Heuristic Optimization of Network Design Considering All-Terminal Reliability

Darren L. Deeter • Andersen Consulting • Chicago

Alice E. Smith • University of Pittsburgh • Pittsburgh

Key Words: Network design, Heuristic optimization, Genetic algorithms, All-terminal reliability

SUMMARY & CONCLUSIONS

This paper describes a heuristic optimization approach using genetic algorithms. The method solves general network design problems to optimality, or near-optimality, with respect to reliability. The optimization formulation in this paper relaxes the previous restrictions that appear in the literature. Network design is expanded to include links of differing reliability and to select from multiple choices for each possible link component. This significantly expands the search space, necessitating a heuristic approach, but also is much more reflective of actual communications network design problems. The approach can use either exact or approximate network reliability calculations and its flexibility, effectiveness and efficiency are demonstrated on a series of increasingly constrained all-terminal reliability test problems.

1. *INTRODUCTION*

Design of networks considering a system reliability constraint or objective has been the subject of many research efforts and has applications in telecommunications and computer networking and related domains in electric, gas and sewer networks. All-terminal network reliability (also called overall network reliability) is calculated from the probability that each and every node in the network is connected to each other. This is distinguished from the more common source-sink (s-t) network where reliability is measured from the probability that the source is connected with the sink. Design of networks considering all-terminal reliability, along with other objectives or constraints (e.g. cost, distance, volume), is an NP-hard combinatorial problem when the design is selected from a set of available components with known specifications (Ref. 1). Compounding the exponential growth (NP) in the search space with the size of the network and available components is that reliability of all-terminal networks is computationally costly to calculate. These two aspects combine to make the optimization of network design when considering all-terminal reliability a problem which is extremely difficult to solve, even for small networks.

This paper presents a heuristic approach to design of networks when considering all-terminal reliability formulated as minimizing cost given a reliability constraint. It is assumed that all nodes are fixed and perfectly reliable. Each pair of nodes can be connected by a single link (arc), which is bi-directional and has a stated reliability and cost. Links fail *s*-independently, and the system fails if any node is not connected, through some path, to any other (i.e., the network does not form at least a spanning tree). Previous research has assumed that reliability of all links in the network are identical while cost depends on which two nodes are connected. Only one reliability/cost option is available for each pair of nodes. The approach in this paper significantly expands the previous research by allowing links to be chosen from different components with different costs and reliabilities. This choice of link components, while more accurately reflecting real design decisions, greatly expands the search space, thus requiring a heuristic rather than an enumerative approach.

A genetic algorithm heuristic is applied with a constant-sized population of candidate network designs maintained throughout the search. These designs are selected for recombination to form new candidate designs with preference for selection given to those networks with better objective function value (lower cost or higher reliability). Designs are also slightly perturbed to expand the search to new areas, thus avoiding local optima. These are the genetic algorithm functions of selection, crossover and mutation. The search continues for a number of generations, where the population size times the number of generations is the number of (non-unique) solutions searched. The search terminates when a pre-defined upper bound on the number of generations is reached. Genetic search is heuristic and does not guarantee optimality. However, previous applications in combinatorial optimization (including reliability design) have shown that genetic algorithms are robust and generally yield near-optimal solutions.

The heuristic approach to network design is demonstrated on multiple test problems with varying degree of constraint. It is also shown that the approach is very easily translated to source-sink networks and to networks with special

architecture restrictions (i.e., not all possible links are allowed). The genetic algorithm approach offers major advantages over previous approaches to the design of reliable networks in terms of flexibility, performance, computational expediency, and ease of coding and use.

2. NOTATION

L	Set of possible links.
l_{ij}	Option of each link. $l_{ij} \in \{0,1,2,...,k\text{-}1\}$
N	Set of given nodes.
n	A node.
$p(l_k)$	Reliability of link option.
$c(l_k)$	Unit cost of link option.
x	Architecture of network design.
C(**x**)	Total cost of network design.
R(**x**)	Reliability of network design.
R_o	Minimum network reliability constraint.
s	Population size.
g	Current generation number.

