높은 신뢰도의 네트워크 설계를 위한 GA 기반 두 단계 방법

조정복*

GA-based Two Phase Method for a Highly Reliable Network Design

Jung Bok Jo*

요 약

일반적으로 네트워크 설계 문제는 네트워크의 크기가 늘어남에 따라 지수적으로 복잡도가 증가하여 전통적인 방법으로는 풀이하기 힘든 NP-hard 조합 최적화 문제 중의 하나로 분류될 수 있다. 본 논문에서는 네트워크 신뢰도 제약을 고려하면서 네트워크 구축비용을 효과적으로 최소화하는, 높은 신뢰도의 네트워크 토폴로지 설계 문제를 풀기 위해 스패닝 트리를 효율적으로 표현할 수 있는 Prüfer수(PN) 기반의 진화 연산법과 2-연결성을 고려하는 휴리스틱 방법으로 구성된 두 단계의 효율적인 해법을 제안한다. 즉, 먼저 스패닝 트리를 찾아내기 위해 진화 연산법 중에 보편적으로 널리 알려져 있는 유전자 알고리즘(GA)을 이용하고 그 다음으로 첫 번째 단계에서 발견한 스패닝 트리에 대해 최적의 네트워크 토폴로지를 찾기 위해서 2-연결성을 고려한 휴리스틱 방법을 적용한다. 마지막으로 수치예의 결과를 통해 제안한 해법의 성능에 대해서 살펴보도록 한다.

ABSTRACT

Generally, the network topology design problem, which is difficult to solve with the classical method because it has exponentially increasing complexity with the augmented network size, is characterized as a kind of NP-hard combinatorial optimization problem. The problem of this research is to design the highly reliable network topology considering the connection cost and all-terminal network reliability, which can be defined as the probability that every pair of nodes can communicate with each other. In order to solve the highly reliable network topology design problem minimizing the construction cost subject to network reliability, we proposes an efficient two phase approach to design reliable network topology, i.e., the first phase employs, a genetic algorithm (GA) which uses Prüfer number for encoding method and backtracking Algorithm for network reliability calculation, to find the spanning tree; the second phase is a greedy method which searches the optimal network topology based on the spanning tree obtained in the first phase, with considering 2-connectivity. Finally, we show some experiments to demonstrate the effectiveness and efficiency of our two phase approach.

키워드

Network Reliability, Combinatorial Optimization, Communication Network, Genetic Algorithm (GA).

Ⅰ. Introduction

There has been an explosive growth in computer networks since the 1980s. Computer networks pervade our every daily life, from automated teller machines, to airline reservation systems, to electronic mail services, to

* 동서대학교 인터넷공학부

접수일자 : 2005. 7. 12

electronic bulletin boards, to the Internet. There are many reasons for the explosive growth in computer networks. We are in an information age and computer networks have been becoming an integral part in the dissemination of information and in our daily life.

Network design problems are one of important issues in the building and expansion of computer networks and have attracted many related researchers' attentions, such as network designers, network analysts, network administrators. These network problems have many applications in telecommunications, computer networking, and the related domains in electric, gas, and sewer networks. Especially, considering network reliability, the networks topology design has a major importance in the computer communication industry. The optimal topology design problems can be formulated as combinatorial optimization problems. Generally, the network topology design problem, which is difficult to solve with the classical method because it has exponentially increasing complexity with the augmented network size, is characterized as a kind of NP-hard combinatorial optimization problem. The problem of this research is to design the optimal network topology considering the connection cost and network reliability. There are the following two kinds of reliabilities, usually used in the network design problems.

All-terminal network reliability (also called overall network reliability): The probability that each and every node in the network is connected to each other, i.e., every node in the network can communicate with every other node over a specified mission time.

Source-sink network reliability: The probability that the source is connected with the sink, it means that the source node in the network can communicate with the sink node over a specified mission time.

Many ideas and methods for solving these networks design problems have been proposed and tested in the past two decades. Recently, there is an increasing interesting in applying genetic algorithms (GAs) to the problems related to computer networks. The solution approaches for optimal network design problems considering these reliabilities are proposed as follows: enumeration-based approaches, heuristic-based approaches, and GA-based approaches.

Enumeration-based approaches are applicable only for small-size networks. Theses enumeration-based approaches are founded on the enumeration of states, minpaths or mincuts. Roughly speaking, these methods use some combination of enumeration and reduction techniques. Enumeration-based methods enumerate a set of probabilistic events which are mutually exclusive and collectively exhaustive with respect to the measure. Jan, Hwang, and Chen proposed an algorithm using decomposition based on branch-and-bound to minimize edge costs with a minimum network reliability constraint [16]. Aggarwal and Rai proposed a method based on spanning trees to evaluate the reliability of computer networks [1]. Wilkov proposed a method to design computer networks with a new reliability measure [27].

