Designing Resilient Networks Using a Hybrid Genetic Algorithm Approach

Abdullah Konak
Information Sciences and Technology
Penn State Berks
Reading PA, 19609-6009, USA
konak@psu.edu

Alice E. Smith
Department of Industrial and Systems Engineering
Auburn University
Auburn University, AL 36849-5346, USA
aesmith@eng.auburn.edu

ABSTRACT

As high-speed networks have proliferated across the globe, their topologies have become sparser due to the increased capacity of communication media and cost considerations. Reliability has been a traditional goal within network design optimization of sparse networks. This paper proposes a genetic approach that uses network resilience as a design criterion in order to ensure the integrity of network services in the event of component failures. Network resilience measures have been previously overlooked as a network design objective in an optimization framework because of their computational complexity – requiring estimation by simulation. This paper analyzes the effect of noise in the simulation estimator used to evaluate network resilience on the performance of the proposed optimization approach.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design- *Network Topology.*

General Terms

Algorithms, Design, Reliability, Experimentation.

Key Words

Resilience, Network Reliability, Network Survivability, Genetic Algorithms, Simulation Optimization.

1. INTRODUCTION

This paper addresses the survivability and resiliency of telecommunication networks. Most modern high capacity telecommunications networks have relatively sparse topologies when compared to the previous generation of copper-based networks [3]. The lack of redundancy or multiple paths in a network makes it vulnerable to component failures. Cost-effective approaches to increase both the survivability and reliability of networks against component failures have been given significant attention. The research in this area has been evolved in two directions: the design of survivable networks and the design of reliable networks.

In telecommunication networks, survivability is achieved by reserving extra paths between nodes so that if a path is completely lost due to a link or a node failure, the traffic is restored by routing through alternative paths. A network is said to be *k-link connected* if at least *k* links must be deleted in order to disconnect any two nodes of the network. Similarly, a network is *k-node connected* if at least *k* nodes must be removed in order to disconnect the remaining nodes. To achieve a *k*-node or *k*-link connected network topology, there must exist at least *k* disjoint paths between every node pair.

The definition of network reliability depends on assumptions about both individual component failures and the services expected from a network. Given individual component reliabilities, a network's reliability expresses its ability to provide a desired service to end-users in terms of a probability. To provide connectivity is the most important service of a network. Therefore, most of reliable network design papers use traditional connectivity-based reliability measures such as all-terminal reliability (the probability that all nodes are connected) and two-terminal reliability (the probability that two specified nodes are connected) [1, 5-8, 11-13, 16].

Connectivity-based reliability measures assume that a network is not functional if the desired connectivity is lost. In practice, however, a network continues to serve the remaining connected components even though one or more nodes have become disconnected. To evaluate the ability of a network to cope with catastrophic failures and recover network services while in a disconnected state, several network resilience measures have been proposed such as the probability that all operative node pairs can communicate, the expected number of node pairs communicating, the expected number of operative nodes communicating [2,4, 9,10,15,17]. However, unlike connectivity-based reliability measures, network resilience measures have not been previously used in the literature as network design criteria.

This paper proposes using a network resilience measure instead of a connectivity-based reliability measure for network topology evaluation. The problem studied this paper is generally defined as follows:

A. Problem Parameters

- Link costs
- Node and link reliabilities

B. Optimization Objective

- Maximize a network resilience measure

C. Constraints

- 2-node connectivity
- Network design cost

2. MODELING NETWORK RESILIENCE MEASURES

2.1. Notation and Definitions

The following are the underlying assumptions:

- Links and nodes can be in either of two states, operational or failed, each with known probabilities.
- Link and node failures are independent. If a node is in the failed state, all links incident to that node are also not operational. In terms of the state space, however, these links are assumed to be in the operational state. The independence assumption is very important for computational tractability.
- No parallel link exists between nodes.
- No repair is considered in the model.

The following are the common notation and definitions used for network resilience calculations:

$G=(V,E)$ undirected network with node set V and link set E,

n, m number of nodes and links, respectively,

(i, j) component (i,j). For $i\neq j$, component (i,j) is undirected link between nodes i and j ((i,j) = (j,i), and component (i,i) denotes node i.)