3. STATEMENT OF THE PROBLEM AND PREVIOUS APPROACHES

In the all-terminal reliability network design problem, there are a set of N nodes with specified topology, which will be interpreted as Euclidean distance between coordinates on a plane. Note that this distance does not have to be an actual distance but that it represents some cost to connect two nodes regardless of type of connection. The nodes are considered fully reliable and assumed not to fail under any circumstances. There is a set of L links which connect the nodes in N. In this problem it is assumed that every link possible is included in L, i.e., a fully connected network. So, for any (n_i,n_j) pair of elements of N there exists the possibility of l_{ij} element of L such that l_{ij} connects n_i and n_j. Note that it is assumed that a link is bi-directional. Thus, if l_{ij} is turned on it lets n_i communicate with n_j and vice versa. Also it is assumed that there can be only one link per location (no redundancy). All links fail *s*-independently and no repair is considered. The total number of links possible in a single design is:

$$|L| = \frac{\left(|N| \bullet \left(|N| - 1\right)\right)}{2} \qquad (1)$$

In the generalization of this problem it is possible for a link to have more than two states. Consequently instead of l_{ij} being either included in the network design (l_{ij}=1) or not included in the network design (l_{ij}=0), the link can have *k* different levels (l_{ij}=0,1,2,...,*k*-1). The first level is always level 0 (l_{ij}=0) where the link has reliability $p(l_{ij}=0) = 0$ and unit cost $c(l_{ij}=0) = 0$. This is equivalent to not including the link in the design. The second level (l_{ij}=1) is where the link is included in the design with a reliability of $p(l_{ij}=1)$ and a unit cost of $c(l_{ij}=1)$ and so on. Note that the number of levels of connection *k* could easily be changed and that *k* does not have to be uniform for each *i*,*j* pair. Thus a candidate solution, **x**, to this problem will consist of a selected number of links $l_{ij}=k$ where *k* is the level at which they connect nodes n_i and n_j. Consequently any problem can be defined as $G(N,L,R(\mathbf{x}),C(\mathbf{x}),R_o)$. The mathematical formulation for this problem when minimizing cost subject to a minimum network reliability constraint[1] is:

$$\text{Min: } C(\mathbf{x})$$
$$\text{s.t. } R(\mathbf{x}) \geq R_o$$

The cost of a specific architecture **x** is given by C(**x**) and the reliability of **x** is given by R(**x**). The problem is then for a given $G(N,L,R(),C(),R_0)$ to find an **x** so that $R(\mathbf{x}) \geq R_o$, and $C(\mathbf{x}) \leq C(\mathbf{x}')$ where **x′** is any other network architecture in $G(N,L,p,c,R_o)$.

Network design considering reliability has been the subject of continuing research. Given that it is NP-hard, approaches have either been enumerative-based which are applicable only for small network sizes (Refs. 2-4), or heuristic-based which can be applied to larger networks but do not guarantee optimality (Refs. 5-13). These previous efforts have yielded effective methods to design reliable networks. However, they are characterized by the simplistic (and restrictive) assumption that all links have identical reliability. This is rarely the case in real problems. Furthermore, the previous approaches assume only two possible choices for each link. Either the link is included in the network (l_{ij}=1) or it is not included in the network (l_{ij}=0). Real design problems normally include choices of differing link reliabilities and costs. For example, in telecommunications, a cable can be sheathed, or not. To enable the flexibility needed for this relaxed formulation, a meta-heuristic genetic algorithm (GA) has been selected. There have been a few uses of genetic algorithms for reliability design (Refs. 6, 10-12, 14-16) with very encouraging results.

Another active area of research related to the network design problem is the calculation or estimation of network reliability. There are two main approaches - exact calculation through analytic methods (Refs. 17-21) and estimation through variations of Monte Carlo simulation (Refs. 22-29). For the all-terminal network reliability problem, efficient simulation is especially difficult because these methods generally lose efficiency as a network approaches a fully connected state. There are also upper and lower bound expressions for network reliability (Refs. 30-31). However, these are too loose to be effective surrogates in the all-terminal design process. Furthermore, many bounding procedures and improved efficiency simulations depend on

[1] The problem is usually formulated to maximize reliability given an upper bound on cost. However, the authors believe the formulation used in this paper is more reflective of actual network design scenarios.

the assumption that all links have the same reliability. This assumption is relaxed in the research presented here.

