Topological Design of Local-Area Networks Using Genetic Algorithms

Reuven Elbaum and Moshe Sidi, *Senior Member, IEEE*

***Abstract*—In this paper, we describe an algorithm for designing local-area networks (LAN's) with the objective of minimizing the average network delay. The topology design includes issues such as determination of the number of segments in the network, allocating the users to the different segments, and determining the interconnections and routing among the segments. The determination of the optimal LAN topology is a very complicated combinatorial optimization problem. Therefore, a heuristic algorithm that is based on *genetic* ideas is used. Numerical examples are provided and the quality of the designs obtained by using the algorithm is compared with lower bounds on the average network delay that are developed.**

I. Introduction

A continually growing number of users exchange increasing amounts of information. Local-area networks (LAN's) are commonly used as the communication infrastructure that meets the demands of the users in the local environment. These networks typically consist of several LAN segments connected together via bridges. Transparent bridges have been selected by the IEEE 802.1 committee as the standard means for interconnecting all 802 LAN's [1], [2]. These bridges operate below the media access control (MAC) sublayer of the open systems interconnection OSI model, so they are transparent to protocols operating at higher levels. This and other attractive attributes, such as self learning, make bridges the casual connecting element between LAN segments. The use of transparent bridges requires "loop free" paths between LAN segments. Therefore, only spanning tree topologies can be used as active LAN configurations.

In this paper, we address the problem of the topological design of LAN's. This problem includes two main issues: clustering and routing. The clustering problem consists of the following questions: 1) how many segments (clusters) should the LAN be divided to, and 2) how to allocate the users (stations) to the LAN segments (clusters) so that each user belongs to a specific segment (cluster). The routing problem is defined as the determination of segments interconnection spanning tree topology. The design objective of this paper is to achieve a minimum average network delay.

Manuscript received July 1995; revised April 1996; approved by IEEE/ACM Transactions on Networking Editor H. Miyahara. This paper was presented in part at the IEEE INFOCOM'95. The work of M. Sidi was partially supported by the Ministry of Science and Arts of the Israeli Government.

R. Elbaum was with the Department of Electrical Engineering, Technion–Israel Institute of Technology, Haifa 32000, Israel. He is now with PowerSpectrum Technology Ltd., Kiryat Bialik 27000, Israel (relb@pst.co.il).

M. Sidi is with the Department of Electrical Engineering, Technion–Israel Institute of Technology, Haifa 32000, Israel (moshe@ee.technion.ac.il).

Publisher Item Identifier S 1063-6692(96)07563-2.

Performance analysis of interconnected CSMA/CD LAN segments (Ethernet) [3] and token ring LAN segments [4], have been presented. However, those studies did not address the issue of network topological design. General discussions of the topological design problem appeared in [5] and [6]. In [6], one can also find some heuristic approaches toward the solution of the problem. These are discussed in Section IV-B. A related problem, namely the topological design of interconnected local-area and metropolitan-area networks (LAN/MAN), is studied in [7]. In this problem, one is looking for the best spanning tree topology, when the clustering of the users into the LAN segments is given. An approach based on simulated annealing is used and compared with lower bounds that are developed. The topology design problem discussed in this paper is equivalent to the LAN/MAN topology design problem when the clustering is given.

The topological design of such networks is a very complicated combinatorial optimization problem, which can be classified as NP-complete [9]. Polynomial algorithms which can find the optimal solution for this problem are not known. Therefore, heuristic algorithms are applied, searching for solutions. In [7], the authors have used a simulated annealing algorithm for a related problem. In this paper we suggest an alternative heuristic approach, called genetic algorithms (GA's) [10], [11] to address the topology design problem. A genetic algorithm is an heuristic search procedure which applies natural genetic ideas, i.e., natural selection, mutation, and survival of the fittest. Genetic algorithms were found to be well suited for related problems [12]–[14]. Lower bounds on the LAN average delay are derived in the paper and are used to examine the quality of the solutions obtained with the GA.

