

A Hybrid Genetic Algorithm Approach for Backbone Design of Communication Networks

Abdullah Konak

University of Pittsburgh
Department of Industrial Engineering
1048 Benedum Hall
Pittsburgh, PA 15261
abkst8+@pitt.edu

Alice E. Smith

University of Pittsburgh
Department of Industrial Engineering
1031 Benedum Hall
Pittsburgh, PA 15261
aesmith@engrng.pitt.edu

Abstract- This paper presents a hybrid approach of a genetic algorithm (GA) and local search algorithms for the backbone design of communication networks. The backbone network design problem is defined as finding the network topology minimizing the design/operating cost of a network under performance and survivability considerations. This problem is known to be NP-hard. In the hybrid approach, the local search algorithm efficiently improves the solutions in the population by using domain-specific information while the GA recombines good solutions in order to investigate different regions of the solution space. The results of the test problems show that the hybrid methodology improves on previous approaches.

1 Introduction

For cost-effectiveness, communication networks are designed in a multilevel hierarchical structure consisting of backbone networks and Local Access Networks (LAN) (Boorstyn and Frank 1977). LAN is the lower level of this hierarchy in which the users in a defined proximity are connected with each other directly or via a communication center (trunk or host). In the higher level, the backbone network connects LANs to each other. A simple model for a backbone network is an undirected graph $G=(V, E)$ with node set V and arc set E . In this model, the nodes represent the connection points where LANs are hooked up to the backbone network via gateways. In addition to being connection points, the nodes are the processing units that carry out traffic management on the network by forwarding data packets to the nodes along their destinations (i.e., known as routing). The arcs represent the high capacity multiplexed lines on which data packets are transmitted bidirectionally between the node pairs.

Notation

$G=(V, E)$ undirected graph G with node set V and arc set E
 $|V|$ cardinality of node set V

$\{i, j\}$ undirected arc connecting node i and j
 $(\{i, j\} \equiv \{j, i\})$
 f_{ij} average traffic load (bps*) on $\{i, j\}$
 c_{ij} capacity (bps) of $\{i, j\}$
 γ_{ij} traffic demand (bps) from node i to node j
 γ total traffic on network, i.e., $\gamma = \sum \gamma_{ij}$
 T average packet delay of a network
 T_{max} maximum allowable packet delay
 d_{ij} cost of $\{i, j\}$
 $P(t)$ population at generation t
 m population size
 a relaxation parameter for the delay constraint
 $A^i(t)$ the i^{th} member of the population at generation t
 NG number of generations

Similar formulations for the backbone design problem can be found in (Boorstyn and Frank 1977), (Gerla and Kleinrock 1977), (Pierre *et al.* 1995), (Pierre and Elgibaoui 1997), and (Tanenbaum 1981). The problem is formulized, from the references, as follows:

Given:

- Location of each node.
- Average peak-hour traffic requirement between node pairs.
- Link options in terms of capacity.
- Cost structures.

Minimize:

- Total network design/operation cost.

Over:

- Topology.
- Capacity of links.
- Routing policy.

Subject to:

- Delay/throughput constraint.
- Network reliability constraint.

* bits per second

The average packet delay of a network (T) is given as

$$T = \frac{1}{\gamma} \cdot \sum_{(i,j) \in E} \frac{f_{ij}}{c_{ij} - f_{ij}} \quad (1)$$

The average traffic flow on a link is a function of the routing algorithm and the traffic load distribution on the network. T , as given here, does not include the delay due to propagation and nodal processing. However, this measurement is accurate enough to represent the steady-state network performance under the assumptions that the packet arrivals are Poisson distributed, the packet length is exponentially distributed, there is no delay on the nodes, and the nodes have infinite storage (Gerla and Kleinrock 1977). Therefore, Eq.1 is used in references (Boorstyn and Frank 1977), (Newport and Varshney 1991), (Pierre *et al.* 1995), and (Pierre and Elgibaoui 1997) and also used in this paper as a good surrogate for network performance.