4. DESCRIPTION OF SOLUTION APPROACH

Genetic algorithms are a heuristic search technique that tend to give good, if not optimal, solutions to hard problems (Ref. 32). A genetic algorithm lends itself to this problem because each network design **x** is easily formed into an integer vector which can be used as a chromosome for the genetic algorithm. Each element of the chromosome represents a possible link in the network design problem so there are N×(N-1) / 2 vector components in each candidate architecture **x**. The value of each of these elements tells what type of connection the specific link has with the pair of nodes it connects. Consider an example with five nodes and k=4 levels of connection shown in Figure 1 and Table 1. Note there are (5*4)/2=10 possible links for this example but only five are included; the other five are at level of connection k=0. Upon examination of the link matrix for Figure 1 it is seen that the matrix is symmetric. Thus all the information needed to record this particular architecture lies in the upper triangular half of the matrix. This information is placed in a chromosome (vector) of length (5*4)/2=10 by copying and concatenating each row into the chromosome, as is seen below:

Chromosome: {0100203102}

Note that the only possible values allowed in each position of the chromosome are 0,1,...k-1. The solution space of possible network architectures is $k^{((N*(N-1))/2)}$.

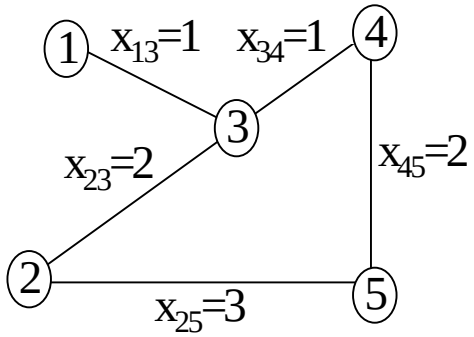

Figure 1. Sample Network Architecture.

Table 1. Connectivity Matrix.

	1	**2**	**3**	**4**	**5**
1	-	0	1	0	0
2	0	-	2	0	3
3	1	2	-	1	0
4	0	0	1	-	2
5	0	3	0	1	-

The genetic algorithm used is described below. It contains the usual components of a GA (selection, breeding, mutation, fitness calculation, culling).

- **Initialize Variables**
  - termination criteria
  - population size, s
  - percentage of population mutated, m
  - mutation rate, r_m
  - penalty rate, r_p
- **Generate Initial Population**
  - random
- **Check for Initial Best Solution**
  - if no solution is feasible the best infeasible solution is recorded
- **Begin Generational Loop**
  - Select and Breed Parents
    - place best solution in new population
    - two distinct parents are chosen using a rank based quadratic procedure
    - children are generated using uniform crossover
    - after a child is created it is mutated
    - when enough children are created they replace the parents
  - Send new population to the reliability calculation function
  - Send new population to the cost calculation function
    - infeasible members are penalized
  - Check for new best solution
    - if no solution is feasible the best infeasible solution is recorded
  - Repeat until termination criteria is met

Selection: Two parents are selected to breed using a rank based quadratic procedure as devised in (Ref. 33). All the parents in the population are ordered according to their level of fitness with 1 being the highest level of fitness. Next, a random number is generated between 0 and $\sqrt{s}$. Then the random number is squared, truncated and has one added to it. The resulting number is the rank of the parent chosen. This process is then repeated to pick the second parent. If the same parent is chosen, the process is repeated until a distinct second parent is chosen. For example if the GA has a population size of 20 then a random number between 0 and $\sqrt{20}$ is generated. Suppose the number chosen is 2.5, then this number is squared giving 4.25, truncated yielding 4 and increased by 1 for a final selection of 5. Thus the parent chosen is the fifth best solution.

Breeding: Breeding is done using uniform crossover (Ref. 32). This type of breeding is accomplished by selecting two parent solutions and randomly taking a component from one parent to form the corresponding component of the child. For example, suppose parent $\mathbf{x_1}$ and parent $\mathbf{x_2}$ are chosen to breed.

$\mathbf{x_1}$	{0120131011}
$\mathbf{x_2}$	{1111012002}
child	{0110132001}

In the child above, the first component of the chromosome comes from parent $\mathbf{x_2}$. The second component of the child

has to be a 1 since both parents have a 1 in this position of the chromosome. The third component of the child also comes from parent $\mathbf{x}_2$, and so on.

