Journal of the Society of Korea Industrial and Systems Engineering
Vol, 28, No. 1, pp.8－15, March 2005.

Economic Design for Expanding Networks using Hybrid Genetic Approach

Chang-Sun Yum · Sang-Wook Bae

Division of Business Administration, Pukyong National University

하이브리드 유전자 접근방법을 이용한 네트워크 확장을 위한 경제적 설계

염창선 · 배상욱

부경대학교 경영학부

이 논문에서는 백본 네트워크의 경제적 확장 방법을 제시한다. 기존 연구에서의 네트워크 확장은 기 설치된 네트워크에 새로운 노드를 추가하는 데 그치고 있다. 이 논문에서는 새로운 노드의 추가 외에 기 설치된 링크의 용량 확장 까지를 고려하는 경제적 확장 설계 방법을 제시한다. 경제적 네트워크 확장은 기 설치된 네트워크와 performability를 제약조건으로 하면서 비용을 최소화하는 문제로 표현된다. 효율적인 해의 탐색을 위해 이진과 K진이 혼용된 하이브리드 유전자 접근 방법을 사용한다.

Keywords : Network Expansion, Economic Design, Performability, Hybrid Genetic Approach

1. Introduction

The backbone network is dedicated to the delivery of information from source to destination using its switching nodes. At the initial deployment of backbone networks, it is very difficult to design and build with considering fast growing demand. In order to meet this demand, the size of the backbone network is gradually expanded according to user requirements. It is critical that new nodes and links are added in an economical manner to meet the performability demand of the network. The performability is defined as the expectation of mean packet delays in various network states caused by link failures.

The design problem of the networks can be formulated as a combinatorial optimization problem. Usually it belongs to NP-hard problems. During the past few decades, various solving methods have been developed. Pierre et al.(1995) solved network design problem minimizing cost constrained mean packet delay and connectivity using simulated annealing. Pierre and Elgibaoui(1997) solved the same type of the problem as in (Pierre et al., 1995) using tabu search. Deeter and Smith(1998) solved network design problem considering link options with different costs and reliabilities using Genetic Algorithm(GA). Kumar et al.(1995) used a GA to expand networks in view of respectively diameter, distance, and reliability constraints. They only added new nodes to existing networks.

We consider network expansion problem for adding new nodes and enhancing the existing link capacities. The network expansion design problem is formulated as the mini-

* 이 논문은 2002년도 부경대학교 연구년 교수지원에 의하여 연구되었음

mizing cost subject to the existing network design and performability. A Hybrid Genetic Approach (HGA) consisting of binary search scheme and k-nary local search scheme is suggested to solve the network expansion problem in this paper.

2. Description of the Backbone Network Expansion Problem

2.1 Assumption and notation

The following assumption is used to describe the network expansion design problem in the paper :

- Links are in either operational or failed (non-operational) state.
- One link is only failed in unit time and the failed link is not repaired.
- Link failures are independent.
- The traffic between each node-pair has a Poisson distribution.
- The packet size has an exponential distribution with mean $1/\lambda$ bits/packet.
- The nodal memory is infinite.
- The interarrival times are i.i.d.
- The transmissition times on each link are i.i.d.

The following notation is also used throughout the rest of the paper :

Notation for network expansion design

E a set of connected links of network design.

$\{i,j\}$ undirected link connecting node i and j ($\{i,j\} \equiv \{i,j\}$).

$x_{ij}(\in \{0, 1, 2, \ldots, k\text{-}1\})$ decision variable representing link type between node i and j,

$\boldsymbol{x}$ ($= \{x_{1,2}, x_{1,3}, \ldots, x_{n-1,n}\}$) architecture of network design.

n,m the number of [nodes, links] of network design.

$a_{ij},t_{ij},p_{ij},q_{ij},d_{ij},c_{ij}$ [capacity, traffic flow, reliability, unreliability, distance, cost] of $\{i,j\}$, where $p_{ij} + q_{ij} = 1$.

γ total traffic on network design.

$D(\boldsymbol{x})$ mean packet delay of network design (msec).

$F(\boldsymbol{x})$ performability of network design (msec).

F_{max} maximum allowable performability (msec).

$C(\boldsymbol{x})$ total cost of network design.

Notation for GA

g GA generation.

g_{max} GA maximum number of generation.

s GA population size per generation.

$m\%$ GA percentage of mutants created at each generation.

r_m GA mutation rate.