The rest of this paper is organized as follows: In Section II, we present a model for the LAN average delay. Then, criteria for the evaluation of LAN configurations are discussed. Section III introduces some lower bounds on the LAN average delay. In Section IV, the LAN topology design problem is formulated as an optimization problem. Methods for solving this optimization problem are then discussed. In Section V, genetic algorithms are discussed. After introducing GA, we present an algorithm, based on GA, customized to the LAN topology design problem. Some examples of topological designs are presented in Section VI.

II. Model Description

Consider a LAN that connects N stations (users). The communication traffic demands between the users are given by an $N \times N$ matrix A which is called "the users traffic matrix."

An element $a_{i,j}$ of the matrix A represents the traffic from user i to user j. Note that traffic between a pair of users can vary from heavy traffic to no traffic. Moreover, this traffic can be bursty or constant. The traffic peak rate is probably an overestimation of the traffic requirement, since most of the time the actual traffic is far below the peak. On the other hand, taking the average traffic rate as the requirement may yield poor results when heavy traffic has to be forwarded. The issue of calculating the "equivalent requirements" is discussed in [15]. For our purposes, we shall assume that traffic characteristics are known and summarized in the traffic matrix A.

We further assume that the LAN is partitioned into P segments (clusters). The users are distributed over those P clusters. The $N \times P$ "clustering matrix" R specifies which user belongs to which cluster. Thus

$$r_{i,j} = \begin{cases} 1, & \text{if user } (i) \in \text{cluster}(j) \\ 0, & \text{otherwise.} \end{cases}$$

A user can belong only to one cluster; thus $\forall i = 1, 2, \cdots, n \sum_{j=1}^{P} r_{i,j} = 1$. We define a $P \times P$ matrix S called the "cluster traffic matrix." An element $s_{i,j}$ of this matrix represents the traffic forwarded from users in cluster i to users in cluster j. Obviously, $S = R^T AR$.

Clusters are interconnected via bridges. The traffic from source cluster i to destination cluster j is forwarded through the bridges. The path between clusters i and j may include a single bridge which connects both clusters, or multiple bridges in case there is no direct connection between the clusters. In the latter case, the traffic travels through bridges and clusters on the connecting path. As stated in Section I, only the use of transparent bridges is considered [1]. The selection of transparent bridges forces the network to be configured in a spanning tree topology [2].

The problem that will be addressed is the topology design of the LAN, namely, the determination of the number of segments, the allocation of the users to the segments, and the interconnection among the segments. To that end, we have to define the performance measure that will be used in the optimization process. Several design criteria are presented in Section II-B. As in many other instances (e.g., [3]–[7]), we use the average delay in the network as our performance measure.

The main goal of this paper is to introduce the use of genetic algorithms for the topological design of LAN's. For that purpose, we shall introduce a simplified model of the average network delay. Note that this model can be simply replaced by other models which describes the behavior of real LAN's. That replacement will only result in another equation for computing the average delay.

A cluster is a LAN segment with a known capacity. A LAN segment can be token-ring, Ethernet, or other kinds of similar architectures. It is clear that the behavior and the performance of each kind of LAN segment is different from each other. Nevertheless, the average delay in all those architectures responds qualitatively in the same manner to the load growth. The average delay increases as the load over the segment builds up. When the load approaches the segment capacity, the delay approaches infinity.

An M/M/1 model [8] is used to describe a single cluster (LAN segment) behavior. For the M/M/1 model, the average delay of the next bit sent over cluster p is expected to be

$$D_p = \frac{1}{C_p - L_p} \tag{1}$$

where C_p is the cluster capacity and L_p is the total traffic load in that cluster.

A bridge is an element which connects between two clusters. The bridge detects packets that have to be passed from one cluster to another. The detection is done using a "look-up table." Packets that reach the bridge from the first cluster may be transferred by the bridge to the target cluster. If, according to the look-up table, it is decided to pass a packet to the target cluster, that packet is queued by the bridge at the target cluster. The building and the updating of the look-up table are beyond the scope of this paper and can be found in [2]. For our purposes, we can assume that the content of the look-up table is unvaried. From the description of the bridge it follows that the bridge delay for each packet is composed of two parts: 1) The delay due to the look-up, i.e., the look-up time can be assumed to be fixed for any packet passing on the bridge, and 2) the queuing delay on the target cluster. This queuing delay is taken into account in the target cluster delay model, since the traffic from the bridge is part of the total load on that cluster. Since only the look-up time contributes to the bridge delay, we can conclude that the delay per bit, due to the bridge between clusters i and j, is $B_{i,j} = b_{i,j}/l$, where $b_{i,j}$ is the look-up delay per packet and l is the packet length. If $F_{i,j}$ is the total traffic which flows on the bridge between clusters i and j, then the average delay due to bridges is