Mutation: Mutation is a way to add new genetic material to a population. This is done to help the GA avoid getting stuck at a local optimum. A child undergoes mutation according to the percentage of population mutated, *m*. For example, if *m*=20% and *s*=30, then six children in the new population are randomly chosen for mutation. Once a child is chosen to be mutated, then the probability of mutation for each vector component is equal to the mutation rate, r_m. If the r_m=0.3 then each component of a selected child will be mutated with a 0.3 probability. When a component is mutated its value must change. If a link was turned off, i.e. l_{ij}=0 then it will be turned on with an equal probability of being turned to any of the on states (1,2,...,*k*-1). If a link is originally on, then it will be turned off with a 0.5 probability and will be left on but at a different level with 0.5 probability. For the child in the above example, a mutation might be:

child	{0110132001}
mutated child	{0010132031}

Reliability Calculation: A backtracking algorithm is used to exactly calculate the system reliability, R(**x**), for the problems in this paper due to their computationally tractable size. The backtracking algorithm outline is given below (Ref. 17):

Step 0:(Initialization). Mark all links as free; create a stack which is initially empty.
Step 1:(Generate modified cutset)
(a)Find a set of free links that together with all inoperative links will form a network-cut.
(b) Mark all the links found in 1(a) inoperative and add them to the stack.
(c) The stack now represents a modified cutset; add its probability into a cumulative sum.
Step 2: (Backtrack)
(a) If the stack is empty, end.
(b) Take an link off the top of the stack.
(c) If the link is inoperative and if when made operative, a spanning tree of operative links exists, then mark it free and go to 2(a).
(d) If the link is inoperative and the condition tested in 2(c) does not hold, then mark it operative, put it back on the stack and go to Step 1.
(e) If the link is operative, then mark it free and go to 2(a).

Fitness: The genetic algorithm attempts to find the minimum cost network architecture which meets or exceeds a pre-specified network reliability, R_o. The GA is constructed so that it may consider infeasible network architectures which are penalized. Infeasible solutions may contain beneficial they may contain beneficial parts. Consequently, breeding two infeasible solutions or an infeasible with a feasible solution can yield a good feasible solution. Also, since there is only one constraint (reliability) that constraint will be active or nearly active for a minimum cost network. Therefore, the optimal design will lie on the boundary between feasible and infeasible designs. A complete discussion of GA penalty functions for constrained combinatorial problems can be found in Ref. 34. The fitness of a network is its cost to which a penalty is added if $R(\mathbf{x}) < R_o$:

$$C_p(\mathbf{x}) = C(\mathbf{x}) + C(\mathbf{x}^*) \times (1 + R_o - R(\mathbf{x}))^{r_p + \frac{s \times g}{50}} \qquad (2)$$

where $C_p(\mathbf{x})$ is the penalized cost, $C(\mathbf{x})$ is the unpenalized cost, $C(\mathbf{x}^*)$ is cost of the best feasible solution in the population, r_p is the penalty rate, *s* is the population size and *g* is the generation number.

5. TEST PROBLEMS AND RESULTS

A five node test problem with multiple levels was studied. The problem is based on Jan et al.'s test problem (Ref. 2) using link costs as distances and using the unit costs and reliabilities of Table 2. Note that unit cost increases greater than linearly with reliability. This problem has a search space of 1,048,576 and is considered at seven different system reliability levels. Since this is a relatively small problem, it is possible to enumerate all solutions to obtain the optimal solutions for the seven different system reliability levels. These appear in Table 3. Note that each optimal solution makes use of two or more link connection levels. This indicates that designs improve by allowing differing link reliabilities.

Table 2. Link Unit Costs and Corresponding Reliabilities.

Connection Type	Reliability	Cost/Unit Distance
0 (not connected)	0.00	0
1	0.85	4
2	0.90	16
3	0.95	48

Table 3. Optimal Solutions to Test Problems.

R_o	Cost	R(x)	Architecture (x)
0.99900	5522	0.99908	3323333323
0.99500	4352	0.99518	3113311313
0.99000	3754	0.99052	3203322303
0.95000	2634	0.95353	3003302303
0.93125	2416	0.93361	2003301303
0.90000	2184	0.91854	3003300303
0.85000	1904	0.85536	3003200203

After exploratory trial runs, effective values of the GA parameters were established: *s*=40, *m*=25%, r_m=0.25, g_{max}=6000, and r_p=6. Ten runs were then made for each system reliability using 10 randomly selected numbers as

random number seeds. This examines the variability of the GA since it is a stochastic search method.

