Design of an Optimal Communication Network Using Multiobjective Genetic Optimization

Rajeev Kumar, V. Prasanna Krishnan and Kartik S. Santhanakrishnan
Department of Computer Science & Information Systems
Birla Institute of Technology & Science
Pilani – 333 031, India
{rajeevk, fd96032, fd95416}@bits-pilani.ac.in

***Abstract*- Designing an optimal network requires careful optimization of conflicting requirements. It is an NP hard problem. Traditional approaches to this problem have been based either on heuristics or on rigorous mathematical programming, queuing theory and network flow concepts. In this work we describe the use of the multi-objective genetic optimization technique to obtain a Pareto front - a set of solutions which are optimal with respect to a set of constraints and non-inferior to each other - for the network design problem. A prototype is developed and is the simulator is currently being tested on different sets of inputs.**

I. INTRODUCTION

Computer and telecommunication network topology design is affected by various factors like network cost, average packet delay, reliability of the network and the maximum link capacities. Optimization of one or more of such factors, which makes the network efficient, is the main objective of design in most cases. Cost and average packet delay are two factors that are often considered. However, reliability and maximum possible traffic in a link should also be kept in mind [1],[2],[3]. This requires optimization of conflicting factors, subject to various constraints. For example, reducing the packet delay could mean an increase in the link capacities, which will result in an increase in the network cost. Exploring the whole solution space for such a design problem is NP hard [4]. Practical applications will thus benefit from an efficient way to optimize these conflicting factors.

Traditional techniques like using spanning tree and queuing can typically be used for single objective optimization [1],[4], but have some disadvantages. A spanning tree network connecting the nodes (with no additional links) minimizes network cost but it fails to ensure reliability [5]. Conventional genetic algorithm solutions have optimized a single objective, subject to a single constraint. For example, Abudali *et al.* [6] assigned terminal nodes to concentrator sites to minimize costs while considering maximum capacities, using GA. However, reliability was not considered.

Reliability is a very important consideration in real networks. So we have included, a reliability constraint for the topological design of an optimal communication network. Two-connectivity repair algorithm [4] can be used for this purpose. But it is the presence of articulation points that makes the network unreliable. A two-connected network can have articulation points and hence be unreliable. So, we consider the number of articulation points in the network as the indicator of reliability and penalize any solution that has articulation points [11].

In this work we try to overcome some of the disadvantages of conventional techniques by using genetic algorithms [7]. We do not combine the objectives into one by assigning weights to them as weights are situation specific. Instead, we use PCGA [9] to optimize multiple objectives simultaneously. In our framework, we provide multiple, equally good solutions to the problem. The user can then choose a solution based on how much average packet delay or network cost can be afforded, in addition to other engineering considerations. A static routing scheme is also developed for the topology designed to get a comprehensive solution to the problem.

Essentially, given the node locations, peak hour traffic and the maximum link capacities, we attempt to minimize the network cost and the average packet delay while ensuring that the network is reliable and the traffic in any given link is within the maximum capacity.

In this paper, we first discuss a mathematical model to represent the problem. Then we discuss Pareto converging genetic algorithms and the implementation of the solution framework in the subsequent sections. We present the results in the penultimate section and briefly discuss them in the last section.

0-7803-5812-0/00/$10.00©2000IEEE

II. COMPUTER NETWORK DESIGN

The design problem has to be modeled mathematically before we can proceed. The following relations are used to obtain a mathematical model of the network design objectives and convert it into a format suitable for application of genetic algorithms for multi-objective optimization. This problem is an adaptation of the problem considered by Gerla and Kleinrock [1] for which they developed a heuristic topological design procedure.

The problem can be framed as follows:

Given :
1. Node locations (distances between the nodes.)
2. Peak hour traffic requirements (in MBPS) between the nodes.
3. The maximum permissible time delay.

A. Objectives

To design the network, we choose the optimal set of links connecting the given nodes and the capacities of these links so as to minimize the following :

1. The total cost of the network:

$$\text{Min } D(c) = \Sigma_i f(d(i)*c(i))$$

where d(i) is the length of the i^{th} link and c(i) is its capacity.