$$D_{\text{bridges}} = \frac{1}{\Gamma}\left[\sum_{i=1}^{P}\sum_{j=1}^{P} F_{i,j}B_{i,j}\right] \tag{2}$$

where Γ is the total offered traffic.

Given the above definitions, we conclude that the total average delay in the LAN is composed of the delays of the segments and the bridges. The average delay of the LAN is therefore

$$D = \frac{1}{\Gamma}\left[\sum_{k=1}^{P} \frac{L_k}{C_k - L_k} + \sum_{i=1}^{P}\sum_{j=1}^{P} F_{i,j}B_{i,j}\right]. \tag{3}$$

A. Further Definitions

The interconnection between the clusters must yield spanning tree configurations. Traffic in a chosen configuration can be expressed in terms of the following decision variables:

$$x_{i,j}^{k} = \begin{cases} 1, & \text{if traffic from cluster } i \text{ to cluster } j \\ & \text{through cluster } k \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

$$y_{i,j}^{k,l} = \begin{cases} 1, & \text{if traffic from cluster } i \text{ to cluster } j \\ & \text{passes through existing bridge} \\ & \text{connecting clusters } k \text{ and } l \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Based on these decision variables and the traffic matrices, the following quantities can be defined for any routing configuration. The total offered traffic is

$$\Gamma = \sum_{i=1}^{N}\sum_{j=1}^{N} a_{i,j} = \sum_{i=1}^{P}\sum_{j=1}^{P} s_{i,j}. \tag{6}$$

The total traffic at cluster k is

$$L_k = \sum_{i=1}^{P}\sum_{j=1}^{P} s_{i,j} \cdot x_{i,j}^{k} \qquad 1 \le k \le P. \tag{7}$$

The total traffic in the network is

$$\gamma = \sum_{k=1}^{P} L_k. \tag{8}$$

The total traffic through bridge(i, j) is

$$F_{k,l} = \sum_{i=1}^{P}\sum_{j=1}^{P} s_{i,j} \cdot y_{i,j}^{k,l} \qquad 1 \le k, l \le P. \tag{9}$$

The sum of the diagonal elements of a $P \times P$ matrix S is

$$trace(S) = \sum_{i=1}^{P} s_{i,i}. \tag{10}$$

The sum of all the elements but a diagonal of the $P \times P$ matrix S is

$$\overline{trace}(S) = \sum_{i=1}^{P} \sum_{\substack{j=1 \\ j \ne i}}^{P} s_{i,j}. \tag{11}$$

B. *Performance Measures and Design Criteria*

The issue of topological design evaluation criteria is not quite clear. We will discuss several design criteria. The suggested criteria can be separated into three different categories.

1) *Traffic Related Criteria:* The first traffic criterion is "traffic locality." The traffic locality index $= trace(S)/\Gamma$ [6]. An index value of one implies that $trace(S) = \Gamma$; thus, the traffic is completely localized, i.e., all the traffic generated in one cluster is destined within that cluster. A zero index value implies that all the traffic is intercluster traffic. A second traffic criterion is "traffic balancing." Using this criterion, we can define a traffic balance index, which reaches higher values as the traffic load between the different clusters is more balanced.
2) *Delay Related Criteria:* The minimum average network delay reflects the average delay between all pairs of users in the network. This average delay is formulated in (3). A second criterion is maximum access time, i.e., the maximum (average) delay between any pair of users. The maximum access time should not exceed a given threshold [7].
3) *Cost Related Criteria:* The equipment price and the maintenance cost can be of great significance, and should be taken into consideration. This cost can be normalized to be expressed in terms of cost per bit, and be included in any other complicated criteria.

Commonly, the criterion by which existing networks are measured is their delay performance. Therefore, the average network delay is chosen as the design criteria to the end of this paper. Nevertheless, careful observation at the other criteria indicates that both traffic criteria are implicit in the average network delay criterion.

