Topological Design of Communication Networks using Multiobjective Genetic Optimization

Rajeev Kumar, Prajna P. Parida and Mohit Gupta

Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur-721302, India
{rkumar, ppparida, mgupta}@cse.iitkgp.ernet.in

Abstract - Designing communication networks is a complex, multi-constraint and multi-criterion optimization problem. We present a multiobjective genetic optimization approach to setting up a network while simultaneously minimizing network delay and installation cost subject to reliability and flow constraints. In this work we use a Pareto Converging Genetic Algorithm, present results for two test networks, and compare results with another heuristic method.

I. INTRODUCTION

Many problems in engineering and related areas require the simultaneous optimization of many objectives that may be conflicting. Topological design of communication networks, particularly mesh/wide area networks is a typical multiobjective problem involving simultaneous optimization of cost of the network and various performance criteria such as average delay of the network, throughput and reliability. 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. The problem can be stated as: given a set of node locations and the traffic between the nodes, it is required to design the layout of links between the nodes while optimizing certain criteria *e.g.*, overall cost, average per packet delay, reliability and provision for expansion. 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 an NP hard problem [1]. Practical applications will thus benefit from an efficient way to optimize these conflicting factors.

Since this is an NP hard problem, heuristic techniques have been used widely for such design. Heuristic methods that have been used include techniques, such as branch exchange, cut saturation etc. For example Jan *et al.* developed a branch and bound based technique to optimize network cost subject to a reliability constraint [2]. Ersoy and Panwar developed a technique for the design of interconnected LAN and MAN networks to optimize average network delay [3]. Clarke and Anandalingam used a heuristic to design minimal cost and reliable network [4]. However, these being heuristics, they do not ensure that the solutions obtained are optimal.

Moreover, most heuristics attempt to optimize just one objective. Alternately, the problem is broken down into a number of subproblems, solved in sequence using some heuristics thereby possibly leading to locally optimal design. Other approach to multiobjective optimization (MOO) is that weights are applied to the set of objectives and a combined objective function is optimized, but such a solution would depend on the particular choice of weights. A practical MOO approach should try to optimize all objectives simultaneously, give a range of results that lie on the (near-) optimal Pareto front, and form a set of non-dominated solutions [5].

Genetic Algorithms have been extensively used in *single* objective optimization for various communication network related optimization problems. For example, Abuali *et al.* assigned terminal nodes to concentrator sites to minimize costs while considering maximum capacity [6]. Ko *et al.* used GA for design of mesh networks but the optimization was limited to optimizing the single objective of cost while keeping minimum network delay as a constraint [7]. Elbaum and Sidi used GA to design LAN with the single objective of minimizing network delay [8]. Kumar *et al.* used GA for the expansion of computer networks while optimizing the single objective of reliability [9]. White *et al*. used GA to design Ring Networks optimizing the single objective of network cost [10].

Multiobjective genetic algorithms have been increasingly being applied to many problem domains. There are many implementations, *e.g.*, Fonseca & Fleming [11], Zitzler & Thiele [12], Knowles & Corne [13] and many others. For the details of the work – see Fonseca & Fleming [5] and Deb [14]. None of these implementations have a notion of convergence, and almost all use sharing/niching for achieving genetic diversity. Apart from selection of domain in which to perform sharing, tuning of such parameters needs *a priori* knowledge of the solution space, which is unknown in most real problems. The Pareto Converging Genetic Algorithm (PCGA) has been demonstrated to work effectively across very complex problem of unknown solutions [15]. PCGA does not need niching/sharing, it monitors convergence using rank-histograms, and thus avoids wastage of runs which do not produce superior sampling of the solution space while producing (near-) optimal Pareto front [16].

In this work, we use PCGA to simultaneously optimize multiple objectives subject to satisfaction of multiple constraints. We present results of the simulation on two network topologies. These solutions are compared with the branch exchange heuristic.

II. STATEMENT OF THE PROBLEM

Mathematically, a general multiobjective 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 $g_k(X) \le c_k$, $k = 1,2,...,K$

where,

$X = (x_1, x_2, ..., x_N)$ is an *N*-tuple vector of variables;

$F = (f_1, f_2, ..., f_M)$ is an *M*-tuple vector of objectives.

We use the following mathematical model for the problem:

A. Terms used

1) Node: Nodes of a network are the sources and sinks of traffic in the network. These are individual machines or complete networks in themselves, which can exchange data with other nodes.