2. The average packet delay:

$$\text{Min } T(f, c) = (1 / R) * \Sigma_i f(i) / (c(i) - f(i)) + K * T_{max} * (f(i) - c(i))^2$$

Where,
f(i) is the flow along link i.
T_{max} is the maximum permissible packet delay.
K=0 if f(i) < c(i) ; no penalty because the flow is less than the capacity.
K=1 if f(i) >= c(i); flow exceeds capacity and hence a penalty has to be imposed on the individual.
R is the sum of all average packet rates flowing between every pair of nodes. Packet rate is the traffic between j and k got from the traffic matrix (bits/sec) divided by the packet size (bits/packet) [4].

B. Constraints

While optimizing these target variables, we keep certain constraints in mind.

1. Flow constraint : Flow along a link should not exceed capacity c. This, is imposed by applying a quadratic penalty on the objective function for delay, if the constraint is violated.
2. Reliability constraint : The number of articulation points is determined and this constraint is imposed on the rank.

Rank = Rank + penalty

Where,
Penalty = ceil (number of articulation points * (maximum Rank / 2))

The design variables are network topology and the routing policy.

C. Routing Policy and link flows

Static routing allows direct calculation of the channel flows and the average packet delay. Adaptive routing requires complex simulation since it is in response to current network traffic and link failures. Network configurations optimized for fixed routing are also near optimal for adaptive routing operations [1]. So we chose to use static routing over dynamic routing. The flow along each link is calculated by using the superposition principle. A summation of the traffic between the source and the destination nodes of all paths using a link gives the total flow through that particular link.

III. MULTI-OBJECTIVE GENETIC OPTIMIZATION

Mathematically, a general multi-objective optimization problem containing a number of objectives to be maximized or minimized along with constraints for satisfaction of achievable goal vectors can be written as:

Minimize/Maximize objective $f_m(X)$; m=1,2,...,M
Subject to constraint

$$g_k(X) <= c_k \qquad k=1,2,...,K.$$

where,
X = {x1, x2, ..., xN) is an N-tuple vector of variables;
F = (f1, f2, ..., fM} is an M-tuple vector of objectives.

Goldberg's condition of Pareto-optimality [7] is stated as : in a minimization problem, if an individual objective vector F_i is partially less than another individual objective vector F_j (symbolically $F_i < F_j$) *iff*

$$(F_i < F_j) \Leftrightarrow (\forall_m)(f_{mi} <= f_{mj}) \Lambda (\exists_m)(f_{mi} < f_{mj})$$

then the individual F_i dominates the individual F_j. If any other in the population does not dominate an individual, it is said to be non-dominated.

In this work we have used Pareto Converging Genetic Algorithm (PCGA) [9] which is based on tied Pareto ranking [8]. The Pareto rank of each individual is equal to the number of individuals dominating him in the multi-objective vector space. All the non-dominated individuals are assigned rank one. If Pi(t) individuals in the current population dominate an individual ai at generation or epoch t, the current position of the individual is given by:

$$Rank(a_i, t) = 1 + P_i(t)$$

The population is selectively moved towards convergence by discarding the lowest ranked individuals in each evolution. In doing so, there is no consideration of the size of sub-population or involvement of the other parameters related to sharing/mating. Additionally we remove all subjective decisions about prioritizing the objectives.

Initially, the whole set of population (N) is ranked and the fitness, which is a simple linear function, is used to map the individuals onto a conventional roulette wheel. The selected individuals are crossed-over to produce offspring. Mutation is applied to the population to introduce random variation. The offspring are inserted into the population according to their ranks against the whole set of individuals. At this stage, the whole population set includes the parents also. Now the population consists of N+2 members. So, the lowest ranked two individuals are eliminated and the population size is restored to N. The process is iterated until a convergence criterion [9] is satisfied. This selection strategy means that we do not, at any stage, lose non-dominated solution(s).

In PCGA, at every stage, the population is evaluated in the most general sense. There is no subjective prioritization of the objectives nor are they randomly picked. In the absence of true scalar fitness, the only way to compare a parent and its offspring is a rank-based metric. In a pair-wise comparison, if the offspring are compared only against their parents, the situation will result in a tie in most practical situations, as is the case with Horn *et al* [10] and the additional safeguards of mating/sharing restrictions have to be employed for resolving the tie.

