Analyser: A Genetic Algorithm Based Network Divider

Craig Farrell, Jamie Simpson, Mike Schulze, Dorota Kieronska*
Department of Computer Science
**Department of Mathematics and Statistics*
Curtin University of Technology
Perth, Western Australia
craig@cs.curtin.edu.au, tsimpsonj@cc.curtin.edu.au, mike@cs.curtin.edu.au
dorota@cs.curtin.edu.au

January 7, 1994

Abstract

Maintaining optimal performance for computer networks is a constant problem for network administrators. Common network resources must be shared amongst competing users with dynamically changing requirements. In this paper we focus on one of the central issues in network performance, that of optimal segmentation of LANs into subnetworks. Network division is generally characterised by the dual objectives of achieving an even distribution of network traffic on proposed sub-networks, while reducing to a minimum the amount of data crossing network bridges or routers.

In this paper we describe the design of Analyser, a tool inspired by genetic algorithm techniques to produce recommendations for optimal LAN divisions. Analyser uses network communication data collected by Etherman rather than relying on network simulation or expected network traffic patterns. Analyser is general enough to be applied to a variety of LAN implementations, however our research to date has only been concerned with the monitoring and sub-networking of Ethernet™ LANs.

1 Introduction.

Local Area Networks (LANs) have quickly become an indispensable part of many organisations' communications infrastructure. This rapid increase in the use of LANs has also increased the need for efficient and cost effective network management. In response to this demand, vendors have quickly released a plethora of network management products all designed to aid network administrators performing the complex and cumbersome task of network management.

The problem of dividing a LAN is a particularly difficult aspect of network management/capacity planning since it is so intimately tied to the nature and utilisation of an individual network. Before discussing specific problems associated with dividing a LAN, let us first consider the question of when does a LAN need to be divided. A network must be segmented to maintain adequate performance when the LAN becomes saturated. Several studies have been done into ethernet[MeB76] saturation levels [BMK88], which has resulted in a number of myths as to what exactly is the saturation point for ethernet and when it has been reached. "One myth is that an ethernet is saturated at an offered load of 37%; this is an incorrect reading of the theoretical studies, and is easily disproved in practice" [BMK88].

For the purposes of this work, we will not address the problems associated with determining the saturation point for a given LAN, rather, we will accept that at some point a network will become saturated and may need to be segmented to improve performance.

Once the decision has been taken to divide a network, a series of additional considerations must be addressed before deciding on a particular configuration. Which hosts belong together? Which configuration will afford maximum potential expansion in the future? How many subnetworks should the LAN be divided into? To answer questions such as these, network administrators have traditionally relied more on intuition rather than data gathered through careful monitoring and statistical analysis. The utility discussed in this paper will aid the network administrator in making some of these decisions by utilising statistics gathered through promiscuous passive monitoring of a LAN segment during its normal operation.

2 Background

Before using any network division technique, care must be taken to collect accurate network utilisation statistics. An approach to collecting such statistics would be to use an RMON[Wal91] probe on the target segment[Lie93]. At the time of commencement for this work we did not have access to a suitable agent so it was decided to modify Etherman[SBF93] to gather network communication statistics. This is necessary since the utilisation statistics, gathered by Etherman provide the statistical basis for Analyser's recommendations.

Figure 1 shows Etherman monitoring ethernet communications. Hosts are shown as enclosed circles where the size of a circle indicates the amount of traffic generated by that host. The links between the hosts indicate network traffic with the thickness and colour indicating the amount and type of traffic on the link. Etherman has been modified to include a new option, which when called will produce a summary of network communications over a period of time. This summary can be interpreted by Analyser to produce a communication matrix$\{a_{ij}\}$ in which the entry a_{ij} is the amount of traffic measured in bytes (or optionally packets) between hosts i and j.