2.2 Model formulation

The network expansion design problem can be formulated as follows :

$$\text{Minimize } C(\boldsymbol{x}), \quad \cdots\cdots (1)$$

$$s.t. \text{ existing network design,}$$
$$F(\boldsymbol{x}) \le F_{max}$$

Total cost $C(\boldsymbol{x})$ means the total cost for installing the links of network design $\boldsymbol{x}$. Cost of $\{i,j\}$, c_{ij} is modeled as : $c_{ij} = \alpha_{ij} \cdot d_{ij} + \beta_{ij}$, where α_{ij} and d_{ij} respectively represent variable cost as a lease cost per unit distance and distance for communication link $\{i,j\}$, and β_{ij} represents fixed cost as a lease cost for a modem, interface or other piece of equipment used to connect $\{i,j\}$ to its end nodes. α_{ij} and β_{ij} depend on link level of $\{i,j\}$(Pierre and Elgibaoui, 1997). If $\{i,j\}$ is existing link and it is replaced by the higher capacity link; for example, a copper cable can be replaced by fiber-optic cable, the cost of $\{i,j\}$ is calculated as $c_{ij} = (\alpha_{ij} \cdot d_{ij} + \beta_{ij}) \cdot (1 + r_e)$, where r_e is extra cost rate due to replacing the existing link.

The objective value above is modified for considering infeasible network designs that do not meet the maximum allowable performability requirement in optimization process(Coit and Smith, 1996). The penalized total cost, $C_p(x)$, is given by,

$$C_p(\boldsymbol{x}) = C(\boldsymbol{x}) \cdot \left(\frac{F(\boldsymbol{x})}{F_{max}}\right)^{\partial} \quad \cdots\cdots (2)$$

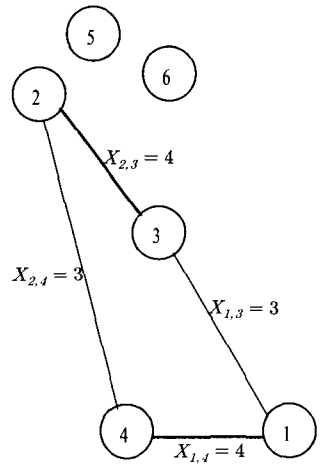
where $\partial = 0$, if $F(\boldsymbol{x}) \le F_{max}$,
$\partial = 1$, if $F(\boldsymbol{x}) > F_{max}$,
and $C(\boldsymbol{x})$ is the unpenalized total cost.

2.3 Description of expanded network design as chromosome

GA requires encoding the problem. Network design is easily formed into a k-nary string which can be used as a chromosome for the GA. Each element of the chromosome is called an allele. There are $n(n-1)/2$ alleles in each chromosome architecture $\boldsymbol{x}$ because each allele of the chromosome represents a possible link in the network design problem(Deeter and Smith, 1998). Therefore, the chromosome architecture $\boldsymbol{x}$ can represent $k^{n(n-1)/2}$ possible network designs because each allele assumes a value of among {0, 1, 2,···, k-1}.

An existing network can be expanded by adding new nodes. Figure 1 shows an example of existing network design with 4 existing nodes and 4 existing links ($x_{1,3}$ = 3, $x_{1,4}$ = 4, $x_{2,3}$ = 4, $x_{2,4}$ = 3), and 2 new nodes to be built. It is encoded as the following chromosome :

chromosome : {0[3][4]00[4][3]00000000}.



(a) Existing network architecture and new nodes

(b) Connectivity matrix

	1	2	3	4	5	6
1	-	0	3	4	0	0
2		-	4	3	0	0
3			-	0	0	0
4				-	0	0
5					-	0
6						-

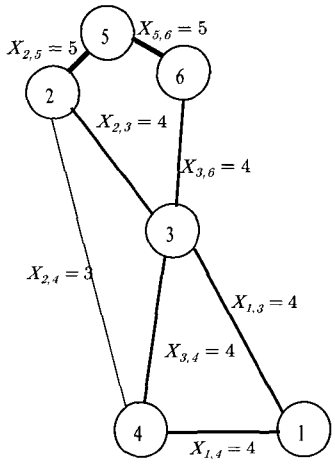
Note) Shadow area is for the existing network design.

<Figure 1> Example of an existing network design added new nodes to be built

The chromosome above is formed by copying and concatenating each row of the upper triangular of the connectivity matrix in Figure 1(b). The allele corresponding with the existing link is called as the constrained allele. It is signaled by covering with rectangular in this paper. Figure 2 shows a candidate expanded network design with 6 nodes and 8 links, given the existing network design above. The expanded network design is encoded as the following chromosome.

chromosome : {0[4][4]00[4][3]50404005}.