The reliability constraint is concerned with the ability of a network to be available to provide the desired service to the end-users. In this sense, network reliability is the steady state availability of a network. The definitions and applications of various quantitative network reliability measures can be found in (Colbourn 1987). Among them, two-terminal (also known as s - t connectivity) and all-terminal reliability are the most widely known. Two-terminal reliability is defined as the probability that there exists at least one path between a source node and a sink node. All-terminal reliability is the probability that there exists at least one path from each node to every other node in the network, i.e., the network has at least one spanning tree. It is unlikely that a polynomial time algorithm exists to calculate two-terminal and all-terminal reliability for undirected networks (Provan and Ball 1983). Therefore, a variety of simulation methods (Fishman 1986) and bounds (Ball and Provan 1983), (Colbourn 1987), (Jan 1993), and (Konak and Smith 1998) have been proposed for efficiently estimating all-terminal and two-terminal network reliability. In addition to quantitative reliability and availability measures, designers tend to use computationally more tractable reliability constraints that depend on network connectivity measures, such as K -node-connectivity and K -arc-connectivity. The connectivity measures (also called survivability measures) are concerned with the vulnerability of a network and ensure a minimum reliability and survivability (Newport and Varshney 1991). A network is said to be K -node-connected, if there exist at least K node-disjoint paths between each pair of the nodes. Likewise, in a K -arc-connected network, each node pair must be connected by at least K arc-disjoint paths (Pierre and Elgibaoui 1997). The K -node-connectivity is a tighter constraint than the K -arc-connectivity, since deleting a node from a network requires removing all of the arcs

incident to that node. In addition, K -node-connectivity deals with the vulnerability of the network against not only link failures, but also node failures.

2 Background

The previous methods proposed for the backbone design problem are basically various types of graph perturbation heuristics, which start from an initial solution and evolve toward a local optimum by small perturbations to the networks. The Branch Exchange Method (BXC) (Tanenbaum 1981) starts with an initial topology. In each iteration, two relatively close links are removed and replaced with two new links using different combinations of the four nodes from which the deleted links emanate. The Cut-Saturation Algorithm (CSA) consists of four basic operations: i) add-only, ii) delete-only, iii) perturbations, and iv) chain collapsing (Boorstyn and Frank 1977). In the add-only operation, a saturated cut, a set of links whose removal separates a network into two sub-networks G_1 and G_2 , is determined by sequentially removing highly utilized links from the network. Then, link $\{x, y\}$ is added to the network such that $x \in V_1$, $y \in V_2$, and nodes x and y are at least five links distance from each other. In the delete-only operation, the most underutilized and expensive link is removed from the network. The perturbation operation is a combination of the add-only and the delete-only operations. Finally, in chain collapsing, a relatively long path between two nodes x and y is replaced by link $\{x, y\}$. It is reported that CSA outperforms BXC (Gerla and Kleinrock 1977). A modified CSA is defined in (Newport and Varshney 1991). In addition to network performance, which is the only consideration in the original CSA, the modified CSA also incorporates network survivability into the design process. MENTOR (Kershenbaum *et al.* 1991) depends on the idea that there should be a direct link among the nodes having sufficiently high traffic volume and the other traffic can be sent via a non-direct path. MENTOR generates topologies with this property. Gavish (1992) formulates an overall design problem, which includes both LAN and backbone design together, as a nonlinear optimization problem. The solution methodology depends on Lagrangean relaxation and subgradient optimization procedures. In (Jan 1993), a branch-and-bound approach is proposed for network cost optimization under an all-terminal reliability constraint. In this paper, all-terminal reliability is estimated by using an upper bound.

With the increasing popularity of artificial intelligence and meta-heuristics techniques, many applications of these modern optimization methods to the backbone design problem have emerged. In (Pierre *et al.* 1995), simulated annealing (SA) is used to find the minimum cost K -node-connected network satisfying the required maximum average packet delay. Using tabu search (TS), Pierre &

Elgibaoui (1997) solve the same type of the problem as in (Pierre *et al.* 1995). The TS approach is also applied to LAN design (Costamagna *et al.* 1998) and ring-chain network topology (Lee and Koh 1997).

Evolutionary algorithms have also been applied to the problem. In (Dengiz *et al.* 1997a and 1997b), a GA combined with local search and network repair algorithms is proposed to find the minimum cost network topology subject to an all-terminal reliability constraint. In this study, the links are assumed to have the identical reliability and all-terminal reliability is estimated by an upper bound and Monte Carlo simulation. In a similar study (Deeter and Smith 1999), the identical link reliability assumption is relaxed by using different link reliabilities and multiple link options. GA has been implemented to address various network-related problems, such as optical network design (Karunanithi and Carpenter 1997), the probabilistic minimum spanning tree problem (Abuli *et al.* 1994), and computer network expansion (Kumar *et al.* 1995).