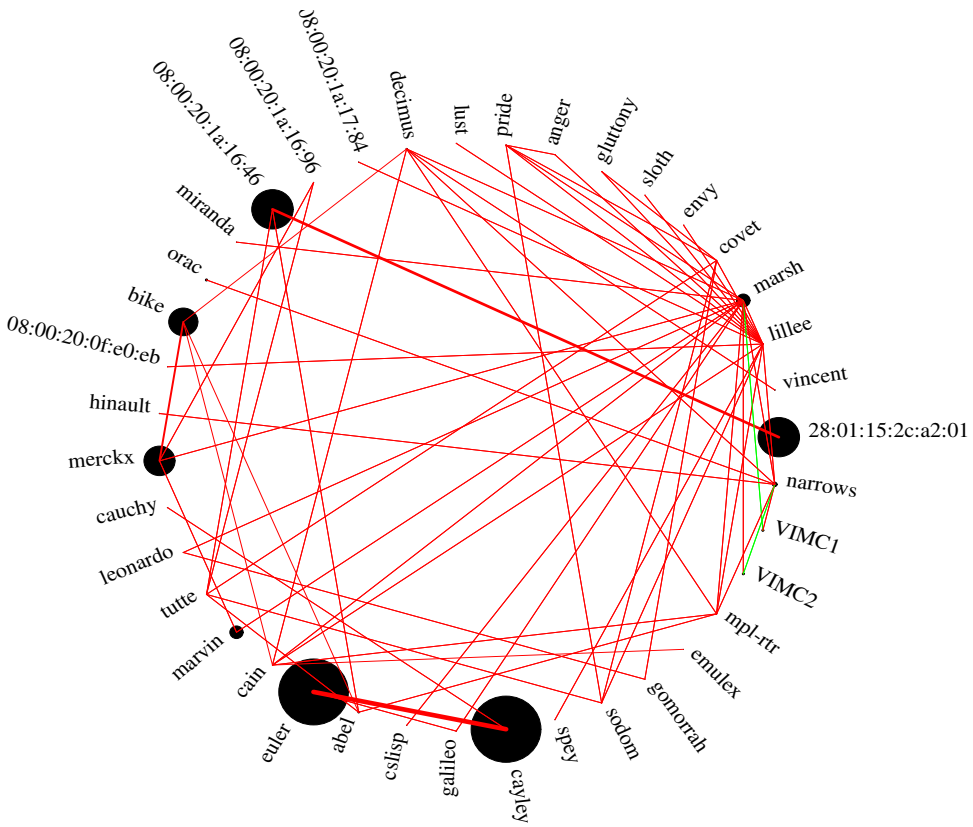

Figure 1: A Sample Etherman Display

3 Dividing Networks

Once a set of sample network utilisation statistics have been produced for a particular LAN, we can set about recommending ways to fine tune and customise the network configuration. This fine tuning can take a variety of forms but one of the common recommendations (especially on heavily loaded networks) is to divide the LAN into a set of smaller sub-networks (called "subnetting").

The goal of subnetting is to segment a LAN in such a way as to evenly distribute the traffic in each subnet, while minimising the amount of traffic propagating between subnets. The literature [RFC 917] discusses approaches to segmenting a network "...it is possible for a small subset of hosts on a LAN to monopolise most of the bandwidth. A common solution to this problem is to divide the hosts into cliques of high mutual communication, and put these cliques on separate cables." [RFC 917 pp 1].

A variety of technologies exist to segment a LAN including switches, routers, or bridges. The network division problem however, remains the same regardless of the segmentation technology. For the purposes of this paper, we will assume some generic network connecting device.

Intuitively, producing an optimal network segmentation would involve generating all the possible combinations of host configurations distributed amongst the subnets, and for each configuration, compute their intra- and inter-network traffic loads. If we have n hosts in the network then there are 2^n ways for distributing them between 2 subnets. For a network with 50 hosts there are 2^{50} possible ways of distributing them amongst 2 subnets. This means that computing an optimal division for a network with hundreds of hosts[1] can potentially take in the order of years.

Computing an optimal network segmentation is difficult because of the time complexity due to the number of possible LAN utilisation scenarios. Consider the following examples which show a few samples (simplified) of observed LAN utilisation scenarios.