$x_{(i,j)}$ state of component (i,j) such that $x_{(i,j)}=1$ if component (i,j) is in the operational state, $x_{(i,j)}=0$ otherwise,

$\mathbf{x}$ 0-1 state vector of the network, $\mathbf{x}=\{x_{(i,j)}: (i, j)\in E\cup V\}$,

S the space of all possible states,

$p_{(i,j)}$ reliability of component (i, j) ($p(i,j)= p(j,i)$ and $p(i,j)=0$ if component (i, j) does not exist in a solution),

$\Phi(\mathbf{x})$ $\Phi(\mathbf{x})=1$ all operational nodes are connected directly or indirectly in state $\mathbf{x}$; otherwise, $\Phi(\mathbf{x})=0$.

The probability of observing a particular network state $\mathbf{x}$ is given as:

$$\Pr\{\mathbf{x}\} = \prod_{(i,j)\in(E\cup V)} [1 - p_{(i,j)} + x_{(i,j)}(2p_{(i,j)} - 1)], \quad (1)$$

In this paper, we considered the following network resilience measure:

$$R = \sum_{\mathbf{x}\in S} P\{\mathbf{x}\}\Phi(\mathbf{x}) \quad (2)$$

One needs to consider 2^{m+n} network states to exactly calculate R. Therefore, an exact method is not computationally feasible.

3. OPTIMIZATION ALGORITHM

In this section, a Hybrid Genetic Algorithm (HGA) is proposed to solve the problem formulated herein. The proposed HGA has two important features: specialized crossover and local search operators that can always generate 2-node connected networks. The second feature of the HGA is that it proposes a unique approach to deal with stochastic noise in the evaluation of the objective function due to simulation while minimizing the effort to evaluate candidate solutions. Parameters and notation used in the HGA are given as follows:

C_{max} Maximum allowable design cost

X a solution

X_{best} the best feasible solution so far

x_{ij} decision variable for solution X such that $x_{ij}=1$ if a link (i,j) is selected in solution X, 0 otherwise. $X=\{x_{ij}\}$

$C(X)$ design cost of solution X

$\mu(t)$ size of the population at the beginning of generation t

$\mu_{\max}$ maximum population size

$\mu_{\min}$ minimum population size

$\theta(t)$ adaptive penalty factor to penalize infeasible solutions at generation t

$f(X,t)$ fitness of solution X at generation t

$R(X)$ estimated resilience measure for solution X

$\sigma^2_{R(X)}$ variance of estimation R(X)

P population

$g_{\max}$ Stopping criteria, maximum solutions allowed to be evaluated

U uniform random number between 0 and 1

3.1. Representation and Initial Solutions

The HGA uses a node-to-node adjacency matrix representation of solutions. In this representation, if link (i,j) is included in solution X, then $x_{i,j}=1$ and $x_{j,i}=1$; otherwise, $x_{i,j}=0$ and $x_{j,i}=0$.

The HGA procedure starts with initial solutions randomly generated by the ear decomposition procedure [14] from a complete network.

3.2. Uniform Crossover with an Efficient 2-Node Connectivity Repair Algorithm

The crossover operator has two parts: uniform crossover and a 2-node connectivity check/repair algorithm. This uniform crossover may produce offspring which are not 2-node connected or not even connected. Therefore, 2-node connectivity of the offspring needs to be checked. The procedure for the connectivity check/repair algorithm is given below.

Step 1: Use uniform crossover to generate offspring Z from parents X and Y. For $i=1,\ldots,n$ and $j=i+1,\ldots,n$, generate a random number U, set $z_{ij}=x_{ij}$ if $U<0.5$, and $z_{ij}=y_{ij}$, otherwise.

Step 2: For each node $v=1,\ldots,n$, perform the following steps.

Step 2.1: Remove node v from offspring Z.

Step 2.2: Starting from the smallest indexed node in $V \setminus v$ performed a depth-first search to determine whether all nodes in $V \setminus v$ can be visited or not. Let $S(v)$ and $\bar{S}(v)$ be the sets of the nodes that have been visited and not visited, respectively, by the depth-first search. If $\bar{S}(v) = \varnothing$, then offspring Z is 2-node connected with respect to node v and stop. Otherwise, go to Step 2.3.

Step 2.3: To repair connectivity, a minimum cost link (i,j) from parents X or Y such that $i \in S(v)$ and $j \in \bar{S}(v)$ is added to offspring Z. Note that parents X and Y must have a link satisfying this condition since they are also 2-node connected.