3 Methodology

Among the approaches summarized above, CSA, BXC, and MENTOR can be classified as local search algorithms. They start with an initial network topology and try to reach the optimum solution by means of small perturbations to the current solution. However, local search algorithms suffer from becoming trapped in local optima if the solution space is complex (Michalewicz 1996), which is the case for the backbone design problem. SA and TS are meta-heuristic techniques depending on neighborhood search concept. Although they can escape from local optima, these techniques cannot simultaneously explore the various regions of the solution space. On contrary, GA performs search by exploiting information sampled from different regions of the solution space (Reeves 1993). The combination of the crossover and mutation helps GA to escape from getting mired in local optima. These properties of GA provide a good global search methodology for the backbone design problem.

Crossover and mutation are generally blind operators, i.e., they do not use any problem specific information. This property creates some problems for the backbone design problem. Evaluating a candidate network requires computationally very expensive tasks such as calculating (or estimating) network reliability, testing for connectivity, and determining the traffic flows. In such circumstances, it is highly desirable to maximize the improvement to the objective function with respect to the computational effort during the search process. By using an existing heuristic method such as CSA, candidate networks can be efficiently improved, which might be inefficient to do through mutation and crossover. In this study, GA and a version of CSA are merged in order to both enhance the effectiveness

of GA and to avoid the limitations of CSA due to its local search nature. The hybrid GA and CSA will be called Hybrid Genetic Algorithm (HGA).

GA requires encoding the problem. The incidence matrix representation is the simplest way to represent graphs. In this representation, a network is stored in an $n \times n$ matrix $\mathbf{A} = \{a_{ij}\}$, where $n = |V|$ and $a_{ij} = 1$ if link $\{i, j\} \in E$, $a_{ij} = 0$ otherwise. The encoding scheme of HGA uses the incidence matrix with a slight modification to handle link options. Networks are represented by an incidence matrix \mathbf{A} such that its ij th entry a_{ij} equals integer values corresponding to the link options if link $\{i, j\}$ exists and 0 otherwise. For example, $a_{12} = 2$ means that link $\{1, 2\}$ exists and it is a type 2 link. Since links are bi-directional, i.e., $\{i, j\} \equiv \{j, i\}$, only the upper half of \mathbf{A} is needed for representation.

HGA uses a uniform crossover operator with a K -node-connectivity repair algorithm. Two parent networks are selected from the population with q -tournament selection with $q=2$ (Bäck 1996). The parents produce only one child network. If both parents possess a link, the child will have that link. Likewise, if both parents do not possess the link, the child will not. If one parent has the link and the other has not, the child will have the link with 0.5 probability. This crossover operator ensures that the child inherits the common topological properties of the parents. After crossover, if the child is not a K -node-connected network, it is brought to K -node-connectivity by directly connecting the nodes violating K -node-connectivity.

The mutation operator is a modified version of CSA and serves as a local search operator. Mutation is only applied to the parents. After mutation, the original network is replaced with the mutated one. The flow chart of CSA is given in Figure 1. When a network has higher average delay than the desired level (T_{max}), it is most likely that some links are operating under traffic loads that are very close to their capacity. In such an instance, network performance can be improved by adding a link, which will share the load of the overloaded links. To find out the best candidate link, highly utilized links are removed from the network, one by one, until the network is separated into two disjoint networks G_1 and G_2 (i.e., $V_1 \cap V_2 = \emptyset$). The removed links constitute a saturated cut set (S). The shortest link $\{i, j\}$ such that $i \in V_1$, $j \in V_2$, and $\{i, j\} \notin S$ is added to the network. However, this procedure sometimes fails to find such a link, particularly, when the network is densely connected. In this case, the algorithm searches for the longest path that data packets have to travel in terms of link distances. The longest path is replaced by directly connecting the two end nodes of the path. While link addition improves network performance/connectivity, link deletion aims to reduce the cost of the networks that are

