network is obtained [Kershenbaum and Van Slyke 1973].

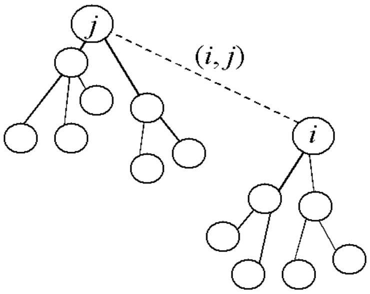


Figure 3: Recurrence Relation

Deriving the recurrence relations is somewhat mechanical. It comes simply from considering the situation depicted in Figure 3. We have two subtrees, one with root i and the other having as its root j. We assume the state of node i and node j are known and we wish to compute the state of j relative to the tree obtained by joining node i into node j with the link (i,j), where node j is the father node of node i.

Then we assume the notations for the procedure tree-based reliability calculation as follows: we have associated with each node i a probability of node failure p_i^f and a probability $p_i^o(= 1 - p_i^f)$ of node being operative. Similarly, for the link (i,j) we have probabilities l_i^f and l_i^o of the link (i,j) failing and being operative respectively. We also define the following state vectors for each subtrees: e_i means the probability that all nodes in the subtree are failed, o_i means the probability that the set of operative nodes, including the root of the subtree, are connected, and r_i means the probability that the root of the subtree is failed and the set of operative nodes in the subtree is connected.

For the tree with root node 1 and n nodes, we can calculate the reliability of tree, *i.e.*, the probability of all operating nodes communicating as follows:

Procedure: Reliability Calculation

Step 1: Set $r_i = 0$, $o_i = p_i^o$, $e_i = p_i^f$, $i = 1, 2, \cdots, n$. Set $i = n$. Go to step 2.

Step 2: If node j is the father node of node i, using the following recurrence relations, recalculate r_i, o_i, e_i:

$$\begin{aligned} r_j' &= r_j \cdot e_i + r_i \cdot e_j + o_i \cdot e_j \\ o_j' &= o_i \cdot o_j \cdot l_i^o + o_j \cdot e_i \\ e_j' &= e_i \cdot e_j \end{aligned}$$

Step 3: Set $i = i - 1$. If $i = 1$, go to step 4; otherwise go to step 2.

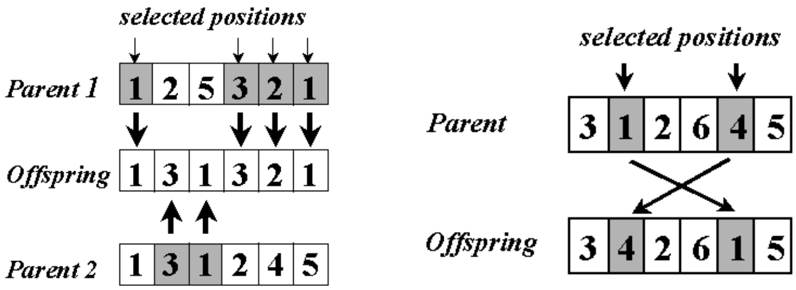
Step 4: Return $r_1 + o_1 + e_1$.

3.4 Selection

The selection used here is the method combined with the *roulette wheel* and *elitist* approach, in order to enforce the GA proposed to freely search solution space. The roulette wheel selection is used to randomly reproduce new generation and the elitist method is employed to preserve the best chromosome for the next generation and overcome the stochastic errors of sampling. Using this selection process, we can keep the best chromosome from the current generation to the next generation.

3.5 Genetic Operators

We employed the multi-point crossover (or called uniform crossover). This type of crossover is accomplished by selecting two parent solutions and randomly taking a component from one parent to form the corresponding component of the offspring, as shown in Figure 4(a). We used here swap mutation which simply select two positions at random and swap their contents, as represented in Figure 4(b).



(a) Crossover (b) Mutation

Figure 4: The Genetic Operators

3.6 Overall Algorithm

We propose the following bi-level genetic algorithm for solving reliable network design problems formulated by hierarchical optimization problem.

Step 1: Generate randomly *popSizeL* chromosomes as the initial population of upper-level (V_k^X, $k = 1, 2, \ldots, popSizeL$). Initialize the upper-level generation number $genL = 0$ and set the maximum generation $maxGenL$.

Step 2: For each V_k^X, repeat the following steps.

- **2-1:** Generate randomly *popSizeF* chromosomes as the initial population of lower-level (V_q^Y, $q = 1, 2, \ldots, popSizeF$). Initialize the lower-level generation number $genF = 0$ and set the maximum generation $maxGenF$.
- **2-2:** Apply crossover and mutation operators.
- **2-3:** After decoding with a given V_k^X and V_q^Y, evaluate chromosomes ($q = 1, 2, \ldots, popSizeF$).
- **2-4:** Perform selection to get the population of the next generation.
- **2-5:** If $genF < maxGenF$, then go to step 2-2, otherwise, go to step 3.

Step 3: Apply crossover and mutation operators.

Step 4: After decoding with V_k^X and $V^{\hat{Y}}$, evaluate chromosomes ($k = 1, 2, \ldots, popSizeL$).

Step 5: Perform selection.

Step 6: If $genL < maxGenL$, then go to step 2, otherwise, stop.