Given an existing network design, the chromosome architecture $\boldsymbol{x}$ for the network expansion design can represent $k^{\{n(n-1)/2-m_0\}}\prod_{\{i,j\}\in E_0}(k-e_{ij})$ possible solutions because the constrained allele of the chromosome has equal or greater level than e_{ij}, where E_0, m_0, and e_{ij} respectively mean a set of existing links, the number of existing links, and the level of existing link $\{i,j\}$. Therefore if the existing network design above with 4 existing links ($x_{1,3}$ = 3, $x_{1,4}$ = 4, $x_{2,3}$ = 4, $x_{2,4}$ = 3) is given and k is the 6, the chromosome architecture $\boldsymbol{x}$ for the network expansion design can present $6^{\{6(6-1)/2-4\}}\times(3\times2\times2\times3)=1.30\times10^{10}$ possible solutions.



(a) Expanded network design

(b) Connectivity matrix

	1	2	3	4	5	6
1	-	0	4	4	0	0
2		-	4	3	5	0
3			-	4	0	4
4				-	0	0
5					-	5
6						-

Note) Shadow area is for the existing network design.

<Figure 2> Example of expanded network design

3. Network Expansion Design using Hybrid Genetic Approach

3.1 Procedure for economic network expansion design

Below is the procedure of expansion design, followed by a more detailed description of the key steps.

(1) Initialize parameters
  1) GA parameter set : $\{g,\ g_{max},\ s,\ m\%,\ r_m\}$
  2) Network design problem parameter set : $\{n,\ m,\ \boldsymbol{x},\ D(\boldsymbol{x}),\ F(\boldsymbol{x}),\ F_{max},\ C(\boldsymbol{x})\}$

(2) Generate initial population, $g = 1$
  1) Randomly generate initial (binary based) population
  2) Send initial (binary based) population to the realization function for generating initial (k-nary based) population
  3) Send initial (k-nary based) population to the performability calculation function
  4) Send initial (k-nary based) population to the local search function for finding the best local solutions
  5) Send the best local solutions to the cost calculation function
    – infeasible solutions are penalized
  6) Check for initial Best Solution
    – if no solution is feasible the best infeasible solution is recorded

(3) Begin generational loop
  1) Copy Best Solution in new population
  2) Crossover and mutation
    – two distinct (k-nary based) parents are chosen from current population using a rank based procedure
    – the chosen (k-nary based) parents are converted into (binary based) parents
    – children are generated from (binary based) parents using uniform crossover
    – after a child is created it is mutated
    – when enough children are created the parents are replaced by the children
  3) Send the children to the realization function for generating (k-nary based) children
  4) Send (k-nary based) children to the performability calculation function
  5) Send (k-nary based) children to the local search function for finding the best local solutions
  6) Send the best local solutions to the cost calculation function
    · infeasible solutions are penalized
  7) Check for New Best Solution
    – if no solution is feasible the best infeasible solution is recorded
  8) Repeat until $g = g_{max}$

3.2 Generation of initial (binary based) population

S (binary based) parents are randomly generated for initial population. If an allele of the parent is constrained allele, it should have one value. Otherwise it should have zero or one value with equal probability. In our example, a possible (binary based) parent is {0[1][1]01[1][1]10101001}. The s (binary based) parents is sent to the realization function for generating the (k-nary based) parents. The above (binary based) parent, {0[1][1]01[1][1]10101001} can be transformed into the (k-nary based) parent, {0[4][5]03[4][4]30304004}.

3.3 Crossover and mutation

Two distinct (k-nary based) parents from current population are chosen using a rank based quadratic procedure(Tate and Smith, 1995) and the chosen (k-nary based) parents are converted into the (binary based) parents by changing positive leveled alleles to one valued alleles. For example, a parent {034044300403005} is converted into {011011100101001}. The (binary based) parents are crossovered using uniform crossover(Goldberg, 1989). This is accomplished by randomly taking an allele from one of the parents to form the corresponding allele of the child. This is only done for each non-constrained allele of the chromosome. For example, suppose parents $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are chosen to crossover.

$\boldsymbol{x}_1$ {0[1][1]01[1][1]00101001},
$\boldsymbol{x}_2$ {0[1][1]01[1][1]11000101},
child {0[1][1]01[1][1]10101001}.