IV. IMPLENTATION

In our solution, every chromosome codes a possible topology for interconnecting the given nodes including the link capacities and the routing vector that gives the path between every pair of nodes. An initial population is randomly generated and it is ensured that the individuals are bi-connected [4].This is done to ensure that the initial population does not contain excessively inferior individuals. The chromosome of an individual is also randomly generated. Each iteration consists of evaluating the population at hand on the basis of the optimization variables and assigning ranks to them. Two parents are selected for a crossover. The crossover point is randomly chosen. Mutations are also randomly introduced. Two new offspring are formed. The fitness of all these individuals is calculated and the worst two are eliminated. Now this population becomes the initial population for the next iteration. Several such iterations are needed before an optimal solution can be obtained. The idea is to get an optimal topological design and a static routing scheme for this network.

A. Data Structures

1) Chromosome: Every chromosome codes the following -

- Possible topologies for interconnecting the given nodes including the capacity (in MBPS) if the links are present. This section of the chromosome is of length N(N-1)/2).
- A routing vector that gives the path between every pair of nodes for the topology represented by the individual. This section of the chromosome is of variable length.

The link numbers are assigned in the fully connected graph. For the particular individual, this graph is represented as:

$$C_0C_1 \ldots C_{maxlinks}$$

Where,

$maxlinks = n(n-1)/2$

Each C_i is an unsigned integer between 2 and 150

C_i = a finite number ; if the link is present

= 0 ; if a link is absent.

2) Path: Every path is of variable length and a path-list representation is used. The paths are represented such that the source and destination are implicit, i.e. path 0 is a path from node 0 to node 1, path 1 is from node 0 to node 2, and so on. If path i is {1} it implies that the full path is {0,1,2}. The source and destination are not explicitly specified.

3) Individual: The characteristics that describe an individual are the chromosome, the fitness, the size of the chromosome, an array of path lengths, the rank, the age, the cost, the average packet delay and the matrix for representing the flow along every path of the individual's network topology.

B. Algorithm

The inputs are the traffic matrix, the distances between nodes and other parameters like crossover and mutation probabilities, number of individuals needed in the population, number of nodes, maximum number of epochs and distances between all pairs of nodes.

First, the initial population is generated as explained in the next paragraph. The flow along each link is calculated using the super-position principle. Next, the objective functions for the N individuals are calculated. This is required to rank the population in the next step. The fitness of an individual is the inverse of its rank. By roulette wheel selection, two parents are selected for mating. The two parent chromosomes are crossed-over at a random site based on a crossover probability. Then the chromosomes of a few randomly chosen individuals are mutated based on a mutation probability. Paths for the new population of size N+2 are then validated. The flows along all links are calculated for all the individuals. The objective functions are then evaluated and the population is ranked. The individuals with the lowest two ranks are eliminated to restore the population size to N. This is performed till the convergence criterion is satisfied. Then, the individuals with rank one constitute the set of optimal solutions.

C. Generation of the Initial Population

The links of an individual are randomly generated. Then, it is ensured that the graph is connected, by doing a depth first search on the graph and checking if there is only one spanning tree. If not, random generation is done again till the graph obtained is connected. Now that we have a connected network, we take care of the reliability constraint by ensuring that each node has a degree of at least two. If not two-connectivity repair algorithm described by Dengiz *et al.* [4] is used. Then, the capacities of the links present are randomly chosen. The paths between every pair of nodes and the lengths of these paths are randomly generated. This decides the sizes of the chromosomes.

D. Simulation Using PCGA

The initial generation of the routing vectors was done randomly. Hence, there is a possibility that a path may be infeasible. So, we find out if any path contains an adjacent pair of nodes, which are not directly connected. If there is any such pair, we use Dijkstra's algorithm, to find the shortest path between the two nodes (greedy choice) and the pair is replaced by the resultant path. Any cycles in the resultant path are eliminated. This is essential because it is not possible to represent paths with cycles in a routing table. Every routing table has an entry indexed by the packet's source and destination addresses and the entry indicates which node the packet is to be forwarded to. The existence of cycles in a path means that there may be more than one such entry per (source node, destination node) tuple, which makes the network non-functional.