Step 1.4: After adding link (i,j), set $S(v) = S(v) \cap j$ and $\bar{S}(v) = \bar{S}(v) \setminus j$, and continue the depth-first search from starting node j until $\bar{S}(v) = \varnothing$.

Step 2: Return offspring Z

The connectivity check/repair algorithm above is efficient since the connectivity check and repair are performed simultaneously. If an offspring violates the connectivity requirements, the repair operation requires only few additional steps for repair.

3.3. Local Search Operators

The local search operators of the HGA are two types of network perturbation heuristics. The first type of heuristic perturbs a solution without changing the number of links. These operators are inspired by the 2-opt and 3-opt exchange heuristics used in the traveling salesperson problem as improvement heuristics [14]. Every 2-node connected network includes at least a cycle. Replacing a cycle with another cycle of the same nodes does not disturb the 2-node connectivity of a network. Therefore, a perturbed solution does not require a connectivity check. The 2-link exchange operator finds a random cycle C in the solution, removes two randomly chosen links (a, b) and (c, d) of cycle C and replaces them by links (a, c) and (b, d), neither of which was in the solution, to obtain a new cycle C'. Similar to the 2-link exchange operator, the 3-link exchange operator transforms a cycle C to cycle C' by removing three links, (a, b), (c, d), and (e, f), and adding three links (a, d), (b, e), and (c, f).

The second type of local search operators (add-a-link, remove-a-link, and 1-link exchange mutation) perturbs a solution at less degree. The add-a-link operator simply adds a non-existing link to a solution. The remove-a-link operator removes a link from a solution without violating the connectivity constraint. To achieve this, a random cycle C is found and a link (a, b) between the nodes of cycle C (excluding the ones on the cycle) is removed. In the 1-link exchange operator, link (a, b) is removed and another link (c, d) between the nodes of cycle C is added.

3.4. Evaluation of Solutions and Fitness Function

The fitness evaluation of the HGA aims to minimize computational effort to evaluate candidate solutions and to remedy the sampling error in estimating network resilience by simulation. This error can interfere with the performance of the optimization algorithm by wrongfully assessing the fitness of solutions. Although the magnitude of the sampling error can be reduced by increasing the number of simulation replications at the expense of CPU time, it cannot be totally eliminated.

3.4.1. Implementation of Simulation:

In this paper, Monte Carlo simulation is used to estimate $R(X)$. The simulation approach used herein is very similar to the one given [5]. Note that the output of structure function $\Phi(\mathbf{x})$ is a Bernoulli random variable; therefore, the variance of the estimator is given as follows:

$$\sigma^2_{R(X)} = \frac{R(X)(1-R(X))}{K}$$

where K is the number of simulation replications to obtain $R(X)$. When a new solution X is produced, first $R(X)$ is estimated by using a low number of simulation replications (K_1); and then if it is feasible with respect to the cost constraint (i.e., $C(X) \le C_{\max}$), it is compared to the best feasible solution so far (X_{best}) using the test statistics as follows:

$$z = \frac{R(X_{best}) - R(X)}{\sqrt{\sigma^2_{R(X_{best})} + \sigma^2_{R(X)}}} \tag{3}$$

If $z \le z_\alpha$, then the new solution X is said to be promising and $R(X)$ is estimated one more time by using a large number of replications (K_2). Otherwise, the fitness of solution is calculated based on the estimation by K_1 replications. After K_2 additional replications are performed, the new estimation of $R(X)$ is corrected as:

$$R(X) = \frac{K_1 \times R(X,K_1) + K_2 \times R(X,K_2)}{K_1 + K_2} \tag{4}$$

with the following variance

$$\sigma^2_{R(X)} = \frac{R(X)(1-R(X))}{K_1 + K_2} \tag{5}$$

where $R(X,K)$ is the estimated network resilience using K simulation replications. Then, the promising new solution X is compared to X_{best} one more time. If necessary, X_{best} is updated.