After a new population is created it goes through a mutation process. A solution undergoes mutation according to the percentage of population mutated. For example if $m\%$

= 20% and s = 25, then five members are randomly chosen and mutated. Once a parent is chosen to be mutated then mutation is performed only for each non-constrained allele. The probability of mutation for each non-constrained allele that is equal to the mutation rate, r_m. Hence if r_m = 0.2 then each non-constrained allele of the parents will be mutated with a 0.2 probability. When a non-constrained allele is mutated, its value must change. If a non-constrained allele had one value, then it will be turned to zero value. Otherwise it will be turned to one value. An example is seen below. The solution has been mutated by changing the fifth allele from a 1 to a 0 and the thirteenth allele from a 0 to a 1.

solution {0[1][1]01[1][1]10101001},
mutated solution {0[1][1]00[1][1]01101101}.

The mutated solutions are sent to the realization function for generating the (k-nary based) solutions.

3.4 Performability calculation function

Network design $\boldsymbol{x}$ can have multiple states due to link failures. In this paper, we consider network states with only one or less link failure. Practically link failures are so rare that it is a very low probability to observe another failure while repairing a link. Therefore the network design $\boldsymbol{x}$ with z links can have $z+1$ possible network states, $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_z\}$, where $\boldsymbol{x}_0$ presents network state with no link failure and $\boldsymbol{x}_k$ presents network state with the k^{th} link failure, where $1 \leq k \leq z$.

For calculating traffic flow of the network state, Dijkstra's shortest path algorithm(Konak and Smith, 1999; Ahuja et al., 1993) is used. For measuring the performance of network state $\boldsymbol{x}_k$, the following mean packet delay, $D(\boldsymbol{x}_k)$ is used.

$$D(\boldsymbol{x}_k) = \frac{1}{\gamma} \sum_{\{i,j\} \in E} t_{ij}/(a_{ij} - t_{ij}), \quad \cdots\cdots (3)$$

where $0 \leq k \leq z$.

$D(\boldsymbol{x}_k)$ does not include the delay due to propagation and nodal processing.

The occurrence probabilities of network state $\boldsymbol{x}_k$ and $\boldsymbol{x}_0$ are given as

$$p(\boldsymbol{x}_k) = \prod_{\{i,j\} \in E} p_{ij} \,/\, (1 - \mathrm{qk}) \cdot \mathrm{qk} \quad \cdots\cdots (4)$$

where $1 \leq k \leq z$, and q_k means unreliability of the k^{th} link,

$$p(\boldsymbol{x}_0) = 1 - \sum_{k=1}^{z} p(\boldsymbol{x}_k) \quad \cdots\cdots (5)$$

Therefore performability of the network design $\boldsymbol{x}$ is calculated as follows :

$$F(\boldsymbol{x}) = \mathrm{E}[D(\boldsymbol{x}) \mid \text{given that only one or less link fails}]. \quad \cdots\cdots (6)$$

The performability was proposed by Li and Silvester (1984) and mentioned in Konak (2001) as the measure of traffic efficiency.

3.5 Realization function for capacity assignment

The realization function transforms a (binary based) network design into a (k-nary based) network design

Step 0. <Initialization> all a_{ij} = 0.

Step 1. <Assign link capacities of each (binary based) network state>
(1) Make $z+1$ network states of the (binary based) network design.
(2) For link $\{i,j\}$ of each network state,
– Identify traffic flow (t_{ij}).
– Assign link capacity (a_{ij}) that is just greater than the traffic flow (t_{ij}).

Step 2. <Assign link capacities of the (k-nary based) network design>
For capacity of link $\{i,j\}$ of the (k-nary based) network design, assign maximum value among z+1 link capacities (a_{ij}) obtained in Step 1(2) and capacity of link $\{i,j\}$ of the existing network design.

3.6 Local search function

Local search function is to efficiently find the best local network design with allowable performability ($F(x) \leq F_{max}$). The search is performed within the confines of link upgraded network designs.

Step 0. <Initialization> all u_{ij} = 0.

Step 1. <Check best local solution>

- If the performability of the network design is equal or less than $F_{\max}$, then the network design is the best local solution.
- Else then go to Step 2.

Step 2. <Check feasibility of the maximum upgraded network design>

When links of the network design are upgraded to maximum link type k-1,

- If the performability is greater than F_{max}, the best local solution cannot be found because the maximum upgraded network design is infeasible.
- Else then go to Step 3 with non-upgraded network design

Step 3. <Search best local solution>

(1) For each link $\{i,j\}$,

(a) Identify u_{ij}, upper link type of $\{i,j\}$, satisfying that the performability is equal or just less than F_{max} where $u_{ij} \leq k$-1. If the performability is not equal or less than F_{max} when the $\{i,j\}$ is k-1, u_{ij} is k-1.