2) Link: Links are network devices that transfer data between two nodes. They are assumed to be bi-directional and completely reliable. For instance, they could be made of any physical media, such as fiber optic cables. (This excludes communication via satellite links.) Links have a cost per unit distance.

3) Network equipment (NE): Network equipment is the generic term used to refer to the class of devices present in the nodes that are used for processing network traffic, and contribute to the delay of traffic flowing through them. These could be FDDI adapters, network cards etc.

B. Given Parameters

i) *N*, the total number of nodes in the network

ii) D_{ij}, physical distance between nodes *i* and *j* in km.

iii) $Traffic_{ij}$, expected peak traffic between nodes *i* and *j* in packets per second.

iv) *K*, the number of types of network equipment (NE) slabs available, giving the following details – per packet delay for processing, and maximum rate of incoming traffic that can be handled by the NE and NE cost. This is represented as an array of *K* structures, *NetSlab*. *NetSlab[i].Capacity* and *NetSlab[i].Cost* give the data handling capacity (in packets per second) and cost, respectively, of a node of type *i*.

v) *M*, the number of types of link slabs available, giving the following details – link cost per unit distance and link capacity. This is represented as an array of *M* structures, *LinkSlab*. *LinkSlab[i].Capacity* and *LinkSlab[i].Cost* give the capacity (in packets per second) and cost per km length of a link slab of type *i*.

C. Objective functions

Here we are using two objective functions - delay and cost which are defined below:

1) Optimizing Cost:

$$Cost = Costnodes + Costlinks + Costamp$$

where,

$$Costnodes = \sum_i C_i$$

$$Costlinks = \sum_i \sum_j C_{ij}$$

$$Costamp = \frac{\sum_i \sum_j D_{ij} \times A}{L}$$

C_i - cost of the network equipment at node *i*, *i.e.* $C_i = NetSlab[NodeType].Cost$

C_{ij} - cost of the link between node *i* and node *j*, *i.e.*, $C_{ij} = D_{ij} \times LinkSlab[LinkType].Cost$

L - maximum distance for which the signal is sustained without amplification, and,

A - cost of each amplifier unit.

2) Optimizing Average Delay:

$$AvgDelay = \frac{(\sum_i \sum_j Delay_{ij} \times LinkFlow_{ij})}{\sum_i \sum_j LinkFlow_{ij}}$$

The *AvgDelay* is ∞ if the network cannot handle the required traffic pattern with the existing capacities of the links and the routing policy adopted.

$LinkFlow_{ij} = \sum_k \sum_l Traffic_{kl}$ for all *k*, *l* such that the route from node *k* to node *l* includes the link (*i*, *j*).

From queuing theory,

$$Delay_{ij} = \frac{1}{(Cap_{ij} - LinkFlow_{ij})}$$

$LinkFlow_{ij}$ and $Delay_{ij}$ are 0 if there is no link between nodes *i* and *j*.

where,

$Delay_{ij}$ - link delay for packets flowing along link (*i*, *j*).

$LinkFlow_{ij}$ - actual flow along link (*i*, *j*), computed by superposition.

Cap_{ij} - capacity of link (*i*, *j*).

D. Constraints

Optimization of cost and delay functions are done subject to the following constraints:

1) Flow constraint: Flow along a link (*i*, *j*) should not exceed capacity Cap_{ij}. Checking whether the total traffic

along a link exceeds the capacity imposes this constraint. If it does, then the network is penalized.

2) Reliability constraint: The network generated has to be reliable. The number of articulation points [17] is a measure of the unreliability of the network. An articulation point of a graph is a vertex whose removal disconnects the graph. The number of articulation points is determined, and this constraint is imposed penalizing the network proportional to their number.

E. Routing Policy

Dijsktra's shortest path algorithm [17] is used for routing. The metric used for this purpose is the length of the link.

III. IMPLEMENTATION

A. Encoding used

In the encoding scheme chosen, every chromosome codes a possible topology for interconnecting the given nodes; *i.e.*, a chromosome represents a network, which is an individual in a set of potential solutions of the problem. This set of potential solutions constitutes a population. A constant length integer string representation was used to represent the chromosome. The chromosome consists of two portions; the first portion containing details of the NE's at the nodes and the second portion consisting of details of the links. For instance, if there are T types of nodes, then $\lceil \log_2 T \rceil$ bits are needed to encode a node. Thus the first portion of the chromosome consists of $\lceil \log_2 T \rceil \times N$ bits. If a link is present between nodes 1 and 2 then the first bit position in the link portion is set to 1. Thus, the second portion of the chromosome consists of $\frac{N \times (N-1)}{2}$ bits. The capacity of the link is then the first capacity value in the link slab that is greater than the minimum of the capacities of the NE's at the two node ends.