Heuristic-based approaches, such as tabu search, simulated annealing, and so on, can be applied to larger networks but do not guarantee optimality. Aggarwal, Chopra, and Bajwa applied greedy heuristic approach to the computer network design problems considering the overall reliability [2]. Chopra, Sohi, Tiwari, and Aggarwal proposed a greedy heuristic approach for the design of network topology to maximize the terminal reliability [6]. Venetsanopoulos and Singh used a two-step heuristic for minimizing cost subject to a reliability constraint [26]. Glover, Lee, and Ryan proposed a method using tabu search in order to design the network topology with least-cost [14]. Koh applied a tabu search to the reliable fiber optic communication network design [18]. Atiqullah and Rao proposed a method using simulated annealing to optimize the reliability of communication networks [3]. Fetterlof et al. applied simulated annealing to the optimization of LAN-WAN internetworking design [10]. Pierre, Hyppolite, Bourjolly, and Dioume proposed an algorithm for the topology design of computer communication networks using simulated annealing [24].

Recently, GA-based approaches, which can be classified as one of meta-heuristics, have taken attention

as the tool of solution method for the optimal design of network considering reliability and are often used to solve many real-world problems [12][13]. GA approaches have been reported in many applications, such as the transportation problems, the minimum spanning tree problems, the production process planning problems, and so on. Kumar, Pathak, Gupta, and Parsaei proposed a GA to design the distributed system topology [20]. Kumar, Pathak, and Gupta developed a GA considering diameter, distance, and reliability to design and expand computer networks [21]. Deeter and Smith presented a GA-based approach to design of networks considering all-terminal reliability with alternative edge options, allowing edges to be chosen from different components with different costs and reliabilities [7]. Dengiz, Altiparmak, and Smith attempted to apply genetic algorithm to the optimal design of networks considering all terminal [8]. Dengiz, Altiparmak, and Smith focuses on large backbone communication network design considering the all-terminal network reliability and used a GA, but appreciably customizes it to the all-terminal design problem to give an effective, efficient optimization methodology [9].

In this paper, we attempt to solve the highly reliable network topology design problem which is minimizing the cost subject to network reliability using the proposed method. Our proposed approach consists of two phases: the first phase employs a GA based method that is the most widely known type of evolutionary computation approaches, to find a spanning tree; the second phase is a greedy method which searches the optimal network topology based on the obtained spanning tree in the first phase, considering the 2-connectivity. We also employ the Prüfer number encoding as the chromosome representation of GA to represent the spanning tree, which is capable of equally and uniquely representing all possible trees [12]. Using Prüfer number encoding, the memory only requires n-2 on the problem with n nodes. The objective of the design problem is to minimize the cost required to design the networks system under a network reliability constraint. We use the backtracking algorithm to calculate the network reliability of a candidate design x. A backtracking algorithm from [4] is used to exactly calculate the system unreliability, $1\text{-}R(x)$, for the problems due to their computationally tractable size. Finally, we show some experiments in order to certify the quality of the networks designs obtained by using the proposed two phase method.

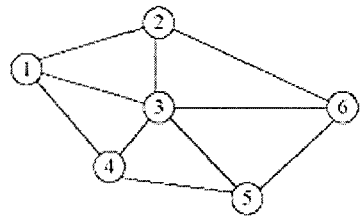


Fig. 1 Sample Network Architecture

Ⅱ. Reliable Network Topology Design

A communication network can be represented by a undirected graph $G = (N, E)$, in which nodes N and edges E mean computer sites and communication cables, respectively, as shown in Figure 1. Any graph G is connected if there is at least one path between every pair of nodes i and j, which minimally requires a spanning tree with $(n\text{-}1)$ edges. The number of possible edges is $n\times(n\text{-}1)/2$. We define the following notations to describe the optimal design problem of all-terminal reliable networks: n is the number of nodes, $x_{ij} \in \{0, 1\}$ is a decision variable representing edge between node i and node j, $\boldsymbol{x}$ (= $(x_{12}, x_{13}, \cdots, x_{\{n\text{-}1,n\}})$) is a topology architecture of network design, $\boldsymbol{x}^*$ is the best solution found so far, p is edge reliability for all edges, q is edge unreliability for all edges (i.e., $p + q = 1$), $R(\boldsymbol{x})$ is the all-terminal reliability of network design $\boldsymbol{x}$, $R_{\min}$ is a network reliability requirement, c_{ij} is cost of edge between node i and node j, δ has the value of 1 if $R(\boldsymbol{x}) \geq R_{\min}$, otherwise, has the value of 0, E' is a set of operational edges ($E' \subseteq E$), Ω is all operational states (E').

$$\min Z(\boldsymbol{x}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} \cdot x_{ij}$$

$$s.t. \quad R(\boldsymbol{x}) \geq R_{\min}$$

Also we have the following assumptions: the location of each node is given, nodes are perfectly reliable, each cij and p are fixed and known, each edge is bi-directional, there are no redundant edges in the network, edges are either operational or failed (non-operational), the failure of edges are mutually statistical-independent, and there is no repair.