(b) If u_{ij} - x_{ij} > 0, calculate upgrading cost, decreasing performability, and decreasing performability per unit cost in upgrading one level of link $\{i,j\}$.

(2) Select a link $\{p,q\}$ maximizing the decreasing performability per unit cost.

(3) When the network design is upgraded by one level of link $\{p,q\}$,

- If the performability is greater than F_{max}, then generate the upgraded network design and go to (1)-(b).
- Else then go to (4).

(4) Select a link $\{r,s\}$ minimizing the upgrading cost to achieve allowable performability.

(5) Upgrade one level of link $\{r,s\}$. The upgraded network design is the best local solution.

4. Illustrative Example

To explain the proposed approach, we consider a set of 6 nodes defined by the Cartesian coordinates in Table 1. The node coordinates give an Euclidean representation of the network and are used to determine the distance between each pair of nodes or the length of each link. We use the first 4 nodes as the existing nodes and the next 2 nodes as new nodes. Table 2 gives the capacity and reliability options, and Table 3 gives the traffic matrix. 25 msec for the performability requirement, 1000 bits for the average packet size ($1/\lambda$), 0.2 for the penalty rate (r_e), and 2-node-connectivity are used. Figure 1 shows the existing network architecture and new nodes, and connectivity matrix for this example. The PC with Pentium III (1GHz) and 128 MB RAM was used to test.

Since the example has 6 link levels its search space size is $6^{\{6(6-1)/2-4\}} \times (3 \times 2 \times 2 \times 3) = 1.30 \times 10^{10}$. After exploratory runs of the HGA, the following were set : s = 25, $m\%$ = 20%, r_m = 0.20 and g_{max}= 1000.

<Table 1> Node coordinates

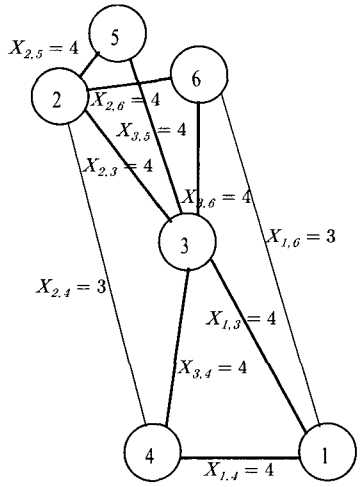
Node	1	2	3	4	5	6
Abscissa	63	22	41	33	32	40
Ordinate	8	72	45	10	84	78

<Table 2> Capacity/reliability options and costs

Link Type	Capacity (KB/sec)	Reliability	Fixed cost ($/month)	Variable cost ($/month/km)
0	0.00	0.00	0.00	0.00
1	9.60	0.70	650.00	0.40
2	19.20	0.75	850.00	2.50
3	50.00	0.80	850.00	7.50
4	100.00	0.85	1700.00	10.00
5	230.40	0.90	2350.00	30.00

<Table 3> Traffic matrix

	1	2	3	4	5	6
1	-	4	7	8	6	9
2		-	9	3	2	8
3			-	7	10	10
4				-	9	6
5					-	8
6						-



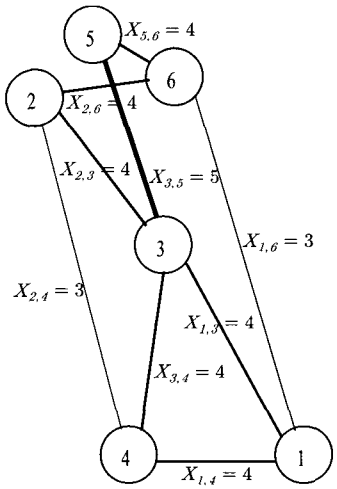
<Figure 3> The first searched network design

<Table 4> Link attributes of the first searched network design

Link *(i, j)*	Link type	Capacity *(KB/sec)*	Reliability	Traffic flow *(KB/sec)*
(1,3)	4 (3)	100.00	0.85	34
(1,4)	4 (4)	100.00	0.85	16
(1,6)	3	50.00	0.80	18
(2,3)	4 (4)	100.00	0.85	26
(2,4)	3 (3)	50.00	0.80	6
(2,5)	4	100.00	0.85	20
(2,6)	4	100.00	0.85	32
(3,4)	4	100.00	0.85	44
(3,5)	4	100.00	0.85	50
(3,6)	4	100.00	0.85	32

Note) Shadow area is for the existing links. The figure in parentheses of second column represents the existing link type.