4 Numerical Examples

The spanning tree-based genetic algorithm proposed in this paper for solving bicriteria LAN topology design problem is implemented in C language and run on a Pentium II PC. The performance of this approach is tested with the following problem: we employ the problem with 6 service centers($n = 6$), 30 users($m = 30$),$C_i = 300$, and $\beta_{ij} = 0.1$. And the user matrix $\boldsymbol{U}$ is found at [Elbaum and Sidi 1996]. We also set the operative probability of centers as 0.95, the operative probability of users as 0.9, the operative probability between centers as 0.9, and the operative probability between center and user 0.85. The parameters for genetic algorithm are set as follows: $popSizeL = 100$, $popSizeF = 200$, $maxGenL = 200$, $maxGenF = 50$, $p_C = 0.4$, $p_M = 0.6$ and experimented by 20 times. The best result is

Centers: 3 2 3 1
Users: 5 4 4 6 4 5 3 3 1 6 5 1 6 2 4 2 1 2 3 3 2 1 1 3 1 5 5 5 5 2

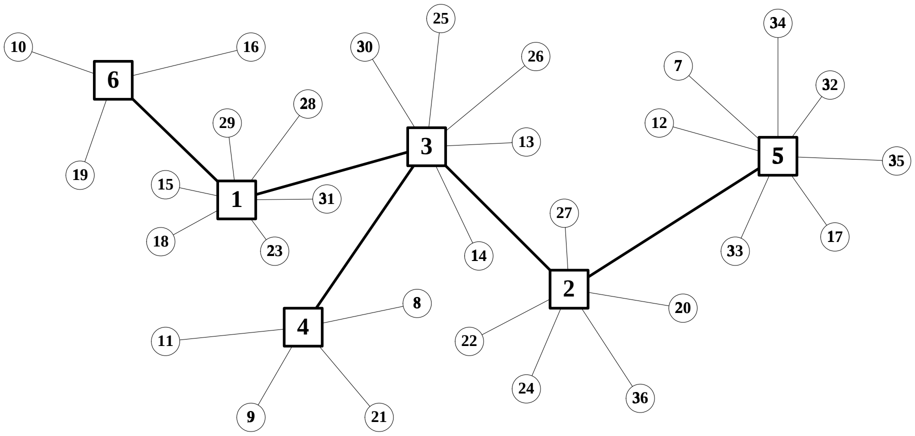


Figure 5: The Best Result

with connection cost 1377, average message delay 0.020446, and network reliability 0.920843. This result is shown in figure 5

We compare our result with the result of reference [Kim and Gen 1999] that is one with connection cost 1418, average message delay 0.16013, and network reliability 0.94758.

5 Conclusion

In this paper, we propose a bi-level hierarchical genetic algorithm for solving reliable network topology design problems with a structure that is comprised of the spanning tree of the centers and users using fiber-optic cables. Our problem has two kinds of decision variables and two kinds of objective functions. This kind of problem is formulated as bi-level programming problem to solve the decentralized planning problems with two decision makers in a bi-level hierarchical organization. In numerical experiments, we can see that the proposed approach has effectiveness, because that the proposed approach can search better solution than the result of reference [Kim and Gen 1999].

In future research, we will consider the more generalized numerical examples and the more suitable genetic operators considering the features of network design problems.

Acknowledgement

This research was supported by the International Scientific Research Program, the Grant-in-Aid for Scientific Research (No.10044173: 1998.4-2001.3) by the Ministry of Education, Science and Culture, the Japanese Government.

References

Bertsekas, D. and Gallager, R. (1992). *Data Networks*, 2nd ed., Prentice-Hall, New Jersey.

Elbaum, R. and Sidi, M. (1996). Topological design of local-area networks using genetic algorithms, *IEEE/ACM Transactions on Networking*, 4(5), 766-778.

Gen, M. and Cheng, R. (1997). *Genetic Algorithms and Engineering Design*, John Wiley & Sons, New York.

Gen, M. and Cheng, R. (1999). *Genetic Algorithms and Engineering Optimization*, John Wiley & Sons, New York.

Kershenbaum, A. (1993). *Telecommunication Network Design Algorithms*, McGraw-Hill, Inc., New York.

Kershenbaum, A. and Van Slyke, R. (1973). Recursive analysis of network reliability, *Networks*, 3, 81-94.

Kim, J. R. and Gen, M. (1999). Genetic Algorithm for Solving Bicriteria Network Topology Design Problem, *Proceedings of IEEE International Conference on Evolutionary Computation* (forthcoming).

Lai, Y. J. (1996). Hierarchical optimization: A satisfactory soultion, *Fuzzy Sets and Systems*, 77, 321-335.

Niwa, K., Nishizaki, I. and Sakawa, M. (1998). Decentalized two-level 0-1 programming through genetic algorithms with double strings, *Journal of Japan Society for Fuzzy Theory and Systems*, 10(4), 735-742 (Japanese).

Sakawa, M., Nishizaka, I. and Hitaka, M. (1999). Interactive fuzzy programming for multi-level 0-1 programming problems through genetic algorithms, *European Journal of Operational Research*, 114, 580-588.

Skiena, S. (1990). *Implementing Discrete Mathematics Combinatorics and Graph Theory with Mathematica*, Addison-Wesley, Reading, MA.

Tanenbaum, A. S. (1996). *Computer Networks*, 3rd ed., Prentice-Hall, New Jersey.

Zhou, G., and Gen, M. (1997). Approach to degree-constrained minimum spanning tree problem using genetic algorithm, *Engineering Design & Automation*, 3(2), 157-165.