3.4.2. Calculation of Fitness:

After evaluating resilience for new solutions generated, the fitness of each solution in the population is calculated. Since the crossover and mutation operators always generate 2-node connected topologies, infeasibility can only be due to the cost constraint. The fitness of a solution X at generation t is calculated as follows:

$$f(X,t) = R(X) - R_{\max}\theta(t)\left(\frac{\max\{0, C(X) - C_{\max}\}}{C_{\max}}\right) \tag{6}$$

where $R_{\max}$ is the maximum resilience observed so far in the search and $\theta(t)$ is an adaptive penalty factor for infeasible solutions at generation t. $\theta(t)$ is updated before calculating fitness values as follows:

$$\theta(t) = \begin{cases} 2\theta(t-1) & r \ge \rho \\ .5\theta(t-1) & r < \rho \end{cases} \tag{7}$$

Table 1. Cartesian Coordinates for the test problems

	Nodes																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
x	26	38	93	74	86	60	26	44	54	52	57	13	32	93	7	33	54	49	99	50
y	5	86	64	8	61	10	70	70	71	36	28	68	9	56	42	52	13	3	56	27
p	.95	.95	.98	.95	.85	.95	.95	.95	.95	.9	.95	.95	.9	.85	.9	.85	.95	.95	.98	.95

where r is the percent of the infeasible solutions in the population and ρ is a infeasibility threshold parameter. At the beginning, the adaptive penalty factor is set to one ($\theta(1)=1$); and if the percent of infeasible solutions in the population is more than ρ percent, the penalty factor is increased by twofold, and otherwise, it is decreased by twofold.

3.5. Overall HGA

The important features of HGA are as follows.

- The population size is dynamic and randomly determined between $\mu_{\min}$ and $\mu_{\max}$ in each generation.
- Each solution participates to crossover with another solution which is randomly selected using the 2 tournament selection.
- Each parent solution is perturbed by applying one of the local search operators. The local search operator that will be applied to a solution is randomly and uniformly selected. For some network topologies, some of the local search operators may not be applicable (e.g., no link can be deleted if a solution consists of a single cycle of all nodes) and perturbations are not successful. In this case, the random selection process continues until a successful perturbation is achieved.
- Whenever a new solution is produced by a crossover or the local search operators, it is compared to the existing solutions in the population. If no identical solution exists, the new solution is added to the population. Otherwise, it is discarded.

The pseudo code for the overall procedure HGA is given as follows:

Step 1: Set t=1, $\theta(t)=1, \mu(t)=UINT[\mu_{\min},\mu_{\max}]$

Step 2: Generate $\mu(t)$ random solutions

Step 3: **Crossover:** For each solution $X \in P$, randomly select another Y solution using tournament selection and crossover X and Y to generate new solution Z, evaluate and add solution to P if it is not identical to a solution in P, and update X_{best} if possible.

Step 4: **Local Search:** For each solution $X \in P$, randomly and uniformly choose a local search operator, apply the operator to solution X to generate a new solution Z. Evaluate and add solution Z to P if it is not identical to a solution in P, and update update X_{best} if possible.

Step 5: Set t=t+1 and update $\theta(t)$

Step 6: Calculate the fitness of solutions and sort the population according to the fitness.

Step 7: Set $\mu(t)=UINT[\mu_{\min},\mu_{\max}]$, keep the first $\mu(t)$ solutions for crossover and remove the others from the population.

Step 8: Stop if more than $g_{\max}$ solutions are evaluated, and evaluate X_{best} with a very high number of simulation replications. Otherwise, go to step 3.

4. COMPUTATIONAL EXPERIMENTS AND DISCUSSIONS

Two test problems, a small problem with 10 nodes and a larger problem with 20 nodes, are used in the computational experiments. The Cartesian coordinates and reliabilities of the nodes for both problems are given in Table 1. The 10-node problem simply uses the first 10 nodes of the 20-node problem. The reliability of each link is 0.90. The cost of each link is equal to \$10 per unit Euclidian distance between its corresponding nodes in addition to a \$100 fixed cost. All runs were performed on PC with 2.6GHz CPU and 1.5 GB memory running on the LINUX operating system.

4.1. 10-Node Problem

The 10-node problem was primarily used to test the algorithmic features of the HGA and to investigate the effect of the simulation parameters on the performance of the algorithm. To test its algorithmic features, four different versions of the HGA were considered as follows:

A: The HGA described in the previous section,

B: The HGA with duplicate solutions allowed in the population,

C: The HGA without the local search operators 2 and 3-link exchange. However, the other mutation operators are included.

D: The HGA without crossover. This version reduces to a hill-climbing algorithm.