Using R_o = 0.99900 is the most constrained, and consequently the hardest, of the seven problems. Results appear in Table 4. Eight of the ten runs found the optimal solution and the other two found a near optimal solution. Of the eight runs in which the optimum was found, six of them were found by searching 4% of the search space or less. Note that all final solutions are feasible despite the severely constrained feasible region.

Table 4. Five Node 0.999 System Reliability Results.[2]

Seeds	Gen Found	R(x)	C(x)	#Evals	Architecture (x)
1	1050	0.999	5522	42000	3323333323
2	90	0.999	5522	3600	3323333323
3	10	0.999	5530*	400	3233323333
4	80	0.999	5522	3200	3323333323
5	5210	0.999	5522	208400	3323333323
6	50	0.999	5522	2000	3323333323
7	200	0.999	5522	8000	3323333323
8	150	0.999	5522	6000	3323333323
9	60	0.999	5530*	2400	3233323333
10	3240	0.999	5522	129600	3323333323

In the second problem, R_o = .99500. Those results can be seen in Table 5. In this case the optimal solution to the design problem is found only four times. However, the same near optimal solution is found in the six other runs.

Table 5. Five Node 0.995 System Reliability Results

Seeds	Gen Found	R(x)	C(x)	#Evals	Architecture (x)
1	340	0.995	4354*	13600	3303222323
2	1740	0.995	4354*	69600	3303222323
3	340	0.995	4354*	13600	3303222323
4	120	0.995	4352	4800	3113311313
5	180	0.995	4352	7200	3113311313
6	410	0.995	4354*	16400	3303222323
7	1790	0.995	4352	71600	3113311313
8	90	0.995	4354*	3600	3303222323
9	30	0.995	4352	1200	3113311313
10	760	0.995	4354*	30400	3303222323

For the other five test problems (R_o = 0.99, 0.95, 0.93125, 0.90, 0.85) the optimal solution is found using each of the ten seeds. Overall, the optimal solution is found in fewer total evaluations as the severity of the constraint of the problem is relaxed, i.e. as R_o is lowered. As is seen in the summary in Table 6, the average portion of the solution space that must be searched in order to find the optimal solution is fairly consistent. Note that this is an upper bound as there is no accounting for duplicate solutions which are likely to occur in GA search. Considering the proportion of the possible network architectures examined (0.44% to 11.34%), that optimal solutions were obtained in most cases, and very good, feasible solutions were obtained otherwise, the GA appears to be extremely effective. Conserving the proportion of the solution space searched is a primary concern in network reliability problems where the calculation or estimation of reliability is very computationally expensive.

Table 6. Summary Results of Seven Test Problems.

R_o	Mean Solutions Searched	% of Space Searched	Number Optimal
0.99900	40560	3.87	8 of 10
0.99500	23200	2.22	4 of 10
0.99000	31400	2.99	10 of 10
0.95000	118880	11.34	10 of 10
0.93125	25560	2.44	10 of 10
0.90000	26160	2.49	10 of 10
0.85000	4640	0.44	10 of 10

6. DISCUSSION

If a user can be assured of optimal, or near-optimal, results when exerting a small fraction of the computational effort required for enumerative methods there is strong motivation to use a heuristic, such as the one developed in this paper,. There is an added attraction that since the GA is an iterative technique, which generally achieves diminishing improvements in objective function value as the search continues, the user may terminate the search at any time and still have a very good, feasible solution. Furthermore, not only a single good, feasible solution is returned. The user may examine a group of superior solutions if further investigation is warranted.

Besides effectiveness and efficiency, the GA is quite flexible. Additional research by the authors (Ref. 35) demonstrate that this meta-heuristic is applicable to other reliability metrics (source-sink), other reliability calculations (Monte Carlo simulation), larger networks (10 to 18 nodes), non-fully connected architectures, and reversal of the objective / constraint functions (maximizing reliability given a cost constraint). These encouraging demonstrations offer promise for real world problems where users would be allowed greater flexibility to critically examine a variety of superior designs and a variety of formulations.

ACKNOWLEDGMENT

The second author gratefully acknowledges financial support from the National Science Foundation CAREER grant DMI-9502134.

REFERENCES

1. M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Co., San Francisco (1979).

[2] In the tables, asterisks denote non-optimal solutions.