B. Initial Population

The following steps are used to generate the initial population. The NE's at the nodes are randomly assigned and maintained in the chromosome. Assuming that the individual is fully connected, a minimal spanning tree is generated using Prim's algorithm [17]. All co-tree links are then removed. A random number of links is then added from the co-tree set to the spanning tree. The number of links added is a random number in between one-third of the total number of links to half of the total number of links. This is done so that the initial population is not limited to spanning trees. This way we adopt a hybrid approach so that the time for exploitation and exploration of the search space is significantly reduced, and minimizes the number of lethals produced for large nets.

C. Evaluation of fitness of individuals

We use the notion of Pareto optimality, which is based on the concept of dominance. Goldberg's condition of Pareto-optimality [5] is stated as:

In a minimization problem, an individual objective vector $A = (a_1, ..., a_m)$ is said to dominate another individual objective vector $B = (b_1, ..., b_m)$ (symbolically $A \prec B$) iff

$$(A \prec B) \Leftrightarrow (\forall i, a_i \le b_i) \wedge (\exists i, a_i < b_i)$$

An individual is said to be non-dominated if there does not exist any other individual dominating it.

We use PCGA [16] for the solution to the problem. PCGA is based on Pareto ranking. The Pareto rank [11] of each individual is equal to one more than the number of individuals dominating it in the multiobjective vector space. All the non-dominated individuals are assigned rank one. The values of the two objectives to be minimized (cost and average delay) are used to calculate the rank of the individual. Cost is calculated as described in the problem statement. Using this scheme and the superposition principle the traffic on each individual node is calculated and hence the average delay for the network is calculated.

Based on these two objectives the rank of the individual is calculated. Fitness of an individual is calculated as:

$$Fitness = \frac{1}{(Rank)^2}$$

To follow the reliability constraint, a penalty is imposed on unreliable networks. Networks with articulation points are considered as unreliable and penalized. Penalizing involves applying a penalty value to decrease the fitness or equivalently increase the rank. The penalty is given by

$$Penalty = \left\lceil \frac{nartpts \times mxrank}{2} \right\rceil$$

where,

nartpts - number of articulation points.

mxrank - maximum rank of any individual in the population.

This value is added to the ranks of all such unreliable networks so that they may be removed as early as possible (but possibly after contributing to genetic diversity) from the population to be replaced by better, fitter and reliable individuals. For networks that have links with traffic flowing through them greater than their capacity, again, a penalty proportional to the population size is imposed. These penalties are added to the rank to get the final rank.

D. Selection of parents

Two individuals are selected by Roulette wheel selection [5] in which the probability of an individual *i* being selected

is proportional to $\frac{Fitness_i}{\sum_i Fitness_i}$. A higher chance is given for selection of fitter individuals. This ensures that fitter individuals have a better chance of being parents.

E. Crossover

Crossover is then applied on the selected parents as follows.

1) Cross over in the first portion: The crossover point is generated randomly. Initially the crossover point would lie at any position in the chromosome irrespective of the boundaries of the bits coding for a particular node's NE. This is to ensure maximum exploration, since node type values are not preserved. As the algorithm proceeds the probability of getting a crossover point within a NE's boundary in the chromosome is constantly reduced so as to exploit the collected experience regarding optimal values of NE types so far. In this case only the existing NE types in the parents can be present in the children.

2) Crossover in the second portion: In the link portion of the chromosome, since a single bit is used to code the presence or absence of the link, such considerations regarding tradeoff between exploration and exploitation do not arise. As a result, the crossover point is random.

Crossover involves exchanging portions of the bit strings corresponding to the chromosomes of the parents. The points of crossover demarcate the portions.

F. Mutation

Mutation is the operation of randomly changing some of the bits of the chromosome representing an individual in order to increase the exploration of the solution space.