At any mission-time, only some edges of G might be operational. A operational state of G is a sub-graph G' = (N, E'). The network reliability of state E' E is as follows:

$$\sum_{\Omega} \left[\prod_{e \in E'} p_e \right] \cdot \left[\prod_{e \in (E-E')} q_e \right]$$

Ⅲ. Proposed Approach

3.1 GA: The First Phase

Many optimization problems from the real world, in particular the manufacturing systems and network systems, are very complex in nature and quite hard to solve by conventional optimization techniques. Since the 1960s, there has been increasing interest in imitating living beings to solve such kinds of hard optimization problems. Simulating the natural systems results in stochastic optimization techniques called evolutionary computation methods, which can often outperform conventional optimization methods when applied to difficult real-world problems. There are currently four main avenues of this research: genetic algorithms, evolutionary programming, evolution strategies, and genetic programming. Among them, genetic algorithms (GAs) are perhaps the most widely known type of evolutionary computation methods today [12][13].

Recently, GAs have received considerable attention regarding their potential as an optimization technique for complex problems and have been successfully applied in the area of industrial engineering. Then well known applications including scheduling and sequencing, reliability design, vehicle routing and scheduling, group technology, facility layout design and location, transportation, network design and routing, and may others. This paper employs the GA, based on Prüfer number encoding method, as the optimization technique of the first phase in order to find spanning tree network configuration for the second phase.

3.1.1 Representation 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. One of the classical theorems in graphical enumeration is Cayley's theorem 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 [12][13]. 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 [25]. Based on this observation we can easily construct an encoding as follows:

Procedure: Encoding of Prüfer number

Step 1: Let node i be the smallest labeled leaf node in a labeled tree T.

Step 2: Let j be the first digit in the encoding as the node j incident to node i is uniquely determined. Here we build the encoding by appending digits to the right, and thus the encoding is built and read from left to right.

Step 3: Remove node i and the edge from i to j thus we have a tree with k-1 nodes.

Step 4: Repeat the above steps until one edge is left. We produce a Prüfer number or an encoding with k-2

digits between 1 and k inclusive.

It is also possible to generate a unique tree from a Prüfer number via the following procedure:

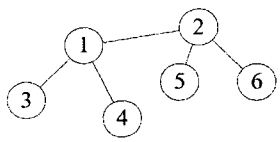
Procedure: Decoding of Prüfer number

Step 1: Let P be the original Prüfer number and let P' be the set of all nodes not included in P, which designates as eligible nodes for consideration in building a tree.

Step 2: Let i be the eligible node with the smallest label. Let j be the leftmost digit of P. Add the edge from i to j into the tree. Remove i from P' and j from P. If j does not occur anywhere in P, put it into P'. Repeat the process until no digits are left in P.

Step 3: If no digits remain in P, there are exactly two nodes, r and s, still eligible for consideration. Add edge from r to s into the tree and form a tree with k-1 edges.

An example is given to illustrate this kind of encoding. Prüfer number (1 1 2 2) corresponding to a spanning tree on a five node complete graph is represented in Figure 2. According to the encoding procedure, the construction of the Prüfer number is described as follows: Locate the leaf node having the smallest label. In this case, it is node 3. Since node 1 is adjacent to node 3 in the tree, assign 1 to the first digit in the Prüfer number, and then remove node 3 and the edge (3, 1). Repeat the process on the sub-tree until edge (2, 6) is left and the Prüfer number of this tree with four digits is finally produced.