III. Lower Bounds

Before proceeding with the description of the optimization problem, which is the design of a LAN topology with minimized average delay, we first present some lower bounds on some important quantities.

To the end of this section, an "one hop network" is assumed, i.e., all the clusters are connected to each other by a bridge. An "one hop network" is the only configuration in which it is guaranteed that traffic from source to destination clusters will not pass through any other cluster. Upon a given clustering, this configuration minimizes the total network load. Thus, "one hop network" can achieve a theoretical minimum average delay, and will be used for the derivation of the delay lower bound. Note that a "one hop network" is not a feasible solution for the topology design problem, since that problem is limited to spanning tree configurations. In the "one hop network," the traffic at cluster k as defined in (7) is reduced to

$$L_k = \sum_{i=1}^{P} (s_{i,k} + s_{k,i}) - s_{k,k} \qquad 1 \le k \le P. \tag{12}$$

Thus, the total traffic in the network is

$$\begin{aligned}\gamma &= \sum_{j=1}^{P} L_j = \sum_{j=1}^{P} \Big(\sum_{i=1}^{P} (s_{i,j} + s_{j,i}) - s_{j,j}\Big) \\ &= 2\Gamma - trace(S) = \Gamma + \overline{trace}(S).\end{aligned} \tag{13}$$

Note that $trace(S)$ is the sum of the traffic *inside* the cluster while $\overline{trace}(S)$ is the sum of the traffic *between* the clusters (i.e., $\overline{trace}(S) = \sum_{i=1}^{P}\sum_{j=1}^{P} F_{i,j}$).

1) A Trivial Lower Bound: Suppose that a given network has been clustered into P clusters. The network and clusters are given by A and S matrices, respectively. Assume that $\overline{trace}(S) = 0$. It follows that $trace(S) = \Gamma$, which means that all the traffic is local traffic inside the clusters. No traffic is traveling between clusters. This case bounds from below the total traffic in the network. Hence,

$$\gamma_{\min} = \min_S \gamma = \min_S (2\Gamma - trace(S)) \ge \Gamma. \tag{14}$$

The network delay is given in (3). Assume first that the bridge has no delay; thus $\forall i, j B_{i,j} = 0$. Then (3) is reduced to

$$D = \frac{1}{\Gamma}\sum_{k=1}^{P} \frac{L_k}{C_k - L_k}. \tag{15}$$

Consider the case in which all the clusters have the same capacity. Thus, $C_k = C, 1 \le k \le P$. For fixed traffic demand γ where $\gamma = \sum_{k=1}^{P} L_k$, it is clear (and can be proved with Lagrange multipliers) that $\min D(\gamma)$ would be obtained when the traffic γ is distributed equally between all the clusters.

Thus $\forall k, L_k = \gamma/P$. The delay then is bounded by

$$D(\gamma) \geq \frac{1}{\Gamma}\left[\sum_{k=1}^{P} \frac{\gamma/P}{C_k - \gamma/P}\right] = \frac{P}{\Gamma}\frac{\frac{\gamma}{CP}}{1-\frac{\gamma}{CP}} = \frac{\gamma}{\Gamma C}\frac{1}{1-\frac{\gamma}{CP}}. \tag{16}$$

The feasible minimum of $D(\gamma)$ is obtained for $\min\gamma$. Recall from (14) that $\gamma_{\min} \geq \Gamma$, provides the "trivial lower bound" on the average delay of the network

$$D_{tr} = \frac{1}{C}\frac{1}{1-\frac{\Gamma}{CP}}. \tag{17}$$

2) A Lower Bound on the Total Traffic in the Network: As we have seen, a lower bound on the network delay can be found when $\min\gamma$ is given. In this section, a tighter lower bound on γ is derived.

