12-1-2006

Network Design With A Genetic Algorithm

Stephen J. Sugden
Bond University, ssugden@bond.edu.au

Graham McMahon
Bond University

Marcus Randall
Bond University, marcus_randall@bond.edu.au

Les Berry

Bruce Murtagh

Network Design With A Genetic Algorithm

G B McMahon, M Randall and S J Sugden
School of Information Technology, Bond University

L T M Berry
RMIT University,

B A Murtagh
Graduate School of Management, Macquarie University

December 1, 2006

Abstract

In telecommunications network design, nodes need to be linked in an economical way to handle expected traffic. Capacity constraints, degree constraints and hop limits are to be respected. A genetic algorithm with some novel features is described. The crossover method generates an optimal child solution for the parents selected.

1 Introduction

We discuss a genetic algorithm for the following problem of network design: nodes such as telephone exchanges are to be linked in order to carry expected traffic between source-destination pairs. Each possible link has a cost per unit of traffic carried, and a capacity limit on traffic carried. There are also constraints on volume of traffic carried through each individual node and on the number of connections to each node (degree limit). In addition, there is a limit on the number of links on each path along which traffic is carried (hop limit). We would like to satisfy the traffic demand by using links which give us the lowest cost solution whilst respecting the constraints.

1.1 Formulation

Let G be the set of all undirected graphs on n nodes with G a member of the set. We represent the same G by an upper triangular node-node adjacency matrix B, with elements b_{ij}. The problem is to find a member, G^*, which minimizes the cost of transporting required origin-destination flows subject to specified link-node capacity, node degree and chain hop limit constraints. The total bandwidth (flow) requirement on virtual path connections between O-D pair $p-q$ is given by F^{pq} (without loss of generality represented as an element of an upper triangular matrix). The partial flow along the route r between nodes p and q is denoted by h_r^{pq}.C_r^{pq} is the cost per unit flow on this route, given by

$$C_r^{pq} = \sum_{i,j} c_{ij} a_{ij,r}^{pq}$$

where $a_{ij,r}^{pq}$ is 1 if the link (i,j) exists on route r between nodes p,q; 0 otherwise.

The linear mathematical programming formulation is given in 1-8.

$$Minimise_{G \in \mathbf{G}} \quad \sum_{p=1}^{n} \sum_{q>p} \sum_{r} C_r^{pq} h_r^{pq} \tag{1}$$

$$subject\ to$$

$$f_{ij} = \sum_{p=1}^{n} \sum_{q>p} \sum_{r} a_{ij,r}^{pq} h_r^{pq} \qquad \forall i,j \tag{2}$$

$$\sum h_r^{pq} = F^{pq} \qquad \forall p,q \tag{3}$$

$$\sum_{p=1}^{n} \left(F^{pi} + F^{ip}\right) + \sum_{j \neq i} f_{ij} \leq 2u_i^{\max} \tag{4}$$

$$0 \leq f_{ij} \leq f_{ij}^{\max} \tag{5}$$

$$\sum_{(i,j)} a_{ij,r}^{pq} \leq H^{\max} \qquad \forall p,q,r \tag{6}$$

$$\sum_{j=1}^{n} (b_{ij} + b_{ji}) \leq d_i^{\max} \qquad \forall i \tag{7}$$

$$0 \leq h_r^{pq} \qquad \forall p,q,r \tag{8}$$

In this formulation, n is the number of nodes, $f_{ij}^{\max}$ is the upper bound on the available capacity (total flow) on the link (i,j), $H^{\max}$ is the upper bound imposed on the number of links in a route (the hop limit). Solutions may be expressed in several different ways. In this work, they are generated as flows on chains. Thus we seek an optimal network synthesis and optimal chain flow pattern. Various alternative formulations of the problem are discussed in [7]; the formulation has a significant impact on the appropriate solution technique. The problem is related to classical work on network flows but adds further realistic constraints, thus making it applicable to industry-based problems.

2 Solution Representation

Genetic Algorithms (GAs) belong to a broad class of function optimization techniques known as evolutionary computing. GAs are modeled on the biological selection and reproduction of genetic material, known as chromosomes. This follows a Darwinian or natural selection approach in which the fittest chromosomes survive and reproduce while the others perish [8]. In terms of network optimization, the chromosomes represent potential network topologies. In this paper, we explore the application of GAs to this problem, as few empirical studies have so far been undertaken.

In designing a genetic algorithm, a key feature is the representation of a solution to the problem in the form of a chromosome. This causes particular problems in the context of network design, where the solutions are large and any simple way of performing a crossover on two complete solutions is likely to be ineffective because the child solution will usually be infeasible.