1	1	2	2

Prüfer number

Fig. 2 A Tree and Its Prüfer Number

On the contrary, we can construct the tree from the Prüfer number using the decoding procedure. For the Prüfer number P = (1 1 2 2), the nodes 3, 4, 5, and 6 are eligible and P' = {3, 4, 5, 6}. Node 3 is the eligible node with the smallest label. Node 1 is the leftmost digit in P. Add edge (3, 1) to the tree, remove node 3 from P' for further consideration, and remove the leftmost digit 1 of P leaving P = (1 2 2). Node 4 is now the eligible element with the smallest label and the second node 1 is the leftmost digit in remaining P. Then add edge (4, 1) to the tree, remove node 4 from P' for further consideration, and remove the digit 1 from P leaving P = (2 2). Because node 1 is now no longer in the remaining P, it becomes eligible and is put into P' = {1, 5, 6}. Then, node 1 is the eligible node with the smallest label, add edge (1, 2) to the tree, remove the leftmost digit 2 of P, and remove node 1 from P'. Now P = (2) and only nodes 5 and 6 are eligible. Add edge (5, 2) to the tree, remove the last digit of P, and designate node 5 is not eligible and remove it from P'. Because node 2 is now no longer in the remaining P, it is added to P'. P is now empty and only nodes 2 and 6 are eligible. Thus add edge (2, 6) to the tree and stop. The tree of Figure 2 is formed.

Prüfer 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 appears d times in Prüfer number, the node exactly has d+1 connections with other nodes.

Prüfer number is more 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 Prüfer number is addressed in [12][13].

From Figure 2, we can see that all digits of the Prüfer number are the figures of non-leaf nodes. The initialization of a chromosome (i.e., a Prüfer number) is performed by that it randomly generates n-2 digits in range [1, n]. Therefore in a chromosome, the non-leaf node is represented but the leaf nodes do not appear in a chromosome and only is used as the elements of the set P'. Here suppose the network has n nodes.

3.1.2 Evaluation

To calculate the value of an evaluation function $eval(v_k)$ for reliable network design problems, we firstly transform a chromosome v_k into an adjacency matrix whose element represents whether the specific edge connects with the pair of nodes or not.

Then we calculate the objective value and evaluate $eval(v_k)$ as follows:

$$eval(v_k) = \begin{cases} 1/Z(v_k), & \text{if } \delta \text{ is equal to 1;} \\ 1/\text{BIG}, & \text{otherwise.} \end{cases}$$

where BIG means a very large number.

By the following equation, we can keep the best chromosome v^* with largest fitness value in each generation. $v^* = \text{argmax}\{eval(v_k)|k=1,2,\cdots, pop\_size\}$ where argmax means argument maximum.

3.1.3 Reliability Calculation

Another active area of research related to the network design problem is how to calculate or estimate network reliability. There are two main approaches: exact calculation through analytic methods and estimation through variations of Monte Carlo simulation.

All known analytic methods for all-terminal network reliability calculation have worst-case computation times which grow exponentially in the size of the network. Hence, these analytic methods are not recommended for large networks. Ball and Van Slyke [4] proposed a method of enumerating modified cutsets by backtracking to compute the overall reliability of an undirected graph. Hanzhong and Dongkui [15] proposed a new method for computing the complex network reliability using the cut method. Mandaltsis and Kontolen [23] researched the method for calculating overall reliability of computer networks which have the hierarchical routing strategies. Liu et al. [22] proposed a solving method for the calculation of a network overall reliability.

For the all-terminal network reliability problem, efficient simulation is especially difficult because these methods generally lose efficiency as the network approaches a fully connected state. However, simulation methods are suitable for large networks because computation time taken by the simulation method grows only slightly faster than linear with network size. Cancela and Khadiri [5] used a recursive variance-reduction algorithm with Monte Carlo method to estimate communication network reliability. Fishman [11] studied about a Monte Carlo Sampling Plan to estimate network reliability and analyzed about four Monte Carlo methods to estimate the probability of source-sink connectedness. Kamat and Riley [17]researched the determination method of reliability with event-based Monte Carlo simulation. Kumamoto et al. [19]proposed an efficient evaluation method of system reliability using Monte Carlo method. Yeh et al. [28] proposed a new Monte Carlo method in order to estimate network reliability.

A backtracking algorithm is used to exactly calculate the system unreliability, $1-R(x)$, for the problems due to their computationally tractable size. The backtracking algorithm outline is given by the following procedure:

Step 0: Initialization

Mark all edges as free and create a stack S which is initially empty.

Step 1: Generate modified cutset

Step 1-1: Find a set of free edges that together with all inoperative edges will form a network-cut.

Step 1-2: Mark all the edges found in Step 1-1 inoperative and add them to the stack.

Step 1-3: Now the stack represents a modified cutset; Add its probability to a cumulative sum.

Step 2: Backtrack

Step 2-1: If the stack is empty, go to Step 3; otherwise go to Step 2-2.

Step 2-2: Take edge off the top of the stack.

Step 2-3: If the edge is inoperative and if when make it operative, a spanning tree of operative edges exists, then mark it free and go to Step 2-1.

Step 2-4: If the edge is inoperative and the condition tested in Step 2-3 does not hold, then mark it operative, put it back on the stack and go to Step 1.

Step 2-5: If the edge is operative, then mark it free and go to Step 2-1.