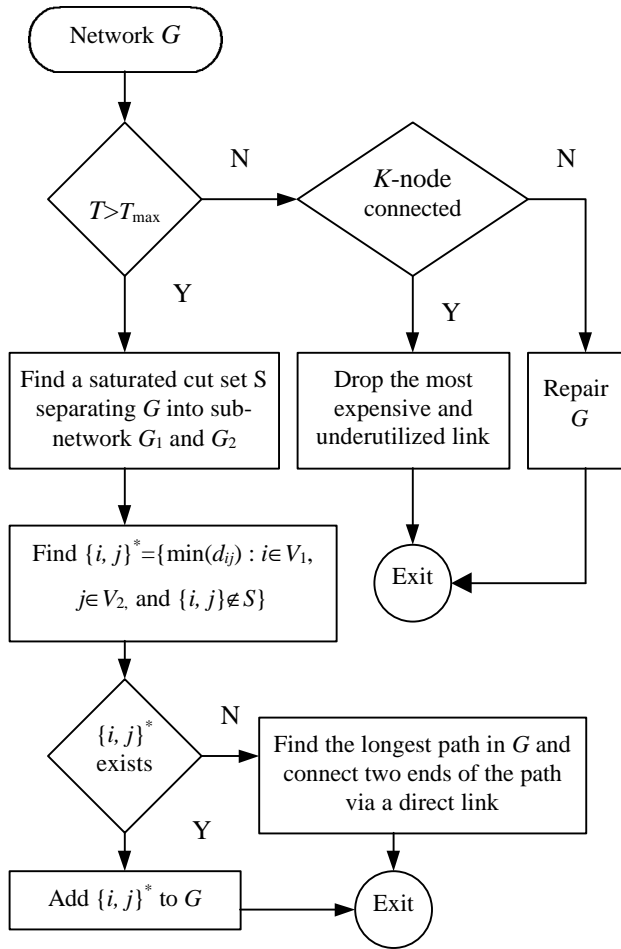


Figure 1 Flow chart of the local search algorithm (the mutation operator).

already feasible. If a network is K -node-connected, CSA deletes the link maximizing the quantity $d_{ij} \times (f_{ij} - c_{ij}) / c_{ij}$. Notice that link deletion may destroy the feasibility of a network in terms of both connectivity and performance. However, the network will not be very far away from the feasibility, indeed, just one link. Handling this infeasibility is discussed next.

Due to the two hard constraints, the selection procedure for the next generation is complex. Although the repair algorithm assures K -node-connectivity for the offspring and the initial solutions, networks violating the connectivity constraint appear in the population due to link deletion in mutation. Besides, CSA is an improvement algorithm and

does not guarantee feasibility with respect to average delay. In conclusion, the population could contain infeasible solutions at any time. Discarding these solutions is not desirable, because an infeasible solution might be closer to the global optimum than many feasible solutions. In the selection routine, first, the offspring and the mutated solutions are sorted according to the following two rules:

- Between two feasible or two infeasible solutions, the one with the lower cost is superior to the other one.
- Between one infeasible and one feasible solution, if $T_{infeasible} \leq (1 + \alpha T_{max})$, where α is less than 0.1, the infeasible solution is treated as if it were feasible and the first rule applies. Otherwise, the feasible solution is superior to the infeasible one, regardless of cost.

After sorting, the first m solutions are selected for the next generation. Notice that the selection procedure allows good infeasible solutions to survive one more generation. By doing so, they are given another chance to undergo crossover and mutation, which might improve them further. The evaluation of networks depends on the routing and the capacity assignment algorithm. Hence, it is problem specific. However, assuming that a static routing algorithm is used (where the routing policy is determined once and is not changed with changing traffic patterns), the general procedure for the evaluation of the networks is as follows: i) determine the routing schema; ii) calculate the average flows on the links; iii) determine the capacity of the links; iv) calculate the cost of the network; v) calculate T and test K -node-connectivity.

HGA starts with m randomly generated solutions that satisfy K -node-connectivity. However, the initial solutions might violate the average delay constraint. In each generation of HGA, m offspring are created by crossover. The offspring are added to the population, which increases the population size to $2m$. The original members of the population (the first m members) are mutated and replaced by the mutated ones. Then, the final population, which includes m offspring and m mutated solutions, is sorted according to the rules given before. Finally, the first m solutions of the sorted population survive for the next generation.