3.1 Grouped Hosts

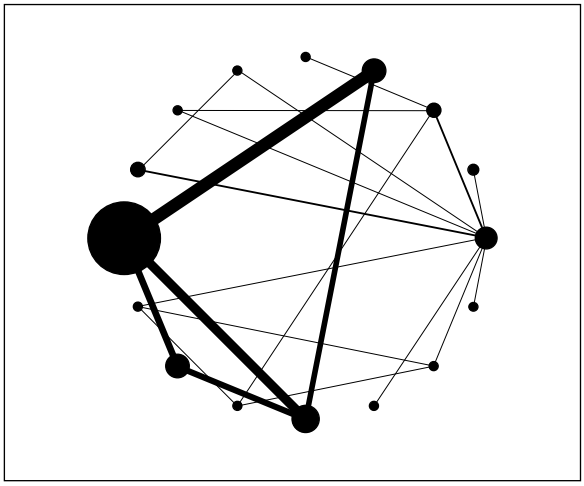

Figure 2:
A LAN communication scenario with a single group of hosts dominating the network untilisation.

1. We thought LANs with hundreds of hosts were rare until we released Etherman.

Figure 2 shows a small group of hosts communicating only with each other and hence members of the group become obvious candidates to be placed in the same subnet. LAN division in these cases is relatively straight forward since we simply divide the network into two subnets with a group of functionally similar hosts in each network.

3.2 Primary Server Example:

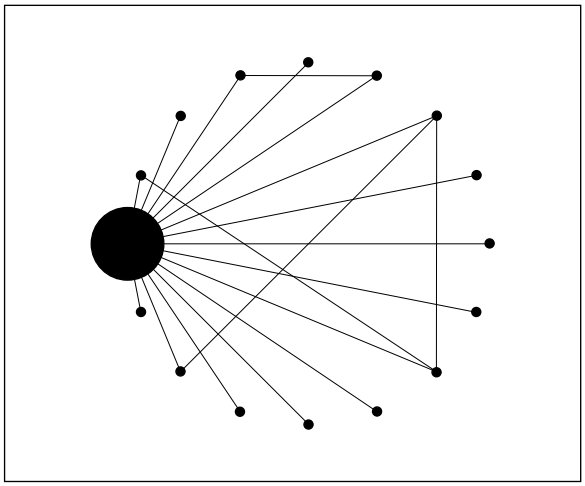

Figure 3:
A LAN with a dominant central server

In Figure 3, a central server dominates all network communications. This scenario generally occurs in LANs with a primary network server (typically NFS or X11) and a number of dependent hosts (e.g. a number of diskless workstations or X terminals). Division of these networks is not always obvious since simply dividing the hosts will not necessarily result in a better performance for hosts that do not reside in the same subnet as the server. The solution in these cases may involve purchasing an additional server as well as LAN segmentation.

3.3 Primary Server that is also part of a Group:

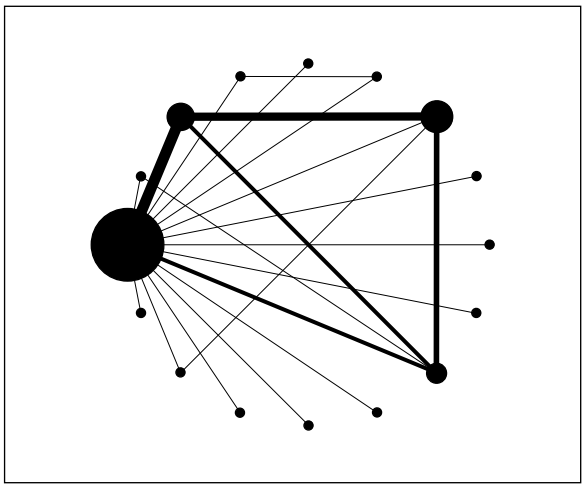

Figure 4:
A Dominant server which is also a member of a functionally similar group of hosts

Figure 4 shows the case where a group of hosts dominate network communications, while at the same time one of these hosts is also a central network server.

In our experience scenarios like Figure 4 are very common. A central server machine has a unique network resource, (requiring all the network hosts to sporadically communicate with it) while also being the primary disk server for a particular package used by a group of hosts. Variations on this theme include multiple servers, multiple groups, and seasonally changing demands made by users. Optimal network division in these cases is not obvious nor trivial to compute.