Step 3: Return the network unreliability and end the procedure.

where the probability of all edges within stack is the product of the failure probabilities of all inoperative edges times the product of one minus the failure probabilities of all operative edges.

3.1.4 Selection

In a GA, the selection plays a very important role. When we regard the genetic operation as the exploration for the search in solution space, the selection can be thought as the exploitation for the GA to guide the evolutionary process. The selection used here is the method combined with the roulette wheel and elitist approach, in order to enforce the proposed GA to freely search solution space. The roulette wheel selection, which is one of the fitness-proportional methods, 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. Then we can express our selection method used here as follows:

Step 1: Calculate a cumulative probability a_p for each chromosome X_p, (p=1, 2, ···, *chr_size*).

Step 2: Generate a random real number r in [0, 1].

Step 3: If $r \leq a_1$, then select the first chromosome X_1 otherwise select the p-th chromosome X_p ($2 \leq p \leq$ *chr_size*) such that $a_{p-1} < r \leq a_p$.

Step 4: Repeat Steps 2 and 3 *pop_size* times and obtain *pop_size* copies of chromosomes.

Step 5: If the best chromosome is not selected in the next generation, replace one randomly from new population by the best one.

Using this selection process, we can keep the best chromosome from the current generation to the next generation.

3.1.5 Crossover and Mutation

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.

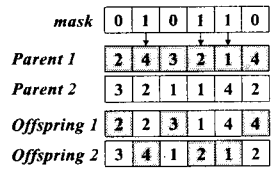


Fig.3 Multi-point Crossover Operator

Swap mutation is used in this paper, which simply selects two positions at random and swaps their contents as shown in Figure 4.

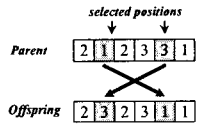


Fig. 4 Swap Mutation Operator

3.2 Greedy Method with 2-connectivity: the Second Phase

The solution searched in the first phase, represents only spanning tree.

For the best chromosome obtained in the first phase, we firstly decode it. Then as calculating the node degree of each nodes, we check whether 2-connectivity is satisfied or not to the best result of the first phase (i.e., if the decoded network topology has one or more nodes with node degree < 2, it is not satisfied). Since the best chromosome form first phase has spanning tree configuration, it dose not pass 2-connectivity check. Then we repair greedily the adjacency matrix of the best result of the first phase with the following rules: one is that if the node degree is larger than two then remove the edge with the largest connection cost among the edges connected with the node until the degree of the node is near to two; the other is that if the node degree is less than two then add the edge with smallest connection cost among the edges which can be connected to the node

until the degree of the node is near to two.

Step 1: Decode the best chromosome of GA into adjacency matrix.

Step 2: Check 2-connectivity as calculating the node degree of each nodes. If 2-connectivity check is not satisfied (if the adjacency matrix has one or more nodes with node degree < 2), go to Step 3; otherwise go to Step 4.

Step 3: Modify greedily the adjacency matrix with the following rules: one is that if the node degree is larger than two then remove the edge with largest connection cost among the edges connected with the node; the other is that if the node degree is less than two then add the edge with smallest connection cost among the edges which can be connected to the node.

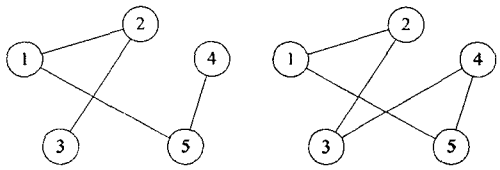
Step 4: If the degree of all nodes is satisfied with 2-connectivity, then stop otherwise go to Step 3.

An illustrative example of the above method is presented as follows. A network with five nodes and edges costs of

$$c_{ij} = \begin{bmatrix} - & 32 & 54 & 62 & 25 \\ & - & 34 & 58 & 45 \\ & & - & 36 & 52 \\ & & & - & 29 \\ & & & & - \end{bmatrix}$$

is shown in Figure 5 (a).

The node 3 and 4 are not satisfied 2-connectivity. Therefore, we modify the connectivity of node 3 and 4. Firstly, for node 3, we search the edge with the smallest cost, which can be connected. In this example, among the edges (1, 3), (3, 4), (3, 5), edge (3, 4) has the smallest cost. Then we add the edge (3, 4) into network. After modifying, we obtain the network shown in Figure5 (b).