Each case was run with 20 random number seeds using the parameters: $C_{\max}$=\$5,000, α=0.05, $\mu_{\min}$=50, $\mu_{\max}$=75, ρ=0.50, K_1=2000, K_2=25,000, and $g_{\max}$=10,000. At the termination, the objective function of the best feasible solution was estimated by additional runs to complete 10^6 simulation replications for each case. Figure 1 shows a box-plot of results. The best results were obtained by version A. In case D where the crossover was disabled, the performance of the algorithm was the worse. This case also illustrates difficulty of using other heuristic approaches based on only local search operators to solve problems similar to one studied herein. As seen in Figure 1 the performance of the HGA without duplicate solutions in the population was better than the one with duplicate solutions. More importantly, the HGA without duplicate solutions was very robust with a low variation over replications. Not allowing same solutions in the population provided a better exploration of the search space. In addition, very expensive simulation time was not wasted to evaluate same solutions again and again.

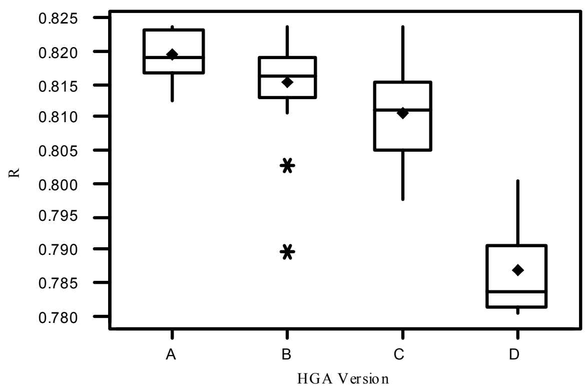


Figure 1. Box plot of the solutions for the 10-node problem.

The objective function of the problem defined in paper includes noise due to the simulation approach used to evaluate candidate solutions. This noise can interfere with the performance of the search by wrongfully assessing the fitness of solutions. Although the magnitude of the noise can be reduced by increasing the number of simulation replications at the expense of CPU time, it cannot be totally eliminated. A full factorial experimental study was carried out to investigate the effect of the simulation parameters of the HGA, α, K_1, K_2, and ρ, on final solution quality. Note that although ρ is not directly related to simulation; however, the implementation of the simulation depends on the feasibility of solutions. Therefore, the interaction of ρ with the other simulation parameters is of interest. The low and high level of the parameters (factors) used in the experiment are given as follows: α=(0.50,0.05), K_1=(1000, 8000), K_2=(25000,100000), and ρ=(0.25, 0.50). Twenty random replications were performed for each of the 16 experimental points. The results were compared using the ANOVA approach. The results showed that α is a significant parameter effecting solution quality with a p-value of 0.011. The performance of the HGA with α=0.05 was superior to the one with α=0.50. The other parameters K_1, K_2, and ρ were not significant factors with p-values of 0.625, 0.148, and 0.605. For α=0.50, the variance of the simulation estimator is not considered when a new feasible solution is compared to the best feasible solution so far, that is the comparison is based on only the means. Therefore, the probability of rejecting a new solution which is actually better than the best feasible solution so far is high. On the other hand, α=0.05 provides a cushion in this comparison, and it can be tested whether a new feasible solution is statistically worse than the best feasible solution with a 95% confidence. Therefore, the probability of rejecting a true new best feasible solution is low.

The results above do not suggest that the noise in the objective function has an effect on the solution quality. Note that parameter α used to control the effect of noise was a significant factor. Additional tests were carried out to investigate the relationship between α and other simulation parameter as follows. First, α was dropped from the ANOVA model as a factor, and an ANOVA test was run for cases with α=0.5 only. As the result of this test, K_2 turned out to be a very significant factor with a p-value of 0.005. Second, an ANOVA test was performed for cases with α=.05 only. Interestingly, as the result of this test, K_1 turned to be most influential factor with a p-value of 0.163. Third, K_2 was also dropped from the model as a factor, an ANOVA test was run for cases with α=.05 and K_2=100000. As the result of this last test, K_1 turned to be a very significant parameter with a p-value of almost zero. From these there experiments, the following intuitive arguments could be made: (*i*) When the variance of the estimation is not included in the comparisons, it is important to evaluate the best feasible solution rigorously. This reduces estimation variance, and in turn decreasing the probability of making a false assessment of the best feasible solution. (*ii*) When the probability of making a false statement about the best feasible solution is reduced by using α=.05, however, a rigorous evaluation of the best feasible solution is not as significant as a rigorous evaluation of the population.