4 Our Approach

The approach taken in Analyser when producing recommendations for LAN divisions was to use a slightly modified genetic algorithm[CoJ91]. The general outline of our approach is given in Figure 5. This segment of 'C' pseudocode shows the modified genetic algorithm that Analyser uses to produce a network segmentation recommendation. For a discussion of how to use genetic algorithms (GAs) to solve this problem see [CoJ91] which discusses GA solutions for the k-way graph partitioning problem (a similar problem).

```
read_in_hosts_and_data();
randomly_assign_hosts_to_a_subnet();

for(i=0 to number_of_generations)
{
        for(j=0 to size_of_population)
        {
                calculate_fitnes(genotype[j]);
                if (fittest_so_far)
                        remember(genotype[j]);
        }

        /* Produce a new population of */
        /* configurations to test */
        if(even(i))
                crossover_hosts();
        else
                mutate_hosts();
}
```

Figure 5: Outline of Analyser's main processing loop

Given this general outline of our approach, the exact nature of these functions is now explained.

4.1 The Fitness Function (*calculate_fitness()*)

This is the function by which we determine if a particular network division is better than another solution. This function must examine a given configuration of hosts (genotype) and return a value (the fitness) which can be compared to the fitness of other genotypes to determine the best solution. The exact specification of this function is very important since even minor changes may cause large changes in the recommended solution.

In Figure 6 we show three ways of partitioning a network into three sub-networks. In every case the fitness function

is different, and dependent on the physical arrangement.

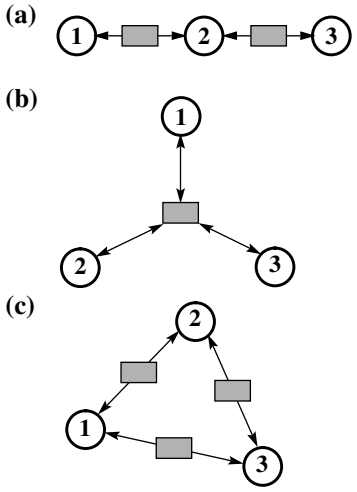

Figure 6: 3-way partitioned networks

In Figure 6(a) for example, assume that the amount of local traffic within each sub-network *i* is *traffic[i]* and *traffic[i,j]* represents the amount of traffic between sub-networks *i* and *j*. The traffic across the connection between two networks incurs extra cost and the traffic between networks 1 and 3 also appears as local traffic within the network 2 and across both connections. The following fitness functions would represent the situations depicted in Figure 6(a):

$$f = traffic[1] + (traffic[2] + traffic[1,3]) + traffic[3] + c_1 * (traffic[1,2] + traffic[1,3]) + c_2 * (traffic[2,3] + traffic[1,3])$$

with $\mathbf{c_1}$ and $\mathbf{c_2}$ being the penalty values representing the cost of sending the traffic through the connection between the sub-networks.

Figure 6(b) shows a single connection between the three sub-networks, hence the fitness of a given partition is determined by:

$$f = traffic[1] + traffic[2] + traffic[3] + c * (traffic[1,2] + traffic[1,3] + traffic[2,3])$$

where **c** is the constant representing the penalty cost of communicating between any two distinct networks. In this case, the cost is the same for communication between any pair of networks.

In Figure 6(c) a ring topology is used, hence the traffic between any pairs of sub-networks can use a single connection:

$$f = traffic[1] + traffic[2] + traffic[3] + c_1 * traffic[1,2] + c_2 * traffic[2,3] + c_3 * traffic[1,3]$$

Each fitness function would result in a different optimal partition.

For our requirements, Analyser was required to segment a LAN into 2 subnets. To produce a recommendation for two subnets, a fitness function similar to the following would be appropriate. This could easily be modified for different network topologies.

$$f = traffic[1] + traffic[2] + c * traffic[1,2]$$

Where **c** is a configurable value reflecting the cost and priority of reducing the amount of inter network traffic. **c** should also reflect the relative speed of the network connecting hardware as compared to ethernet speeds. A large value of **c** would mean that even a small increase in network traffic will make a configuration much less "fit", and hence increase the likelihood of it being rejected.