(a) Spanning Tree (b) Modified Topology

Fig. 5 A network with five nodes that is modified for 2-connectivity by adding an edge (3,4)

Table 1. The Comparison of Results

Problem	n	N_e	p	R_{req}	B&B	GA			Proposed Method		
					Obtained Cost	Best Cost	Worst Cost	Mean Cost	Best Cost	Worst Cost	Mean Cost
1	5	10	0.80	0.90	255	201	252	215.2	156	156	156.0
2	5	10	0.90	0.95	201	156	218	189.0	156	156	156.0
3	7	21	0.90	0.90	720	630	800	703.0	630	755	680.5
4	7	21	0.90	0.95	845	630	815	702.0	630	680	646.5
5	7	21	0.95	0.95	630	670	775	708.5	630	186	652.0
6	8	28	0.90	0.90	208	173	231	194.3	151	184	176.6
7	8	28	0.90	0.95	247	173	243	192.1	179	184	181.0
8	8	28	0.95	0.95	179	179	212	192.4	151	184	177.4
9	9	36	0.90	0.90	239	209	324	253.3	207	247	237.3
10	9	36	0.90	0.95	286	211	291	255.7	209	248	233.7
11	9	36	0.95	0.95	209	209	276	249.0	207	261	233.0
12	10	45	0.90	0.90	154	144	217	169.8	144	192	163.2
13	10	45	0.90	0.95	197	144	239	166.7	146	176	159.0
14	10	45	0.95	0.95	136	144	248	188.1	136	176	148.0
15	15	105	0.90	0.95	--	--	--	--	183	266	218.3
16	20	190	0.95	0.95	--	--	--	--	157	281	201.7
17	25	300	0.95	0.90	--	--	--	--	308	567	453.3

N_e: The number of edges
B&B: Branch and Bound method proposed by reference [16]
GA: Genetic Algorithm proposed by Dengiz, Altiparmak, and Smith
-- : Solution cannot be founded.

Table 2. Comparison of Search Efforts

Problem	Search Space	Solutions Searched	Fraction Searched	R_{req}	GA R(x) (Mean)	Proposed Method R(x) (Mean)
1	1.02 E03	6.00 E02	5.86 E-01	0.90	0.9653	0.9379
2	1.02 E03	6.00 E02	5.86 E-01	0.95	0.9884	0.9877
3	2.10 E06	1.50 E04	7.14 E-03	0.90	0.9879	0.9875
4	2.10 E06	1.50 E04	7.14 E-03	0.95	0.9878	0.9869
5	2.10 E06	1.50 E04	7.14 E-03	0.95	0.9972	0.9971
6	2.68 E08	2.00 E04	7.46 E-05	0.90	0.9884	0.9896
7	2.68 E08	2.00 E04	7.46 E-05	0.95	0.9884	0.9888
8	2.68 E08	2.00 E04	7.46 E-05	0.95	0.9973	0.9974
9	6.87 E10	4.00 E04	5.82 E-07	0.90	0.9881	0.9888
10	6.87 E10	4.00 E04	5.82 E-07	0.95	0.9881	0.9888
11	6.87 E10	4.00 E04	5.82 E-07	0.95	0.9973	0.9974
12	3.52 E13	8.00 E04	2.27 E-09	0.90	0.9881	0.9889
13	3.52 E13	8.00 E04	2.27 E-09	0.95	0.9885	0.9890
14	3.52 E13	8.00 E04	2.27 E-09	0.95	0.9973	0.9974
15	4.06 E31	1.40 E05	3.45 E-27	0.95	--	0.9976
16	1.57 E57	2.00 E05	1.27 E-52	0.95	--	0.9977
17	2.04 E90	4.00 E05	1.96 E-85	0.90	--	0.9981

Ⅳ. Experimental Results

The test problems are summarized in Table 1. These problems are fully connected networks. Node for the networks ranges from 5 to 25. The available edges of networks were ranged from 10 to 300. The edge costs for all networks were randomly generated over the range [1, 100] except for problems 3-5 which used selected costs over the range [75, 250]. Each problem for the proposed method was run 10 times, each time with a different random number seed. We implement our proposed method by C language and run on a Pentium PC. Solutions, as obtained by the B&B method proposed by reference [16], are given in Table 1. The method in [16] cannot be practically used on the larger problems such as problems 15-17 because of the amount of computation in the B&B procedure. And in Table 1, we can see that our proposed method has better performance than the GA proposed by Dengiz, Altiparmak, and Smith [8]. Also, the GA method by Dengiz, Altiparmak, and Smith cannot be practically used on the larger problems (problems 15-17) because of the needed memory space when computing. Table 2 lists the search space for each problem along with the proportion actually searched by the our approach and the GA during a single run *pop_size* × *max_gen*. *max_gen* ranged from 30-20000, depending on the problem size. This proportion is the upper bound because our approach and the GA can (and often do) revisit solutions already considered earlier in the evolutionary search. Our method and the GA approach examine only a very tiny fraction of the possible solutions for the larger problems, yet still yield optimal or near-optimal solution. Table 2 also represents that our approach can find more reliable network configuration than the GA by Dengiz,