2. Rong-Hong Jan, Fung-Jen Hwang, and Sheng-Tzong Chen, "Topological optimization of a communication network subject to a reliability constraint," *IEEE Transactions on Reliability*, Vol. 42, No. 1 (1993) 63-70.

3. R. S. Wilkov, "Design of computer networks based on a new reliability measure," *Symposium on Computer-Communications Networks and Teletraffic*, Polytechnic Institute of Brooklyn (1972) 371-384.

4. K. K. Aggarwal, Y. C. Chopra and J. S. Bajwa, "Topological layout of links for optimising the overall reliability in a computer communication system," *Microelectronics and Reliability*, Vol. 22, No. 3 (1982) 347-351.

5. Mir M. Atiqullah, and S. S. Rao, "Reliability optimization of communication networks using simulated annealing," *Microelectronics and Reliability*, Vol. 33, No. 9 (1993) 1303-1319.

6. Berna Dengiz, Fulya Altiparmak and Alice E. Smith, "A genetic algorithm approach to optimal topological design of all terminal networks," *Intelligent Engineering Systems Through Artificial Neural Network, Vol. 5* (C. H. Dagli, M. Akay, C. L. P. Chen, B. R. Fernandez and J. Ghosh, editors) (1995) 405-410.

7. Peter C. Fetterolf and G. Anandalingam, "Optimal design of LAN-WAN internetworks: an approach using simulated annealing," *Annals of Operations Research*, Vol. 36 (1992) 275-298.

8. Fred Glover, Micheal Lee and Jennifer Ryan, "Least-cost network topology design for a new service: an application of a tabu search," *Annals of Operations Research*, Vol. 33 (1991) 351-362.

9. Seok J. Koh and Chae Y. Lee, "A tabu search for the survivable fiber optic communication network design," *Computers in Industrial Engineering*, Vol. 28, No. 4 (1995) 689-700.

10. Anup Kumar, Rakesh M. Pthak, Yash P. Gupta and Hamid R. Parsaei, "A genetic algorithm for distributed system topology design," *Computers and Industrial Engineering*, Vol. 28, No. 3 (1995) 659-670.

11. Godfrey A. Walters and David K. Smith, "Evolutionary design algorithm for optimal layout of tree networks," *Engineering Optimization* Vol. 24 (1995) 261-281.

12. Anup Kumar, Rakesh M. Pthak and Yash P. Gupta, "Genetic-algorithm-based reliability optimization for computer network expansion," *IEEE Transactions on Reliability*, Vol. 44, No. 1 (1995) 63-72.

13. Samuel Pierre, Michel-Ange Hyppolite, Jean-Marie Bourjolly and Oumar Dioume, "Topological design of computer communication networks using simulated annealing," *Engineering Applications of Artificial Intelligence*, Vol. 8, No. 1 (1995) 61-69.

14. David W. Coit and A. E. Smith, "Reliability optimization of series-parallel systems using a genetic algorithm," *IEEE Transactions on Reliability* Vol. 45, No. 2 (1996) 254-260.

15. K. Ida, M. Gen and T. Yokota, "System reliability optimization with several failure modes by genetic algorithm," *Proceedings of 16th International Conference on Computers and Industrial Engineering* (1994) 349-352.

16. Laura Painton and J. Campbell, "Genetic algorithms in optimization of system reliability," *IEEE Transactions on Reliability*, Vol. 44 (1995) 172-178.

17. Michael Ball and Richard M. Van Slyke, "Backtracking algorithms for network reliability analysis," *Annals of Discrete Mathematics*, Vol. 1 (1977) 49-64.

18. Chi Hanzhong and Li Dongkui, "A new algorithm for computing the reliability of complex networks by the cut method," *Microelectronics and Reliability*, Vol. 34, No. 1 (1994) 175-177.

19. D. Mandaltsis and J. M. Kontoleon, "Overall reliability determination of computer networks with hierarchical routing strategies," *Microelectronics and Reliability*, Vol. 27, No. 1 (1987) 129-143.

20. K. K. Aggarwal and Suresh Rai, "Reliability evaluation in computer-communication networks," *IEEE Transactions on Reliability*, Vol. R-30, No. 1 (1981) 32-35.

21. Chen Liu, Mingde Dai, Xin-Yu Wu and Wai-Kai Chen, "A network overall reliability algorithm," *Proceedings of Neural, Parallel and Scientific Computations*, Atlanta (1995) 287-292.