The GA algorithm and CSA are merged in the mutation operator. The mutation operator, which is based on CSA given in Figure 1, functions as a local search operator. A the mutation operator is expected to improve the performance or cost of solutions. The outline of HGA is given in Figure 2.

Input:

GA parameter set: $\{\eta, NG, a\}$

Problem: number of nodes and locations, cost matrix,
traffic requirement, and constraints.

start

$t:=0$

initialize: $P(0):=\{A^1(0), A^2(0), \dots, A^m(0)\}$

evaluate: $P(0)$

while $(t < NG)$ **do** {

for $i:=0$ **to** m **do** {

 select $A^m(t)$ & $A^f(t)$ **such that** $A^m(t) \neq A^f(t)$

$A^{m+i}(t) := \text{crossover}(A^m(t), A^f(t))$

 evaluate: $A^{m+i}(t)$ }

for $i:=0$ **to** m **do** {

$A^i(t) := \text{mutate}(A^i(t))$

 evaluate: $A^i(t)$ }

 sort: $P(t):=\{A^1(t), A^2(t), \dots, A^{2m}(t)\}$

 select: $P(t+1):=\{A^1(t), A^2(t), \dots, A^m(t)\}$

$t:=t+1$; }

print the best feasible:

end

Figure 2 The outline of HGA.

4 Computational Results

To test the efficiency and effectiveness of HGA, the problem set given in (Pierre and Elgibaoui 1997) is used. In this reference, TS is applied to the problem and the results are compared with the results of CSA, SA, and GA. Therefore, it provides a good base for comparison. In order to save space, the input data for the problems are not presented here. The given data of the problems includes the Cartesian coordinates of the nodes, the traffic load between each node pair, the average packet size, the possible link options in terms of capacity, the fixed and variable costs of the link options, and the routing policy. The objective function is to minimize network cost subject to K -node-connectivity and maximum allowable packet delay constraints. The problem utilizes the shortest path policy for routing. In this routing policy, the packets between two nodes are sent via the shortest path between them. In HGA, the shortest paths are determined by Dijkstra's algorithm (Ahuja *et al.* 1993) and K -node-connectivity is tested by using the shortest path-augmenting algorithm given in (Ahuja *et al.* 1993). The connectivity repair algorithm is also an extension of this algorithm.

The problem of determining link capacities for a given network topology and the traffic flows can be formulated as a linear or a mixed integer program depending on the nature of the cost-capacity relation (Gerla and Kleinrock

1977). In (Pierre and Elgibaoui 1997), however, a simple heuristic is used to determine the link capacities by assigning the minimum available link capacity (from the link options) which is greater than the flow on the links. Although better heuristics might be developed, in order to make comparisons consistent between HGA and the others, the same heuristic is also used in this paper.

Table 1 summarizes the results of HGA. To test the robustness of the methodology, HGA is run with various problem sizes, constraints, and parameters. For each case, five replications are performed using different random number seeds. The best, the worst and the average of the five replications are presented in Table 1. The average results are compared with the best results given in (Pierre and Elgibaoui 1997) by using a t -test. The p value indicates the significance of the hypothesis that HGA generates lower cost networks than the results presented in (Pierre and Elgibaoui 1997) for the given problem set. In almost all cases, HGA manages to improve on the earlier results within reasonable CPU times. The average improvements are statistically significant at a level of 0.05. The best results of HGA are always better (excluding case 9). In some cases (e.g., cases 1, 2, 3, and 4), even the worst results given by HGA are better than the earlier results.

Cases 5 and 6 present important outcomes concerning the population selection procedure implemented in this research. Both cases are the same problem, start with the same initial solutions, and have the same HGA parameters except a . In case 6, a is set to 0; hence, fewer infeasible solutions will survive to the next generation. As expected, the solutions in case 5 have lower T than ones in case 6. However, the solutions in case 5 are significantly better than the solutions in case 6 with respect to cost (with $p < 0.001$ in a paired t -test) and they are still feasible. The same experiment is repeated in cases 14 and 15 with a 20-node network. Similarly, case 14 ($a=0.1$) has better average cost than case 15 ($a=0.0$), yet the difference is not statistically significant. These results show that by letting good infeasible solutions survive, HGA may more efficiently explore the boundaries of the solution space where the global optimum will most likely reside.