Table 3. Comparison of Memory Space and Computation Time

Problem	n	GA		Proposed Method	
		Memory $(n(n-1))/2$	ACT (sec.)	Memory $n-2$	ACT (sec.)
1	5	10	0.2172	3	0.0841
2	5	.10	0.1831	3	0.0790
3	7	21	10.1074	5	7.0963
4	7	21	9.8602	5	7.5468
5	7	21	12.1495	5	8.7879
6	8	28	41.2264	6	12.9225
7	8	28	41.0972	6	11.0289
8	8	28	40.1506	6	11.1440
9	9	36	230.9028	7	26.2449
10	9	36	317.4627	7	35.7372
11	9	36	308.8991	7	37.9256
12	10	45	1834.2825	8	93.1180
13	10	45	2191.8337	8	98.2712
14	10	45	2172.2625	8	64.9963
15	15	105	--	13	2324.4823
16	20	190	--	18	6895.8018
17	25	300	--	23	10322.8720

Altiparmak, and Smith.

Table 3 compares the efficiency of our proposed method with the GA by Dengiz, Altiparmak, and Smith in case of the required memory and computation time. From the Table 3, we can see that our proposed method get optimal or near-optimal solution more quickly than the GA with smaller memory size required to computations.

V. Conclusion

In this paper, we tried to solve the network topology design problem which is minimizing the cost subject to etwork reliability using our proposed method, which is comprised of two phases, i.e., GA approach based on Prüfernumber encoding and a heuristic method. In our approach, firstly, to find the spanning tree, GA was used; secondly, a heuristic method is employed, in order to search the optimal network topology based on the spanning tree obtained in the first phase, considering 2-connectivity. We employed the Prüfer number encoding as the chromosome representation of GA to represent the spanning tree, which is capable of equally and uniquely representing all possible trees. Using Prüfer number encoding, the memory only required n-2 on the problem with n nodes. The objective of the design problem was to minimize the cost needed to design the networks system subject to network reliability. We also used the backtracking algorithm to calculate the network reliability of a candidate design x. A backtracking algorithm from [4]was used to exactly calculate the system unreliability, $1-R(x)$, for the problems due to their computationally tractable size. Finally, inorder to certify the quality of the networks designs obtained by using proposed method, we compared the results of our approach with the results of GA by Dengiz, Altiparmak, and Smith. From the results, we can see that our approach has effectiveness and efficiency.

Recent researches for the genetic representation of spanning trees [29][30] have reported the weak point of Prüfer number encoding method because it has poor locality and heritability. So, our future research will be tried to overcome this shortcoming of the GA employed in the first phase of our proposed method.

References

[1] K.K. Aggarwal and S. Rai, "Reliability evaluation of computer-communication networks", IEEE Trans. Rel., vol.R-30, no.1, pp.32-35, 1981.

[2] K.K. Aggarwal, Y.C. Chopra, and J.S. Bajwa, "Topological layout of links for optimising the overall reliability in a computer communication", Microelectronics & Reliability, vol.22, no.3, pp.347-351, 1982.

[3] M.M. Atiqullah and S.S. Rao, "Reliability optimization of communication networks using simulated annealing, Microelectronics & Reliability, vol.33, no.9, pp.1303-1319, 1993.

[4] M. Ball and R.M. Van Slyke, "Backtracking algorithms for network reliability analysis", Annals of Discrete Mathematics, vol.1, pp.49-64, 1977.

[5] H. Cancela and M.E. Khadiri, "A recursive variance-reduction algorithm for estimating communication-network reliability", IEEE Trans. Rel., vol.44, no.4, pp.595-602, 1995.

[6] Y.C. Chopra, B.S. Sohi, R.K. Tiwari, and K.K. Aggarwal, "Network topology for maximizing the terminal reliability in a computer communication networks", Microelectronics & Reliability, vol.24, pp.911-913, 1984.

[7] D.L. Deeter and A.E. Smith, "Economic design of reliable network", IIE Trancstion, vol.30, no.12, pp.1161-1174, 1999.

[8] B. Dengiz, F. Altiparmak, and A.E. Smith, "Efficient optimization of all-terminal reliable networks using evolutionary approach", IEEE Trans. Rel., vol.46, no.1, pp.18-26, 1997.

[9] B. Dengiz, F. Altiparmak, and A.E. Smith, "Local search genetic algorithm for optimal design of reliable networks", IEEE Trans. Evolutionary Computation, vol.1, no.3, pp.179-188, 1997.