4.2 Genetically Evolving (*mutate_hosts()*)

Genetic mutation is the stepwise refinement (sometimes called "hill climbing") of a solution, towards a better solution. In this function sub-nets have their hosts randomly permuted to other sub-networks to see if a "better" (fitter) solution results.

The mutation process from one generation to the next involves taking the best solution from the previous generation and for each host randomly (approximately 10% of the time) reassigning that host's location. Performing this random permutation on copies of the best solution multiple times produces a whole population (generation) of new genotypes, some of which may be better (fitter) solutions.

In Analyser, a population of 2048 genotypes is used at each generation. Of these, 1024 are evolved from the best solution from the previous generation and 1024 are randomly generated at each iteration. The rationale for the random generation of 1024 new configurations at each iteration is to reduce the possibility of finding local maxima as opposed to finding the overall maximum of the fitness function.

All these values are user definable constants in Analyser and can be customised to allow for different sized networks and particular preferences[2].

4.3 The Crossover (*crossover_hosts()*)

The *crossover* is an important part of any genetic algorithm. In Analyser it provides a mechanism to allow a group of hosts to mutate together. The problem with only genetically evolving, is that it implements "hill climbing" where some initial state is slightly modified to see if a better solution can be produced. The problem with exclusively using this technique is that it is unlikely to produce a solution which requires a number of mutations to occur simultaneously.

In terms of the network dividing problem, the crossover is used to group a community of hosts (where their inter host communication is large) into the same subnet in a single generation. Genetic mutation can not easily accomplish this since it is unlikely to randomly mutate a group of heavily communicating hosts to the same subnet in a single mutation. Mutating a small subset of hosts (or even a

2. At present this requires the source to Analyser to perform these customisations. The source to Analyser comes as part of a Netman Source Distribution.

single host) away from a clique will dramatically decrease the fitness of the particular genotype.

The crossover implemented in Analyser is a combination of several possible strategies. Analyser's crossover involves selecting a candidate host and grouping it along with a set of other chosen hosts in the same subnet. The choice of other hosts to be grouped with the selected hosts could be a random choice (as would occur in a standard GA), but in the case of optimal network division better strategies are available. The following subsections discuss some of the heuristics used by the crossover to produce the subsequent generation.

4.3.1 Move all hosts

The "move all hosts" strategy involves selecting a candidate host and placing that host along with all the other hosts with which it communicates in the same subnet. This strategy is relatively easy to implement and works well in networks where a small group of hosts spend all their time communicating amongst themselves (see Figure 2). In networks where communities of interest exist, attempting to move one host independently of the rest would be undesirable, however crossing over the whole group of hosts from one subnet to the other may produce a better recommendation for overall network sub-division.

4.3.2 Move top n% of hosts

Select a candidate host and group it along with the top **n**% (where **n** is the grouping constant) of the hosts it communicates with in the same subnet. This strategy is most useful for a host which communicates with a large number of other hosts on the network (see Figure 3).

4.3.3 Move a random number of top hosts

Select a candidate host and group it and a random number *(n)* of hosts it communicates with (where $2 < n <$ the total number of hosts it communicates with) in the same subnetwork. This strategy is similar to that outlined in the previous strategy, however the idea of grouping a random number of hosts may produce better results in some cases.

Each of these strategies has it's own particular costs and benefits but in order to produce a general tool, we decided to adopt a strategy which was a combination of these. The crossover strategy we implemented was to select a host at random to be crossed over and if the host talked to less than half the other hosts in the network, place that host and all the hosts it communicates with in the same subnet. If the selected host communicates with more than half the other hosts in the network, place that host and the top n% of other hosts it communicates with in the same subnet.

5 Results

We have implemented Analyser in C under *Ultrix*, *SunOS*, and *IRIX*. Assuming a mutation rate of 10% Analyser's recommendation for network segmentation are given here. The solutions should be compared based upon the quality of solution produced (its fitness) as well as the number of generations taken to produce an acceptable solution.