G. Presence of unconnected components

Sometimes, as a result of the crossover and mutation operations, unconnected networks are generated as offspring. In this paper, we do not eliminate unconnected networks from further consideration or penalize them since we assume that unconnected networks can generate connected networks after crossover. A pool of unconnected networks is maintained. The selection of two individuals for crossover is made either from the population of connected networks or from the pool of unconnected networks. Since it is not possible to compute finite values of the objective functions for unconnected networks, the rank cannot be computed as explained earlier. Hence a different metric is required for the fitness. The fitness of unconnected networks is taken to be proportional to the number of links present in the network. The pool is restricted in size, and once it is full, a newly generated unconnected network can enter the pool only by replacing a network with the lower fitness value than itself. The network with the lowest such fitness is selected for replacement. This approach of maintaining unconnected, unfit individuals separately in the population is in accordance with the philosophy that unfit individuals can produce fit children.

H. Multi-tribal Approach

We have used the multi-tribal approach [16] to combine results from different runs of the algorithm, which have different initial populations. It was seen that in many cases after the initial population had converged there was very little improvement over successive epochs. This is true particularly for complex problem. We argue that there is always a certain inheritance of genetic material belonging to a population and there is unlikely to be any significant gain beyond some point. Therefore the multi-tribal approach has been used to obtain improved results over tribes than just running one tribe for a large number of epochs.

IV. RESULTS

The algorithm was run for networks with up to 40 nodes and convergence to an optimal Pareto front was observed in all cases. The algorithm was run with the following parameters in all cases except where explicitly mentioned:

i) Number of individuals in population – 100,
ii) Inter-node traffic between all nodes was a randomly generated number (0 –100) for each pair of nodes,
iii) All runs were performed for 100 epochs (for observing the benefits of convergence over fixed number of runs)
iv) Mutation probability was fixed at 0.05, and
v) Single point crossover was used.

The GA was run for the same problem as solved by Ko *et al.* [7]. In brief, the problem consisted of designing a packet switched mesh communication network among 10 major Chinese cities with realistic topology and traffic requirements. The design assumed a cost structure proportional to the distance among nodes and accounted for three different line rates: 6, 45 and 150 Mbps. Fig. 1 shows the initial and final population after running the algorithm for 100 epochs. Here the number of nodes is 10. (However, Ko *et al.* did not use a generic multiobjective optimization, and thus generated a single optimized value, we generate a set of solutions spread across the Pareto front.) Fig. 2 shows the movement of the Pareto front with successive epochs for the same inputs. Fig.3 shows the comparison of the genetic results by branch exchange method. Fig. 4 shows the initial and final population for a 40 node network.

The conventional optimization technique of generating the initial population from spanning trees has been used. The GA takes over from this point to give a set of optimal individuals. As observed from the figures, the GA produced considerable improvements over the initial population (we are unable to include results here because of limited space). In some of the

runs of the algorithm it was found that the initial population had networks where certain links had capacities, which were lower than the total peak traffic through them. These individuals of the population have an infinite delay. It was found that in the final population, other better individuals had replaced all these individuals.

V. DISCUSSION

The multiple objectives to be optimized have not been combined into one and hence the general nature of the solution is maintained. 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. Convergence to a Pareto Front is achieved, in some runs, by less than 100 epoch, after which the improvement is very marginal. In Fig. 2, we observe that there is no marked advantage of running the GA beyond 10 epochs, and no gain beyond 40 epochs. So, in the absence of a convergence criterion, it is a waste to run for a pre-determined and fixed number of runs. Fig. 3 shows the results obtained from a multi-tribal approach to the 10-node problem. Here, the non-dominated solutions from three tribes were merged. This gave a bit superior results than those obtained using a single tribe. Further, the merged population converges very close to the Pareto-optimal front, indicating that the algorithm is converging. The multi-tribal algorithm was run on the 40-node problem as well. It was seen that here also the multi-tribal approach produced much better results than those obtained by using a single tribe.

Thus, we conclude that a multi-tribal approach is particularly suited for large and complex problems and gives superior sampling of the front. Instead of running the algorithm on a single tribe for more number of epochs, using a multi-tribal approach is preferred. In addition to having good convergence properties, the multi-tribal approach is scalable to large networks. By using the traditional method of generating spanning trees, the hybrid approach reduces the search and this speeds up the process of genetic optimization. For small networks, the traditional approach yields near optimal solutions that are improved only marginally by the GA. However, as the number of nodes increases, the traditional method is not able to optimize the random networks much. As a result, much of the optimization is left to the Genetic Algorithm.