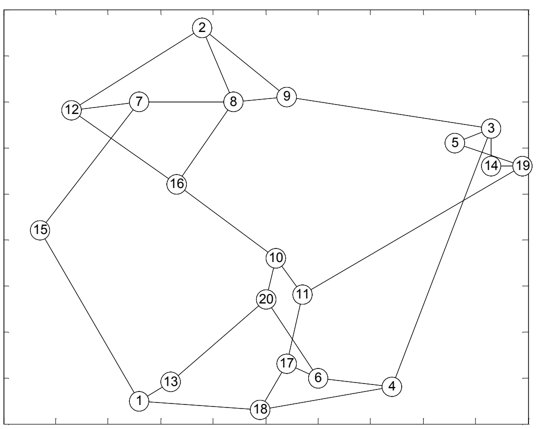
4.2. 20-Node Problem

The parameters of HGA in all runs were as follows: μ_{min}=50, μ_{max}=75, α=0.05, and g_{max}=15,000, and for simulation, K_1=2000 and K_2=50,000. Unlike network reliability measures, a network resilience measure has not been previously used in the literature as a design objective. Therefore, we could not test the proposed optimization algorithm against previous results from the literature. To measure the quality of the solutions found by the HGA, they are compared to solutions found by a hill–climbing algorithm using the local search operators of the HGA and the same penalty function. Table 2 lists a summary of the results found in 10 replications. As seen in the table, solutions found by the HGA are far superior to the ones found by the hill-climbing approach. As discussed in the case of the 10-node problem, the results for the 20-node problem showed that GA, evolutionary approaches based on recombination of solutions is very useful for the class of network problems with a constraint similar to 2-node connectivity. In column (6), the percent of solutions that were evaluated using K_2=50,000 replications are given. This number is also averaged over all of the replications. This measure is an indicator of the effectiveness of the proposed implementation of simulation in this paper. As seen in the table, only a small fraction of solutions were actually evaluated by using 50,000 simulation replications. For example when C_{max}=6,000, more than 99.6 percent of total solutions were evaluated by using only 2,000 simulation replications. This significantly reduced the CPU time. However, as networks get denser, the effectiveness of the proposed approach is also reduced.

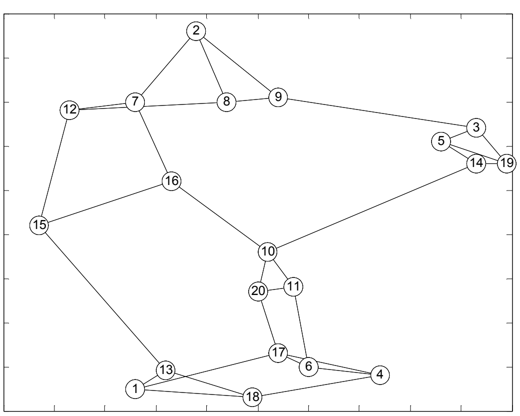
We also compared solutions found using the resilience measure used in this paper with solutions found using all-terminal reliability. For the all-terminal reliability case, the HGA used all-terminal reliability as the objective, and the rest of the algorithm and the parameters were the same. Figure 2 illustrates the best solutions found for C_{max}=9000 using network resilience and all-terminal reliability. The main difference between these two solutions is that in the former, the nodes with a low reliability such as 5, 14, and 10 have smaller numbers of incident links than in the latter. In other words, the failure of these low reliability nodes will have less effect on the connectivity of the other nodes. In the all-terminal reliability case, this is not the case. Therefore, a network resilience measure should be used if network nodes are subject to failure since the all-terminal reliability measure does not consider node failures.

Table 2. Results for the 20-node problem

	(1)	(2)	(3)	(4)	(5)	(6)
C_{max}	***R*(*X*)** The best solution in 10 rep.	$\sigma_{R(X)}$ **for the best solution**	**Range**	**Avg. CPU seconds**	**Percent of rigorous evaluations**	***R*(*X*)** The best solution of the hill-climbing
6000	0.28751	0.00045	0.07706	399	0.39	0.15114
6500	0.48567	0.00049	0.07224	415	0.58	0.26441
7000	0.58542	0.00049	0.12249	451	0.96	0.39219
7500	0.68904	0.00046	0.08301	474	1.27	0.48250
8000	0.74761	0.00043	0.05868	505	1.62	0.51218
8500	0.79641	0.00040	0.03276	552	2.21	0.60093
9000	0.83749	0.00036	0.03627	554	2.15	0.67190
9500	0.88736	0.00031	0.03787	558	2.08	0.69814
10000	0.92009	0.00027	0.03299	575	2.25	0.72817
10500	0.92707	0.00027	0.01510	610	2.66	0.74057



Cost=8997.40 Resilience=0.83744



Cost=8977.08 Reliability=0.95752

Figure 2. Solutions found for C_{max}=9000 using network resilience and all-terminal reliability as the design objectives.