The first searched solution {044034344444000} has cost 19056.10 $/month and performability 24.5 msec. The first searched network design and its attributes are respectively shown in Figure 3 and Table 4. The best solution {044034304450004} was found in 5.15 seconds. The best solution has cost 18420.23 $/month and performability 24.1 msec. The best solution was searched with a very small fraction of the possible search space, the solutions searched / search space size = 1294 / (1.30×10^{10}) = 0.00000995%. The HGA performed with 0.497 local search ratio (local solutions searched / solutions searched = 644 / 1294). The best network design and its attributes are respectively shown in Figure 4 and Table 5. The best network design was obtained by upgrading or adding the following new links on the existing network design; one upgraded link between the existing nodes; $x_{1,3}$ = 4, one new link between the existing nodes; $x_{3,4}$ = 4, and four new links for the new nodes; $x_{1,6}$ = 3, $x_{2,6}$ = 4, $x_{3,5}$ = 5, $x_{5,6}$ = 4, $x_{3,4}$ = 4.



<Figure 4> The best network design

<Table 5> Link attributes of the best network design

Link *(i, j)*	Link type	Capacity *(KB/sec)*	Reliability	Traffic flow *(KB/sec)*
(1,3)	4 (3)	100.00	0.85	34
(1,4)	4 (4)	100.00	0.85	16
(1,6)	3	50.00	0.80	18
(2,3)	4 (4)	100.00	0.85	26
(2,4)	3 (3)	50.00	0.80	18
(2,6)	4	100.00	0.85	32
(3,4)	4	100.00	0.85	32
(3,5)	5	230.40	0.90	70
(5,6)	4	100.00	0.85	40

Note) Shadow area is for the existing links. The figure in parentheses of second column represents the existing link type.

5. Conclusions

This paper has presented a Hybrid Genetic Approach to solve backbone network expansion design problem with minimum cost subject to the existing network design and performability. For reflecting actual network expansion, the problem was considered in view of adding new nodes to the existing network and enhancing the existing link

capacities. The HGA is consisted of binary search scheme and k-nary local search scheme. The binary search scheme is to operate crossover and mutation regardless of link options. The k-nary local search scheme is to efficiently search the best local solution. An illustrative example was shown to explain the approach. Future studies are suggested to apply the approach to the real world network expansion problems.

References

[1] Coit, D. W. and Smith, A. E.; "Penalty Guided Genetic Search for Reliability Design Optimization", Computers and Industrial Engineering, 30(4) : 895-904, 1996.

[2] Deeter, D. L. and Smith, A. E.; "Economic Design of Reliable Networks", IIE Transactions, 30 : 1161-1174, 1998.

[3] Goldberg, M. E.; Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[4] Konak, A.; "A Multiobjective Genetic Algorithm Approach to the Telecommunication Network Design Problems Considering Reliability and Performance", Doctoral Dissertation, University Pittsburgh, U.S.A., 2001.

[5] Konak, A. and Smith, A. E.; "A Hybrid Genetic Approach for Backbone Design of Communication Networks", Proceedings of the 1999 Congress on Evolutionary Computation, Washington D.C., IEEE, pp. 1817-1823, 1999.

[6] Kumar, A. R., Pathak, M., and Gupta, Y. P.; "Genetic-Algorithm-based Reliability Optimization for Computer Network Expansion", IEEE Transaction on Reliability, 44(1) : 63-70, 1995.

[7] Li, V. O. K. and Silvester, J. A.; "Performance Analysis of Networks with Unreliable Components", IEEE Transaction on Communications, 32(10) : 1105-1110, 1984.

[8] Pierre, S. and Elgibaoui, A.; "A Tabu-Search Approach for Designing Computer-Network Topologies with Unreliable Components", IEEE Transactions on Reliability, 46(3) : 350-359, 1997.

[9] Pierre, S., Hyppolite, M., Bourjolly J., and Dioume, O.; "Topological Design of Computer Communication Networks Using Simulated Annealing", Engineering Applications of Artificial Intelligence, 8(1) : 61-69, 1995.

[10] Tate, D. M. and Smith, A. E.; "A Genetic Approach to the Quadratic Assignment Problem", Computers and Operations Research, 22 : 73-83, 1995.