This problem may be circumvented by storing as a chromosome not the solution itself, but some more compact string from which the solution may be derived. In earlier work, the network topology was used as the chromosome, from which an optimum solution may be obtained by linear programming (LP), so long as the cost function is linear [4]. Crossover, while not trivial, is achievable for this representation. This approach has considerable appeal because the problem is divided into a discrete and a continuous part, with the genetic algorithm finding the topology, and the LP optimizing the flows on the topology. For this particular chromosome, the time taken by the LP procedure is a serious problem for large networks, despite our use of the well-tested MINOS code of Murtagh and Saunders [9].

Another approach is to use a permutation of the nodes as a very compact chromosome. We then need an heuristic procedure, controlled by the permutation, which generates good quality solutions. Provided this heuristic is efficient, we then have a method which solves large problems in acceptable time, since crossover on permutations is simple and well understood [8]. The main drawback is that since the number of permutations is much less than the number of feasible solutions, the heuristic procedure can generate solutions in only a small subset of the solution space, and the optimum solution will often lie outside this space. Of course genetic algorithms usually do not find optimum solutions for large problems, so this disadvantage may not be serious.

An alternative method is described here which uses yet another form of chromosome, containing most of the solution details. We use an array of strings, each string representing a path used to carry traffic in the solution. The only information discarded from the solution is the quantity of flow on each path.

3 Solution generation

Since the ow quantities are missing, if we wish to regenerate a complete solution from a chromosome of this type, then linear programming (or an heuristic) must

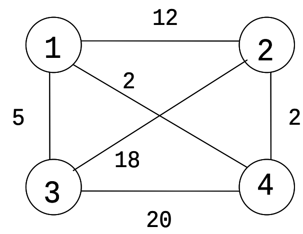

Figure 1: Figure 1 Link costs for example

be used to find the optimum allocation of flow on the paths. When a new child chromosome is created, the same process is used to generate a complete solution. However, the paths available will be fewer than those which are available from the topology, and hence the flow optimization is faster than with a method which uses the topology as a chromosome.

3.1 Crossover

A problem with the crossover technique of reproduction is that it often produces infeasible child solutions [8]. To overcome this, we allow the LP solver to perform the crossover as well as the ow optimization aspects of the problem.

Instead of constructing a new chromosome from sections of the parent chromosome, restricting the new chromosome to a fixed size, we commence by forming the union of the parent chromosomes. That is, we construct the set of all paths used to carry traffic in both parents.

This set of paths is then passed to the LP and flows optimized on this larger set of paths. In general, not all paths will carry traffic in the optimum flow pattern. The paths actually used become the child chromosome.

An advantage of the method is that provided we can generate an initial population of feasible solutions, then the feasibility of each subsequent solution is guaranteed, since we solve a problem which has two known solutions (the parents).

Another feature is that the child is the optimum solution which can be generated from these parents. This means that the genetic algorithm will converge more rapidly than a normal GA. If mutation is not implemented, then the GA will often converge to a solution which consists of the optimum allocation of flow on the set of all paths which were present in the initial population of solutions.

Table 1: Two Parent Solutions

Solution	A		Solution	B
Path	Traffic		Path	Traffic
1,2	5		1,2	5
1,3	7		1,3	8
1,2,3	1		1,4	10
1,4	10		2,3	1
2,1,3	3		2,1,3	2
2,4	6		2,4	6
3,2,4	2		3,2,4	2

3.2 Example

A simple four node problem is used to illustrate the method. The link costs appear in Figure 1. The matrix of traffic demands is as follows:

$$\begin{pmatrix} 0 & 5 & 8 & 10 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

The capacity of each link is 10 units, and the capacity of each node is 30 units. The hop limit is 2 links. Two solutions to this problem, expressed as flows on paths/ are shown in Table 1.

To obtain a child solution from these parents, we discard the flow quantities and produce the union of these paths:

$$\{(1,2),(1,2,3),(1,3),(1,4),(2,3),(2,1,3),(2,4),(3,2,4)\}$$

This enlarged set is submitted to the LP for flow allocation, and unused paths are discarded. In this trivial example, the better parent is obtained again, as a child.

3.3 Initial population

An initial population is obtained using a method devised during some earlier work [5]. For each solution, a random permutation of the nodes is first generated. Then the nodes are taken in the permutation order and a modified version of a single source shortest path algorithm used as a means of satisfying the traffic demand from each node. Modifications to the standard algorithm are needed to take into account node capacities, link capacities and hop limits. The initial population serves as a good starting point since it contains a variety of good paths for satisfying the traffic demand.

Figure 2 Solution progress with various problem sizes

Table 2: Summary of results

n		Init cost			Best cost			Time	(sec)
	min	mean	max	min	mean	max	min	mean	max
10	373	373	373	373	373	373	4	4	4
20	1517	1525	1537	1511	1521	1529	16	17	18
30	2510	2525	2538	2510	2519	2532	63	68	75
40	5905	5913	5925	5885	5896	5902	223	226	231
50	8882	8906	8949	8882	8899	8907	505	564	755

3.4 Selection and mutation

Parents are chosen for breeding using a standard roulette wheel technique [8] which favours solutions of low cost. Currently, the method is operating without mutation, and thus is restricted to using paths which are present in the initial population of solutions. The subsequent processing has the task of choosing the best combination of paths and of finding the best allocation of flows on those paths.