VI. CONCLUSIONS

In this paper, we have presented a novel approach to topological design of communication networks. We have presented the results for networks of two sizes. Results have shown that, for solving complex real-world problems using multiobjective genetic optimization, CPU time wastage can be avoided by assessing convergence to the Pareto-front, and improved solution space can be obtained by a multi-tribal approach. It is demonstrated that the approach scales well with larger networks. Results thus obtained are compared with those obtained by traditional approaches. Solving a network design problem of additional complexity by including larger number of objectives and constraints is an area of further investigation.

VII. REFERENCES

[1] 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, pp. 179-188, Sept. 1997.

[2] R. H. Jan, F. J. Hwang and S. T. Cheng, "Topological optimization of a communication network subject to a reliability constraint", *IEEE Trans. Reliability*, vol. 42, no. 1, pp. 63-69, Mar. 1993.

[3] C. Ersoy and S. S. Panwar, "Topological design of interconnected LAN/MAN networks", *IEEE J. Select. Areas Commun.*, vol. 11, no. 8, pp. 1172-1182, Oct. 1993.

[4] L. W. Clarke and G. Anandalingam, "An integrated system for designing minimum cost survivable telecommunication networks", *IEEE. Trans. Systems, Man and Cybernetics- Part A*, vol. 26, no. 6, pp. 856-862, Nov. 1996.

[5] C. M. Fonseca, and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimisation", *Evolutionary Computation*, vol. 3, no. 1, pp. 1-16, 1995.

[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, pp. 242-246, 1994.

[7] K. T. Ko, K.S. Tang, C.Y. Chan, K.F. Man and S. Kwong, "Using genetic algorithms to design mesh networks", *IEEE Computer*, pp.56-58, Aug. 1997.

[8] R. Elbaum and M. Sidi, "Topological design of local-area networks using genetic algorithms", *IEEE/ACM Trans. Networking*, vol.4, no.5, pp. 766-777, Oct. 1996.

[9] A. Kumar, R. M. Pathak and Y.P. Gupta, "Genetic-algorithm based reliability optimization for computer network expansion", *IEEE Trans. Reliability*, vol. 44, no. 1, pp. 63- 72, Mar. 1995.

[10] A. R. P White, J. W. Mann and G. D. Smith, "Genetic Algorithms and Network Ring Design", *Annals of Operational Research*, vol. 86, pp. 347-371, 1999.

[11] C. M. Fonseca and P. J. Fleming, "Multiobjective optimisation and multiple constraint handling with evolutionary algorithms-Part I: a unified formulation", *IEEE Trans. Systems, Man & Cybernetics-Part A: Systems and Humans*, vol. 28, no. 1, pp. 26-37, 1998.

[12] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach", *IEEE Trans. Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, 1999.

[13] J. D. Knowles and D. W. Corne, "Approximating the Nondominated Front using the Pareto Archived Evolution Strategy", *Evolutionary Computation*, vol. 8, no. 2, pp. 149-172, 2000.

[14] K. Deb, *Multiobjective Optimization Using Evolutionary Algorithms.* Chichester, UK: Wiley, 2001.

[15] R. Kumar and P. I. Rockett, "Multiobjective genetic algorithm partitioning for hierarchical learning of high-dimensional pattern spaces: A learning-follows-decomposition strategy", *IEEE Trans. Neural Networks*, vol. 9, no. 5, pp. 822-830, 1998.

[16] 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, pp. 19-23, 1997.

[17] T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 1990.

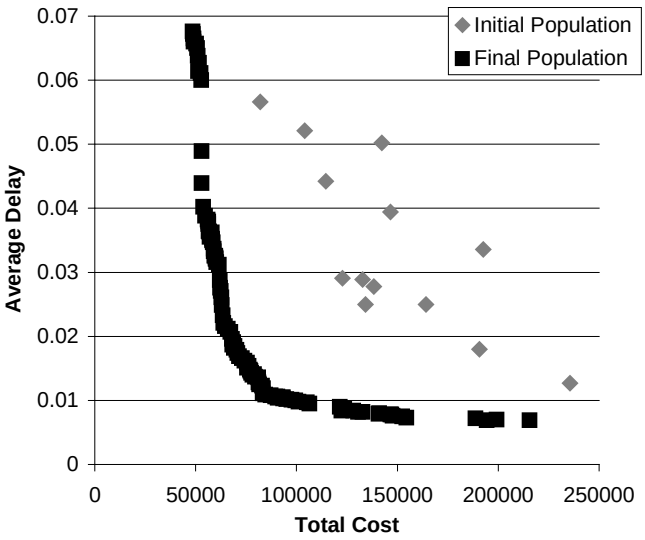

Fig.1 Initial and final population (N = 10)