Although HGA is superior in most cases, the improvement is not as significant as when the network is densely connected. In case 9, TS is better. This can be explained by the way that CSA performs. CSA searches for the bottleneck (the saturated cut) in the network. However, it can fail to find a small saturated cut set when the network is densely connected. Therefore, it cannot determine the best link to improve performance at a minimum cost. When the network is sparsely connected, CSA can efficiently determine the bottleneck in the network. Therefore, the best improvements are found when the networks are 2-node-connected.

Table 1 The results of the computational experiments (T =msec, Cost= \$/month, CPU= minutes)

Problem #	GA $\{ V , K, T_{max}\}$	GA Best		GA Average			GA Worst		Best in (Peirre & Elgibaoui 1997)		HGA vs. TS Ave.	
		Cost	T	Cost	T	CPU	Cost	T	Cost	T	% Δ Cost	p
1	{10, 2, 100}	24171	89	24923	94	0.78	25230	97	25272	69	1.3	0.076
2	{10, 2, 100}	24507	99	24901	95	1.5	25641	93	25272	69	1.4	0.069
3	{10, 2, 100}	24137	99	24646	98	2.3	24883	99	25272	69	2.4	0.004*
4	{15,2,150}	55008	114	55985	92	2.9	59138	102	65161	142	14.0	0.000*
5	{15, 2, 100}	54838	98	55510	91	3.6	56092	90	N/A			
6	{15, 2, 100}	55112	83	56369	79	3.6	57825	80	N/A			
7	{15, 3, 200}	57239	79	57946	92	2.8	58423	84	60907	79	4.8	0.000*
8	{15, 4, 200}	57334	122	59293	101	3.7	61134	92	60013	93	1.2	0.162
9	{15, 4, 200}	57334	122	59293	101	3.7	61134	92	41626	90	-29.7	
10	{20, 2, 150}	98017	74	102272	79	8.4	105284	60	144131	104	29.0	0.000
11	{20, 3, 250}	99759	102	101439	85	12.3	103714	88	105495	112	3.8	0.001*
12	{20, 4, 250}	100054	101	101166	90	16.1	102264	79	103524	88	2.2	0.002*
13	{20, 5, 250}	102617	96	104665	83	10.9	106240	72	106122	83.1	1.3	0.074
14	{20, 3, 50}	106413	49	107846	49	7.8	109739	48	N/A			
15	{20, 3, 50}	105337	49	108454	49	9.9	111026	50	N/A			
16	{23, 2, 80}	137270	70	139925	64	12.8	144460	66	210070	58	34.6	0.000*
17	{23, 3, 190}	131096	86	132774	75	20.4	134562	65	148340	91	10.5	0.000*
18	{23, 4, 190}	132002	67	133637	81	23.0	134469	85	163129	90	18.0	0.000*
19	{23, 5, 190}	133411	80	134741	89	26.0	135606	90	139154	86	2.5	0.000*

5 Conclusions

The hybrid approach of GA and CSA has shown to be a robust and efficient alternative for the backbone design problem. The effectiveness of the methodology, however, can be increased by perfecting the local search mechanism, which has difficulty with densely connected networks. Also, the selection procedure using a is an effective mean for handling highly constrained problems.

Bibliography

Abuali, F.N., Schoenefeld D.A., and Wainwright R.L. (1994) "Designing Telecommunication Networks Using Genetic Algorithms and Probabilistic Minimum Spanning Trees," in *Proc. 1994 ACM Symp. Applied Advance Computing*, pp. 242-246

Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. (1993) *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, New Jersey

Bäck, T. (1996) *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, Oxford University Press, New York

Ball, M.O., Provan, J.S. (1983) "Calculating Bounds on Reachability and Connectedness in Stochastic Networks," *Networks*, vol. 13, pp. 253-278

Boorstyn, R., Frank, H. (1977) "Large-Scale Network Topological Optimization," *IEEE Trans. on Communication*, vol. COM-25, no. 1, pp. 29-47

Colbourn, C.J. (1987) *The Combinatorics of Network Reliability*, Oxford University Press, New York