The connection between γ and the cluster traffic matrix S is given in (13)

$$\gamma = 2\Gamma - trace(S) = \Gamma + \overline{trace}(S)$$

minimizing $\overline{trace}(S)$ is equivalent to minimizing γ. Hoffman and Donath [16] have shown that for any clustering of N users network into P clusters

$$\overline{trace}(S) \geq -\sum_{i=1}^{P} m_i\lambda_i(A-U) \tag{18}$$

where

A	$N \times N$ real symmetric matrix.
$\lambda_i(M)$	Eigenvalues of M, such that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$.
m_i	Number of users in cluster i, such that $m_1 \geq m_2 \geq \ldots \geq m_P$; $\sum_{i=1}^{P} m_i = N$.
U	$N \times N$ diagonal matrix that satisfies the condition $trace(U) = \Gamma$.

Note that in [16] A is a graph adjacency symmetric matrix that contains only $\{0, 1\}$ values. Nevertheless, careful examination of [16] shows that the results of that paper, including (18), are valid for any real symmetric matrix. In order to use (18) the "user traffic matrix" A must be symmetric. If this is not the case, A should be replaced by a symmetric matrix A' in which $a'_{i,j} = a'_{j,i} = (a_{i,j} + a_{j,i})/2$. The replacement can take place since for the lower bound derivation, we are interesting in the total amount of traffic between pairs of users, while the direction of the traffic is with no importance.

Relation (18) holds for any U satisfying $trace(U) = \Gamma$. In particular, U such that $\forall i \neq j : u_{i,j} = 0$; $u_{i,i} = \sum_{j=1}^{N} a_{i,j}$, which means that the sum of each row in $(A - U)$ is zero, satisfies that condition. When this particular U is chosen, the eigenvalues $\lambda_i(A-U)$ are $0 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N$. In order to obtain $\min_m \sum_{i=1}^{P} -\lambda_i m_i$, it is desired to choose m_1 as large as possible, then m_2 as large as possible and so on. An algorithm of finding a lower bound on the quantity $\min_m \sum_{i=1}^{P} -\lambda_i m_i$ is described in Appendix A.

3) Lower Bound on the Network Average Delay: Let $\alpha = \min_m \sum_{i=1}^{P} -\lambda_i(A-U)m_i$ be the lower bound on the traffic between clusters. Then, the lower bound on the total traffic is obtained by (13) and (18)

$$\min_S \gamma = \Gamma + \min_S \overline{trace}(S) \geq \Gamma + \alpha. \tag{19}$$

Let us assume again that the traffic is equally distributed among the different clusters, so $\forall k, L_k = \gamma/P$, and that all clusters have the same capacity C. Also assume that all the bridges have the same delay, so $\forall i, j : B_{i,j} = B$. Then

$$\begin{aligned} D &= \frac{1}{\Gamma}\left[\sum_{k=1}^{P}\frac{L_k}{C_k - L_k} + \sum_{j=1}^{P}\sum_{j=1}^{P} F_{i,j}B_{i,j}\right] \qquad (20)\\ &\geq \frac{1}{\Gamma}\left[P\frac{\gamma/P}{C-\gamma/P} + B\cdot\overline{trace}(S)\right]\\ &\geq \frac{P}{\Gamma}\frac{\frac{\Gamma+\alpha}{PC}}{1-\frac{\Gamma+\alpha}{PC}} + \frac{B}{\Gamma}\alpha. \end{aligned}$$

Thus, the improved lower bound on the network delay is

$$D_{\text{bound}} = \frac{P}{\Gamma}\frac{\frac{\Gamma+\alpha}{PC}}{1-\frac{\Gamma+\alpha}{PC}} + \frac{B}{\Gamma}\alpha. \tag{21}$$

The lower bounds for the nonequal capacities case can be found in Appendix B.

Note that the quality of the lower bound on the network average delay depends strongly on the lower bound on the traffic between clusters (α). As we demonstrate in Section VI-A5, there are examples in which our lower bound is quite loose.