4 Results

Behavior with a single 10 node problem is shown below with various population sizes. Because the heuristic for creating the initial population generally performs very well, the GA has a very good starting point. It then finds worthwhile improvements quite rapidly. This is demonstrated in Figure 2. The execution time is dominated by the repeated calls to the LP solver to allocate flows for each solution. The elapsed time for problems of various sizes appears in Figure 3.

GAs make essential use of randomness, and the initial population is random in this case. Hence there is value in solving each problem a number of times (10 in our case). Table 2 shows typical results from solving problems of sizes ranging from 10 to 50 nodes.

Since we have not yet implemented limitations on node degree in this method, we do not yet have direct comparisons for the quality of the solutions derived by this method. However some limited comparison is possible.

For a 10 node problem [3] with known solution of 1723 using a degree limit of 4, the method generates a solution of value 1672. However the solution has a node with degree 5. For a problem of size 50 (*op cit*), a solution with objective 238,382 was obtained using an earlier method. This is now known to be far from optimal, since we have obtained a solution of value 107,296 with the newer methods. Again, a direct comparison is not appropriate/ since this better solution contains two nodes with degree higher than 8.

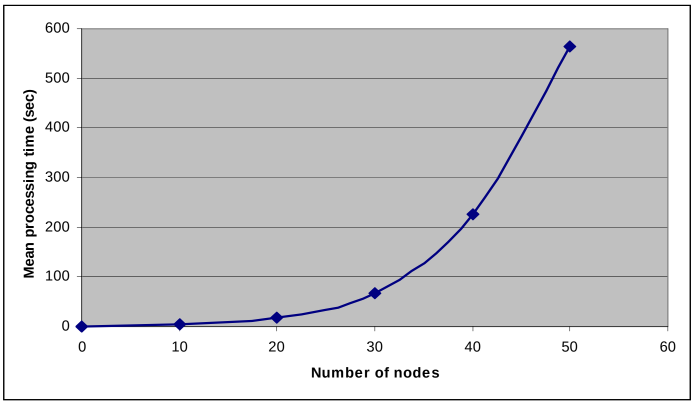

Figure 2: Processing time

5 Discussion

Even without mutation we are getting good quality solutions in reasonable time. Part of the credit must go to the good initial population. It is also gratifying that the breeding process gives rapid improvement. The lack of mutation limits the ultimate solution quality. The paths used in the initial population provide an envelope for our solutions which we cannot escape. We plan to introduce mutation in the next version of the algorithm, and also the handling of degree limits at each node. Incorporation of degree limits appears to be quite difficult. Mutation may be achieved in many ways and is comparatively trivial to implement.

6 Acknowledgment

We thank James Montgomery for the initial implementation of the algorithm. This work was supported by a grant from the Australian Research Council.

References

[1] Ahuja R. K., Magnanti T. L., and Orlin, J. B., *Network Flows,* Prentice-Hall, 1993.

[2] Berry, L.T.M., McMahon, G.B., Murtagh, B.A. and Sugden, S.J., Design of communication networks: formulations and solution strategies, APORS'97, The Fourth Conference of the Asian-Pacific Operational Research Society, within IFORS (1997).

[3] Berry, L.T.M., Murtagh, B.A., McMahon, G.B., Sugden, S.J. and Welling, L.D. Genetic algorithms in the design of complex distribution networks. *International Journal of Physical Distribution and Logistics Management*, 28, No. 5, pp377-381 (1998).

[4] Berry, L.T.M., Murtagh, B.A., McMahon, G.B., Sugden, S.J. and Welling, L.D. An integrated GA-LP approach to communication network design. *Telecommunication Systems Journal* (1990).

[5] Berry, L.T.M., Murtagh, B.A., McMahon G.B., Randall, M. and Sugden, S.J., Fast network design for telecommunications. Technical Report TR99-11, Bond University School of Information Technology (1999).

[6] Berry L. T. M., Murtagh B.A., Sugden S.J., and McMahon G.B., Application of a genetic-based algorithm for optimal design of tree-structured communications networks. *Proceedings of the Regional Teletraffic Engineering Conference of the International Teletraffic Congress*, South Africa, September 1995, pp. 361-370.

[7] Berry, L.T.M., McMahon, G.B., Murtagh, B.A. and Sugden, S.J., Optimization models for communication network design. *Proceedings of the Fourth International Meeting of The Decision Sciences Institute*, Sydney, July 1997.

[8] Goldberg D. E., (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley, (Reading).

[9] Murtagh B. A. and Saunders M. A., (1987), MINOS 5.1 Users Manual, Report SOL 83-R, Dept. of Operations Research, Stanford University.