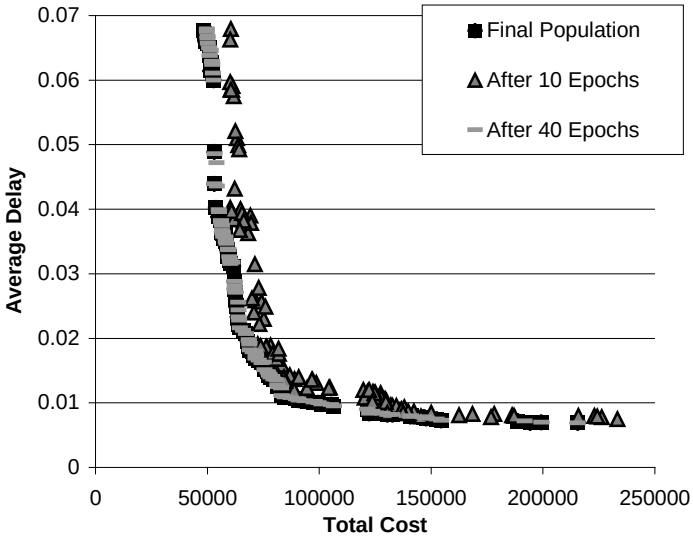

Fig. 2 Movement of the Pareto front (N = 10)

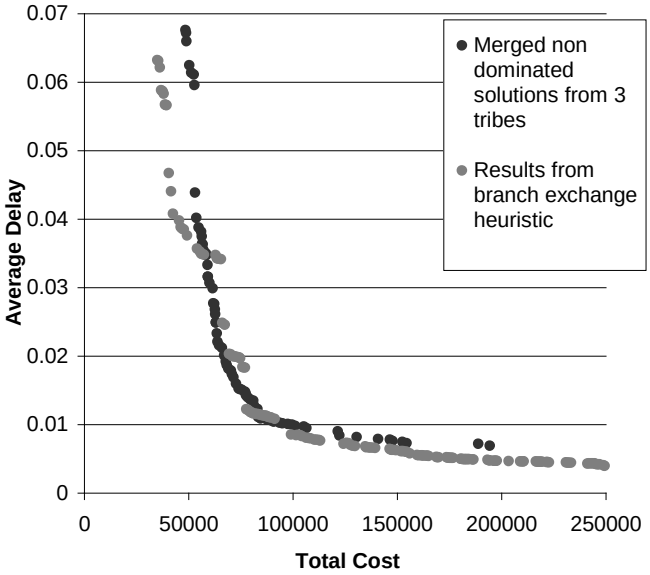

Fig. 3 Comparison of multi-tribal approach with branch exchange method (N = 10)

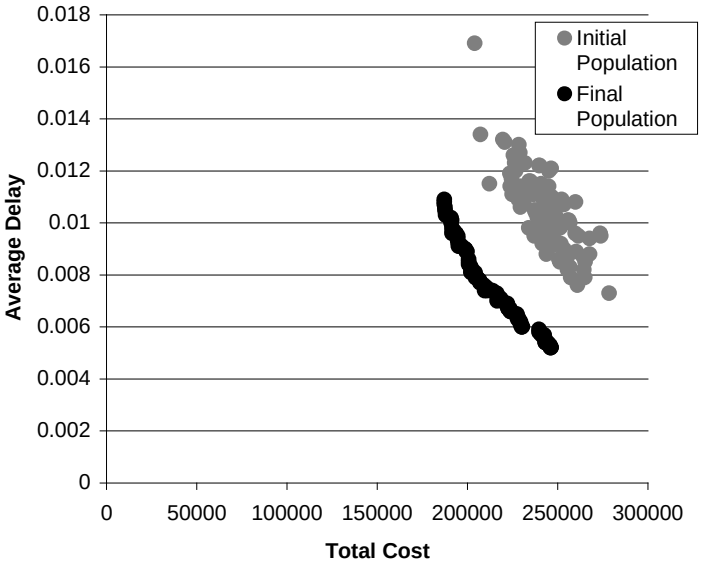

Fig. 4 Initial and Final population (N = 40)