Then the flow along each link is calculated using the super-position principle. A summation of flows between the source and destination nodes of all paths using a link gives the total flow through that link. The objective functions for cost and delay are calculated using the relations discussed earlier. If the flow along a link exceeds the link capacity, a penalty is imposed on the objective function for average packet delay.

The individuals are ranked using tied Pareto ranking. A penalty is imposed on individuals that have articulation points, in order to ensure reliability of the network. The rank is used to calculate the fitness of each individual. Roulette wheel selection, using the fitness of the individuals, is done to select the two parents for mating. This ensures that fitter individuals have a better chance of selection as parents.

The crossover point is generated randomly. If the crossover point happens to be in the part representing the capacities, parts of the chromosome strings of the two parents are interchanged. The path length arrays of the two individuals are swapped. If the crossover point is in the part representing the paths, the viability of all the paths in the resultant chromosomes is to be taken into account. So, we use the crossover point to interchange parts of the path-length arrays of the individuals first. Then we calculate the crossover point (for the actual chromosomes) by adding the path lengths in the path-length arrays till the crossover point. Using this, we interchange the chromosome strings to get two new, viable chromosome strings.

Mutations may randomly change the capacities or the paths of certain individuals. This introduces random changes in the topology and/or in the routing scheme. Again, the paths have to be made valid for these new individuals. The flows through their links and the objective functions are calculated. Then, all the N+2 individuals are ranked and the two most inferior are eliminated. This process is repeated for successive epochs.

V. RESULTS

The initial population contains a large number of sub-optimal solutions (Figure 1.) But as the iterations

progress, it can be seen that the set of non-dominated solutions moves towards the Pareto front (Figure 2 and Figure3 - please note that the graphs differ in the ranges of the co-ordinates.) The population can thus be seen to improve in terms of the objectives - network cost and average packet delay. The non-dominated individuals also represent reliable networks as it has been ensured that they do not contain articulation points. In addition, the final optimal set contains a number of such alternative solutions. Detailed results and analysis will be presented during the conference.

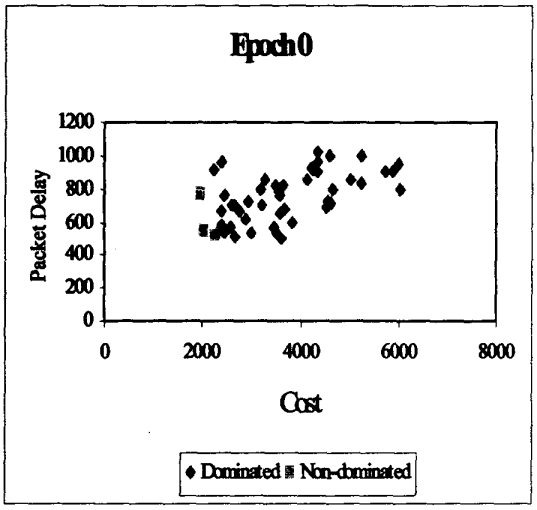

Figure 1: Population at the beginning (randomly initialized)