In order to judge the quality of solution we present segmentation recommendations after 100 generations using the communication matrix given. The fitness function we used to generate segmentation recommendations was derived as follows:

Assuming α is the traffic[0], β is the traffic[1], T is the total traffic and *r* is the inter network traffic:

$$r = \mathbf{T} - \alpha - \beta$$

Our aim is to get *r* small while keeping α and β approximately equal, that is $(\alpha - \beta)^2$ small. These two requirements are not independent: minimising the inter network traffic can be achieved (with no inter network traffic) by placing all the hosts in network 0 and none in network 1, while making α and β close to equal may result in a large values for inter network traffic which should be minimised. Combining these two requirements we have to minimise:

$$r + c * (\alpha - \beta)^2 = \mathbf{T} - \alpha - \beta + \mathbf{c} * (\alpha - \beta)^2$$

where c is a constant reflecting the emphasis of balancing the respective loads on the two subnets. Since T is independent of the configuration this is equivalent to maximising

$$\alpha + \beta - \mathbf{c} * (\alpha - \beta)^2$$

A possible way of choosing **c** can be calculated as follows. Assume the maximum inter network traffic we will allow is M, and the maximum intra network traffic is N, where M and N are expressed as percentages of the total traffic T.

$$\mathbf{c} = \mathbf{M} / (\mathbf{2} * \mathbf{N} - \mathbf{T})^2$$

For brevity, we have produced some test data (see Table 1) with hostnames as single alphabetic characters. Analyser's recommendations after 100 generations were:

```
In net 0 Host A, Host B, Host K,
         Host J,Host G.
In net 1 Host C, Host D, Host E,
         Host F,Host H, Host I

Traffic in network 0 = 241947.0000 34%
Traffic in network 1 = 219293.0000 31%
Inter network Traffic = 233682.0000 33%
```

Looking at the segmentation recommendations we see hosts J and K cluster together as well as hosts A and B. This is expected given the large inter host communication. The recommended segmentation would actually realise 68% of traffic on network 0 and 65% on network 1 since all inter network traffic would be repeated on both networks in this topology.

The next set of results show performance of Analyser on some real network utilisation statistics. The input data was obtained by monitoring the Curtin University campus backbone ethernet for 15 minutes which yielded 82 hosts with 142 different links. The results given here compare the performance of Analyser in terms of the quality of solution (fitness) against the number of generations taken to produce the solution.

Table 1: Communication Matrix

Host	A	B	C	D	E	F	G	H	I	J	K
A		99000	5192	3005	9000	1009	2587	10900	7000	9128	2013
B			8090	2000	2096	3005	5480	9000	14570	6080	2409
C				11000	11980	12050	19500	21000	19000	3000	5700
D					8498	16080	14780	22600	12460	5700	1009
E						8000	19800	14500	14755	13450	6500
F							11650	26980	14500	5790	8790
G								2899	5690	7650	8600
H									5809	6750	14570
I										8000	5687
J											99000
K											

.

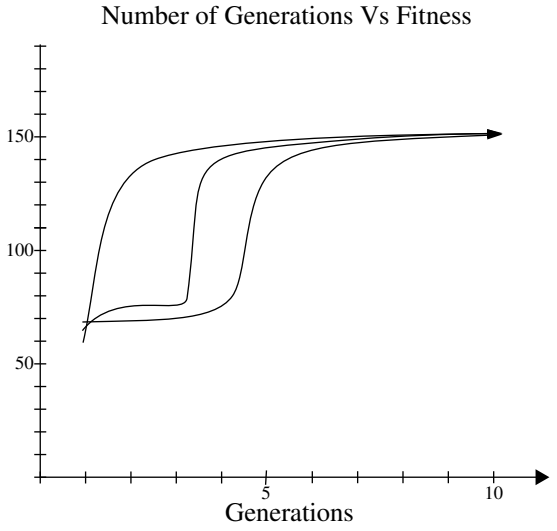

Figure 7:
Multiple runs on the same actual data and a grouping constant set at 20%.