Costamagna, E., Fanni, A., Giacinto, G. (1998) "A Tabu Search Algorithm for the Optimization of Telecommunication Networks," *European Journal of Operational Research*, vol. 106, pp. 357-372

* Significant results at a level of 0.05

- Davis, L. and Cox, A. (1993) "A Genetic Algorithm for Survivable Network Design," in *Proc. 5th Int. Conf. Genetic Algorithms*, pp. 408-415
- Deeter, D.L. and Smith, A.E. (1999) "Economic Design of Reliable Networks," *IIE Transactions*, vol.30, 1999, in print.
- Dengiz, B., Altiparmak, F., and Smith, A.E. (1997a) "Efficient Optimization of All-terminal Reliable Networks, Using an Evolutionary Approach," *IEEE Transactions on Reliability*, vol. 41, no. 1, pp. 18-26
- Dengiz, B., Altiparmak, F., and Smith, A.E. (1997b) "Local Search Genetic Algorithm for Optimal Design of Reliable Networks," *IEEE Trans. on Evolutionary Computation*, vol.1, no. 3, 1997 September, pp. 179-188
- Fishman, G.S. (1986) "A Comparison of Four Monte Carlo Method for Estimating the Probability of $s-t$ connectedness", *IEEE Trans. on Reliability*, vol. 35, no. 2, pp. 145-155
- Gavish, B. (1992) "Topological Design of Computer Communication Networks-The overall design problem," *European Journal of Operational Research*, vol. 58, pp.149-172
- Gerla, M. and Kleinrock, L. (1977) "On the Topological Design of Distributed Computer Networks," *IEEE Trans. on Communications*, vol. COM-25, no. 1, pp. 48-60
- Jan, R.H. (1993) "Design of Reliable Networks," *Computers and Operations Research*, vol. 20, no. 1, 1993, pp. 25-34
- Konak, A. and Smith, A.E. (1998) "A General Upper Bound for All-Terminal Network Reliability and Its Uses," *IERC'98 Proceedings-CD*
- Karunanithi, N. and Carpenter, T. (1997) "SONET Ring Sizing with Genetic Algorithms", *Computers and Operation Research*, vol. 24, no. 6, pp. 581-591
- Kershenbaum, A., Kermani, P., and Grover, G.A. (1991) "MENTOR: An Algorithm for Mesh Network Topological Optimization and Routing", *IEEE Trans. on Communication*, vol. 39, no. 3, pp. 503-513
- Kumar, A., Pathak, R.M., and Gupta, Y.P. (1995) "Genetic-Algorithm-Based Reliability Optimization for Computer Network Expansion", *IEEE Transactions on Reliability*, vol. 44, no. 1, pp. 63-72
- Lee, C.Y. and Koh, S.J. (1997) "A Design of the Minimum Ring-Chain Network with Dual-Homing Survivability: A Tabu Search Approach," *Computers and Operation Research*, vol. 24, no. 9, pp. 883-897
- Michalewicz, Z. (1996) *Genetic Algorithms + Data Structures = Evolution Programs*, 3. Edition, Springer, New York
- Newport, K.T. and Varshney, P.K. (1991) "Design Of Survivable Communication Networks under Performance Constraints", *IEEE Trans. On Reliability*, vol. 40, no. 4, pp. 433-440
- Pierre, S., Hyppolite, M.A., Bourjolly, J.M, and Dioume, O. (1995) "Topological Design of Computer Communication Networks Using Simulating Annealing," *Engineering Application of Artificial Intelligent*, vol. 8, no. 1, pp. 61-69
- Pierre, S. and Elgibaoui, A. (1997) "A Tabu-Search Approach for Designing Computer-Network Topologies with Unreliable Components," *IEEE Trans. on Reliability*, vol. 46, no. 3, pp. 350-359
- Provan, J.S. and Ball, M.O. (1983) "The Complexity of Counting Cuts and of Computing the Probability that a Graph Is Connected," *Siam Journal of Computing*, vol. 12, no. 4, pp. 777-788
- Reeves, C.R. (1993) "Genetic Algorithms," *Modern Heuristic Techniques for Combinatorial Problems* (C.R. Reeves, Ed), Blackwell Scientific Publications, 1993, pp. 151-196
- Tanenbaum, A.S. (1981) *Computer Networks*, Prentice-Hall, Englewood Cliffs, New Jersey