5. CONCLUSIONS

This paper presented a GA based algorithm that can be used to design 2-node connected networks taking a network resilience measure into account. Although network reliability measures have been frequently used in the literature as network design criteria, network resilience measures represent a viable alternative approach. The network resilience measure used in this work is more comprehensive than traditional network reliability measures since it incorporates node failures. The proposed algorithm, the HGA, has been proven to be effective. Using specialized local search operators and a very basic adaptive penalty function, good feasible solutions were found even for very constrained problem instances. The HGA also used the variance of the estimator while comparing solutions to minimize the negative effect of noise in the objective function evaluation due to simulation.

6. REFERENCES

[1] Altiparmak, F., Dengiz, B., and Smith, A.E. Optimal design of reliable computer networks: a comparison of metaheuristics. *Journal of Heuristics,* 9(6) (2003), 471-487.

[2] Autenrieth, A., Brianza, C., Clemente, R., Demeester, P., Gryseels, M., Harada, Y., Jajszczyk, A., Janukowicz, D., Kalbe, G., Ohta, S., Ravera, M., Rhissa, A.G., Signorelli, G., and Van Doorselaere, K. Resilience in a multi-layer network. *CSELT Technical Reports*, 26(6) (1998), 869-882.

[3] Balakrishnan, A., Magnanti, T.L., and Mirchandani, P. Designing hierarchical survivable networks. *Operations Research*, 46(1) (1998), 116-136.

[4] Ball, M.O. Complexity of network reliability computations. *Networks*, 10(2) (1980), 153-165.

[5] Deeter, D.L. and Smith, A.E. Economic design of reliable networks. *IIE Transactions*, 30(12) (1998), 1161-1174.

[6] Dengiz, B., Altiparmak, F., and Smith, A.E. Efficient optimization of all-terminal reliable networks, using an

evolutionary approach. *IEEE Transactions on Reliability* 46(1) (1997), 18-26.

[7] Dengiz, B., Altiparmak, F., and Smith, A.E. Local search genetic algorithm for optimal design of reliable networks. *IEEE Transactions on Evolutionary Computation*, 1(3) (1997), 179-188.

[8] Hsieh, C.-C. and Hsieh, Y.-C., Reliability and cost optimization in distributed computing systems. *Computers & Operations Research*, 30(8) (2003), 1103-1119.

[9] Huang, H. and Copeland, J.A. A series of Hamiltonian cycle-based solutions to provide simple and scalable mesh optical network resilience. *IEEE Communications Magazine*, 40(11) (2002), 46-51.

[10] Hwang, F.K., Najjar, W., and Gaudiot, J.L. Comments on Network resilience: a measure of network fault tolerance. *IEEE Transactions on Computers*, 43(12) (1994), 1451-1453.

[11] Jan, R.-H. Design of reliable networks. *Computers and Operations Research*, 20(1) (1993), 25-34.

[12] Jan, R.-H., Hwang, F.-J., and Chen, S.-T. Topological optimization of a communication network subject to a reliability constraint. *IEEE Transactions on Reliability*, 42(1) (1993), 63-70.

[13] Konak, A. and Smith, A.E. Multiobjective optimization of survivable networks considering reliability. In *The 10th International Conference on Telecommunication Systems*. (October, 2002). Naval Postgraduate School, Monterey, CA.

[14] Monma, C.L. and Shallcross, D.F. Methods for designing communications networks with certain two-connected survivability constraints. *Operations Research*, 37(4) (1989), 531-541.

[15] Najjar, W. and Gaudiot, J.-L. Network resilience: a measure of network fault tolerance. *IEEE Transactions on Computers,* 39(2) (1990), 174-181.

[16] Soni, S., Narasimhan, S., and LeBlanc, L.J. Telecommunication access network design with reliability constraints. *IEEE Transactions on Reliability*, 53(4) (2004), 532-541.

[17] Veitch, P. and Johnson, D. ATM network resilience. *IEEE Network,* 11(5) (1997), 26-33.