22. Mainak Mazumdar, David W. Coit and Fen-Ru Shih, "An efficient Monte Carlo method for assessment of system reliability based on a Markov model," University of Pittsburgh, *Department of Industrial Engineering Technical Report* (1995).

23. Hector Cancela and Mohamed El Khadiri, "A recursive variance-reduction algorithm for estimating communication-network reliability," *IEEE Transactions on Reliability*, Vol. 44, No. 4 (1995) 595-602.

24. George S. Fishman, "A Monte Carlo sampling plan for estimating network reliability," Operations Research, Vol. 34 (1986) 581-594.

25. George S. Fishman, "A comparison of four Monte Carl methods for estimating the probability of s-t connectedness," *IEEE Transactions on Reliability*, Vol. R-35, No. 2 (1986) 145-155.

26. J. Satish Kamat and Michael W. Riley, "Determination of reliability using event-based Monte Carlo simulation," *IEEE Transactions on Reliability*, Vol. R-4, No. 1 (1975) 73-75.

27. Tiromitsu Kumamoto, Kazuo Tanaka and Koichi Inoue "Efficient evaluation of system reliability by Monte Carlo method," *IEEE Transactions on Reliability*, Vol. R-26, No. 5 (1977) 311-315.

28. Min-Sung Yeh, Jsen-Shung Lin and Wei-Chang Yeh, "A New Monte Carlo method for estimating network reliability," *Proceedings of the 16th International Conference on Computers & Industrial Engineering* (1994) 723-726.

29. Malcolm C. Easton and C. K. Wong, "Sequential destruction method for Monte Carlo evaluation of system reliability," *IEEE Transactions on Reliability*, Vol. R-29, No. 1 (1980) 27-32.

30. Rong-Hong Jan, "Design of reliable networks," *Computers and Operations Research*, Vol. 20, No. 1 (1993) 25-34.

31. Shuichi Shinmori, Takeshi Koide and Hiroaki Ishii, "On lower bound for network reliability by edge-packing," *Transactions of the Japan Society for Industrial and Applied Mathematics*, Vol. 5, No. 2 (1995) 139-151.

32. David E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA. (1989).

33. David M. Tate and Alice E. Smith, "A genetic approach to the quadratic assignment problem," *Computers and Operations Research*, Vol. 22 (1995) 73-83.

34. David W. Coit, Alice E. Smith and David M. Tate, "Adaptive penalty methods for genetic optimization of constrained combinatorial problems," *INFORMS Journal on Computing*, Vol. 8, No. 2 (1996) 173-182.

35. Darren L. Deeter, *A Genetic Algorithm for Solving a Generalized Network Design Problem*, unpublished Master's Thesis, Department of Industrial Engineering, University of Pittsburgh (1996).

BIOGRAPHIES

Darren L. Deeter
Andersen Consulting Inc.
Chicago, IL

Darren Deeter was a graduate student in the Industrial Engineering Department of the University of Pittsburgh and received an M.S.I.E. in Summer of 1996. He currently works for Andersen Consulting in Chicago as an Analyst. His Master's Thesis focuses on heuristic combinatorial optimization of network design when considering reliability and cost. He has a B.S. in Mathematics and Physics from Allegheny College (Phi Beta Kappa) and an M.A. in Mathematics from Kent State University. He is a member of IIE, INFORMS and APICS.

Alice E. Smith, Ph.D., P.E.
Department of Industrial Engineering
University of Pittsburgh
1031 Benedum Hall
Pittsburgh, PA 15261 USA
Internet (email): aesmith@engrng.pitt.edu

Alice Smith is Associate Professor of Industrial Engineering. After ten years of industrial experience with Southwestern Bell Corporation, she joined the faculty of the University of Pittsburgh in 1991. Her research interests are in modeling and optimization of complex systems using computational intelligence techniques, and her research has been sponsored by Lockheed Martin Corporation, ABB Daimler-Benz Transportation, the Ben Franklin Technology Center of Western Pennsylvania, the National Institute of Standards and the National Science Foundation, from which she was awarded a CAREER grant in 1995. Her work in design optimization when considering system reliability has appeared in *IEEE Transactions on Reliability, INFORMS Journal on Computing, Computers & Operations Research* and *Computers & Industrial Engineering.* Dr. Smith is a senior member of IEEE, IIE and SWE, and a member of ASEE and INFORMS.