[10] P.C. Fetterlof and G. Anandalingam, "Optimal design of LAN-WAN internetworks: an approach using simulated annealing", Annals of Opers. Research, vol.36, pp.275-298, 1992.

[11] G.S. Fishman, "A comparison of four monte carlo methods for estimating the probability of s-t connectedness", IEEE Trans. Rel., vol.R-35, no.2, pp.145-155, 1986.

[12] M. Gen and R. Cheng, "Genetic Algorithms and Engineering Design", John Wiley & Sons, New York, 1997.

[13] M. Gen and R. Cheng, "Genetic Algorithms and Engineering Optimization", John Wiley & Sons, New York, 2000.

[14] F. Glover, M. Lee, and J. Ryan, "Least-cost network topology design for a new service: an application of a tabu search", Annals of Opers. Research, vol.33, pp.351-362, 1991.

[15] C. Hanzhong and L. Dongkui, "A new algorithm for computing the reliability of complex networks by the cut method", Microelectronics & Reliability, vol.34, no.1, pp.175-177, 1994.

[16] R.H. Jan, F.J. Hwang, and S.T. Chen, "Topological optimization of a communication network subject to a reliable constraint", IEEE Trans. Rel., vol.42, no.1, pp.63-70, 1994.

[17] S.J. Kamat and M.W. Riley, "Determination of reliability using event-based monte carlo simulation", IEEE Trans. Rel., vol.R-24, no.1, pp.73-75, 1975.

[18] S.J. Koh and C.Y. Lee, "A tabu search for the survivable fiber optic communication network design", Comput. & Indust. Eng., vol.28, no.4, pp.689-700, 1995.

[19] H. Kumamoto, K. Tanaka, and K. Inoue, "Efficient evaluation of system reliability by monte carlo method, IEEE Trans. Rel., vol.R-26, no.5, pp.311-315, 1977.

[20] A. Kumar, R.M. Pathak, Y.P. Gupta, and H.R. Parsaei, "A genetic algorithm for distributed system topology design", Comput. & Indust. Eng., vol.28, no. 3, pp.659-670, 1995.

[21] A. Kumar, R.M. Pathak, and Y.P. Gupta, "Genetic-algorithm-based reliability optimization for computer network expansion", IEEE Trans. Rel., vol.44, no.1, pp.63-72, 1995.

[22] C. Liu, M. Dai, X.Y. Wu, and W.K. Chen, "A network overall reliability algorithm", Proc. Neural, Parallel & Scientific Computations, pp.287-292, Atlanta, 1995.

[23] D. Mandaltsis and J.M. Kontolen, "Overall reliability determination of computer networks with hierarchical routing strategies", Microelectronics & Reliability, vol.27, no.1, pp.129-143, 1987.

[24] S. Pierre, M. Hyppolite, J. Bourjolly, and O. Dioume, "Topological design of computer communication networks using simulated annealting", Eng. Applications of Artificial Intelligence, vol.8, no.1, pp.61-69, 1995.

[25] S. Skiena, "Implementing Discrete Mathematics Combinatorics and Graph Theory with Mathematica", Addison-Wesley, Reading, MA, 1990.

[26] A.N. Venetsanopoulos and I. Singh, "Topological optimization of communication networks subject to reliability constraints",Problem of Control and Information Theory, vol.15, pp.63-78, 1986.

[27] R.S. Wilkov, "Design of computer networks based on a new reliability measure", Sym. Computer-communications Networks and Teletraffic, Ploytechnic Institute of Brooklyn, pp.371-384, 1972.

[28] M.S. Yeh, J.S. Lin, and W.C. Yeh, "A new monte carlo method for estimating network reliability", Proc. 16th Inter. Conf. on Comput. & Indust. Eng., pp.723-726, 1994.

[29] G.. R. Raidl and B. A. Julstrom, "Edge sets: an effective evolutionary coding of spanning trees", IEEE Trans. on Evol. Compu., Vol.7, No.3, pp.225-239, June, 2003.

[30] F. Altiparmak, M. Gen, B. Dengiz, and A. E. Smith, "A network-based genetic algorithm for design of communication networks," Journal of Society of Plant Engineers Japan, Vol. 15, No, 4, 184-190, Feb. 2004.

저자 소개

Jung Bok Jo

He received the B.S. degree in computer engineering from Kyungpook University in 1978 and the M.S. degree in electronic engineering from Yeungnam University in 1986. He received the Ph.D. degree in system engineering science from Tokyo Metropolitan Institute of Technology in 1996. He is currently an assistant professor in division of internet engineering, Dongseo University and visiting professor at graduate school of information, production and systems, Waseda University, Japan. His research interests genetic algorithms, fuzzy queueing system, and the applications of soft computing.