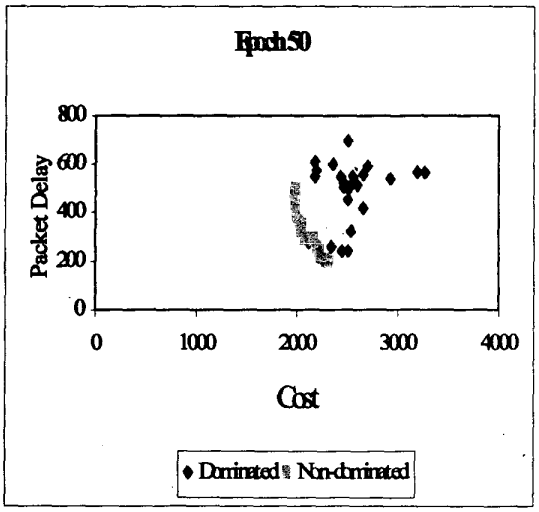

Figure 2: Population after 50 epochs

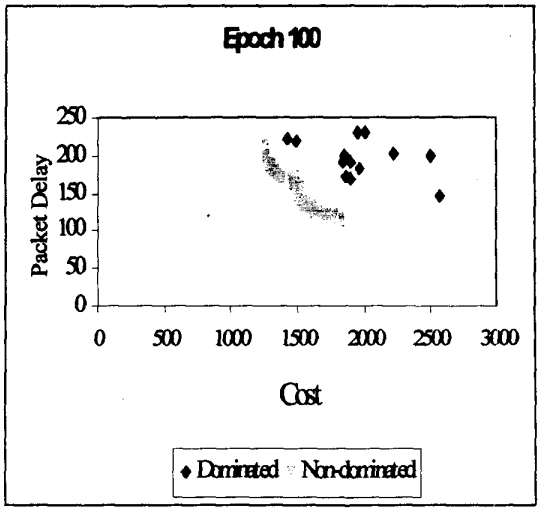

Figure 3: Population after 100 epochs

VI. DISCUSSION AND CONCLUSIONS

The outcome of the simulation is that we have a set of optimal network topologies that are non-inferior with respect to each other. We have avoided combining the multiple objectives into one based on weights in order to retain the general nature of the solution. A network designer having a range for network cost and packet delay in mind can examine several optimal topologies simultaneously and choose one based on these requirements and other engineering considerations. These topologies are reliable in case of single node failures and it is guaranteed that the maximum packet load on any link will not exceed the link capacity. Additionally, a static routing scheme is developed for each of the above networks.

We have thus provided a framework for developing a tool for communication network topology design using Pareto converging genetic algorithm. We are currently testing the simulator with different sets of inputs and tuning the genetic algorithm parameters for getting (near-) optimal sets of the solutions with minimal computational efforts. We will then analyze the solutions so obtained with those obtained with conventional approaches. The simulation can also be extended to develop a scheme for dynamic routing, which takes into account the network congestion and changes in network configuration due to node or link failures.

VII. REFERENCES

[1] M. Gerla and L. Kleinrock, "On the Topological Design of Distributed Computer Networks," *IEEE Trans. Communications,* vol. com-25. no.1, Jan. 1977, pp. 48-60.

[2] K. K. Aggarwal, Y. C. Chopra and J. S. Bajwa, "Reliability evaluation by network decomposition," *IEEE Trans. Reliability*, vol. R-31, 1993, pp. 355-358.

[3] A. N. Ventetsanopoulus and I. Singh, "Topological optimization of communication networks subject to reliability constraints," *Problem of Contr. Inform. Theory,* vol. 15, 1986, pp. 355-358.

[4] B. Dengiz, F. Altiparmak and A. E. Smith, "Local Search Genetic Algorithm for Optimal Design of Reliable Networks," *IEEE Trans. Evolutionary Computation,* vol. 1, no. 3, Sept. 1997, pp. 179-188.

[5] F. N. Abuali, D. A. Schnoenefeld and R. L. Wainwright, "Terminal Assignment in a Communications Network using Genetic Algorithms," in *Proc. ACM Computer Science Conf.*, 1994, pp. 74-81.

[6] F. N. Abuali, D. A. Schnoenefeld and R. L. Wainwright, "Designing Telecommunication Networks using Genetic Algorithms and Probabilistic Minimum Spanning Trees," in *Proc. 1994 ACM Symp. Applied Computing,* 1994, pp. 242-246.

[7] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning,* Addison-Wesley, MA: Reading, 1989.

[8] C. M. Fonseca and P. J. Fleming, " Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization," in *Proc. Fifth Int. Conf. Genetic Algorithms,* S. Forrest Ed., San Mateo, CA.: Morgan Kaufmann, 1993, pp. 416-423.

[9] R. Kumar and P. I. Rockett, "Assessing the convergence of rank-based multiobjective genetic algorithms," in *Proc. IEE/IEEE Second Int. Conf. Genetic Algorithms in Engineering Systems: Innovations & Applications (GALESIA 97,)* IEE Conference Publication No. 446, 1997, pp. 19-23.

[10] J. Horn, N. Naftpliotis and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective genetic optimization," in *Proc. IEEE Int. Conf. Evolutionary Computation,* pp. 82-87.

[11] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *Data Structures and Algorithms*, Addison-Wesley, MA: Reading, 1987, pp. 239-246.