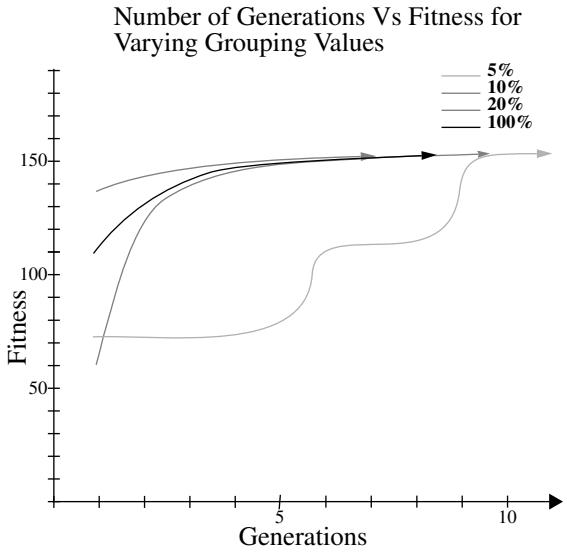

Figure 8:
Effects of altering the grouping constant on the same data

Figure 7 shows the fitness values obtained for multiple runs of Analyser on data collected from the Curtin University Campus backbone. The segmentation recomendation produced by Analyser had a fitness valuse of 1504944 which would yeild traffic of 75963 (45%) of total traffic in subnet 0, 752245 (45%) of traffic in subnet 1 and 156046 (9%) of traffic accross the segmentation hardware.

Figure 8 shows the effects on segmentation recommendations obtained by altering the grouping ratio. These results are inconclusive in providing an empirically derived optimum for the grouping constant. The variations in time taken to get close to the optimum fitness values are influenced more by the randomness of the genotype generation than any feature of the crossover algorithm.

6 Conclusion

In this paper we have described the design of Analyser, a tool inspired by genetic algorithm techniques to produce recommendations for optimal LAN divisions. Analyser has been shown to produce good approximates to the optimal solution for LAN segmentation in a manageable time. The fact that the recommendations are not guaranteed to the optimal solution is not considered to be a problem since the input data is only a sample of the actual LAN utilisation.

Recommendations are produced by maximising (or minimising) a fitness function which measures the quality of an individual configuration. The initial data used for recom-

mendations is actual LAN utilisation data collected by Etherman rather than results from network simulation or expected network traffic patterns. Analyser is a general utility which could be used for a variety of LAN technologies, however our research to date has only been concerned with the monitoring and sub-networking of Ethernet LANs.

7 Acknowledgements

Many thanks must go to the following people who contributed either knowingly or inadvertently to this project. Jeff Mogul, Phil Dench, George Christos, Doug Bruce and Louis Caccetta. Many thanks also to the internet community as a whole for the tools, suggestions and assistance they have all contributed so willingly.

8 References

[BMK88] Boggs, D., Mogul, J. and Kent, C. “Measured Capacity of an Ethernet: Myths and Reality”, Digital WRL Research Report 88/4, 1988.

[CoJ91] Collins, R.J. and Jefferson, D. R. “Selection in Massively Parallel Genetic Algorithms”, Proc. of the 4th Int. Conf. on Genetic Algorithms, pp. 249-256, 1991.

[Lei93] Leinwand, A. “Network Design Using the RMON MIB”, The Simple Times, vol. 2 no. 6., pp. 1-4, Dec. 1993.

[MeB76] Metcalfe, R.M. and Boggs, D.R. “Ethernet: Distributed Packet Switching for Local Computer Networks”, Communications of the ACM, vol. 19 No. 7, pp. 395-404, 1976.

[RFC917] Mogul J, "Internet Subnets", RFC 917, Stanford University, October 1984.

[SBF93] Schulze, M., Benko, G. and Farrell, C. “Homebrew Network Monitoring: A Prelude to Network Management”, TR 9-1993, Dept. of Computer Science, Curtin University of Technology, 1993.

[Wal91] Waldbusser, S. “Remote Network Monitoring Management Information Base”, RFC 1271.