IV. Problem Definition

The topology design problem, which is the problem of finding the minimum-delay LAN topology, can be formulated as follows: Given a user traffic matrix A, find a clustering matrix R, and a set of decision parameters $\{x_{i,j}^k, y_{i,j}^{k,l}\}$ such that

$$\text{minimize } D = \frac{1}{\Gamma}\left[\sum_{k=1}^{P}\frac{L_k}{C_k - L_k} + \sum_{i=1}^{P}\sum_{j=1}^{P} F_{i,j}B_{i,j}\right]$$

subject to

$$S = R^T A R$$

$$\sum_{j=1}^{P} r_{i,j} = 1, \quad 1 \leq i \leq N$$

$$r_{i,j} \in \{0, 1\}, \quad 1 \leq i, j \leq N$$

$$\{x_{i,j}^k, y_{i,j}^{k,l} | 1 \leq i, j, k, l \leq P\} \quad \text{form a spanning tree}$$

$$L_k < C_k, \quad 1 \leq k \leq P.$$

1) Solution Space Size: The topological design problem is a difficult combinatorial optimization problem. There are $\sum_{i=0}^{P}(-1)^i\frac{(P-i)^N}{i!(P-i)!}$ different variations of clustering N users into P clusters so that no cluster is empty [17], and P^{P-2} different variations of inter-cluster spanning trees [18]. Thus, the solution space size of the optimization problem is $\sum_{P=1}^{N} P^{P-2}\sum_{i=0}^{P}(-1)^i\frac{(P-i)^N}{i!(P-i)!}$.

The clustering problem itself is harder from the known "graph partition" problem which is classified in [9] as NP-complete. As mentioned in [7], a simplified version of the routing problem, namely the capacitate spanning tree problem [9], is also NP-complete. Thus, we can classify our problem as NP-complete. The huge size of the solution space makes the exhaustive search for the best solution impractical, even for medium size problems. Therefore, heuristic algorithms are applied seeking for the global minimum, though those heuristics do not guarantee finding that global minima at all. The heuristic algorithm solutions can be evaluated comparing to a known lower bound.

2) Heuristic Approaches for the Solution of the Topological Design Problem: In [6], it was pointed out that in large LAN's traffic tends to be local. We might be able to identify several groups of users which communicate almost exclusively with each other. That phenomena is called "locality of traffic." Based on this observation, the following approach is suggested: First, find a clustering which minimize the traffic locality index. Then, assign the clusters to LAN segments and find a routing which maximize the traffic balance index.

It can be noticed that the lower bounds presented in Section III have been found according to a similar approach. Bounds of the locality of the traffic were found, resulting with bounds on the total traffic in the network. Then, a bound on the average minimum delay in the network is obtained when that traffic is distributed keeping on perfect balancing between clusters. Unfortunately, both low traffic balance index and high traffic locality index may contradict one another. So, when we try to apply this suggested approach, we find that we can not optimize simultaneously both indexes. Moreover, for achieving minimum average delay the load in each LAN segment should be minimized, while in this approach the segments can be loaded up to their capacity. Also note that the load in the LAN segments is effected by both clustering decisions and routing decisions. Recall that we try to minimize the network average delay under all those considerations. It seems that there is no point in solving the clustering and the routing problems separately. There is no way but to solve the topology design problem as the comprehensive optimization problem. In this paper, we present such a comprehensive approach for solving the topological design optimization problem. This approach is based on the GA heuristic as explained in the next section.

V. Genetic Algorithms

Genetic algorithms [10], [11] are "search procedures" based on the mechanics of natural selection and natural genetics. The algorithm operates on a set of offered solutions called "populations." Each solution, called an "individual," can be any solution in the solution space represented by a string called "chromosome." Different solutions will be coded into different chromosomes values. The solution space is searched in order to find the optimum for a "target function." Each individual which represents a solution is evaluated by the target function. The resulting value is referred as the "fitness value" of the solution.

The current population is evolved creating a new generation with higher fitness. The evolution is done using three operators.

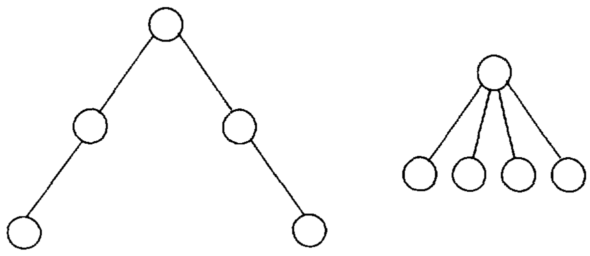


Fig. 1. Line and star configurations for $P = 5$. The line configuration depth is $\lfloor 5/2 \rfloor = 2$ since two branches are used.

1) *Reproduction:* This operator selects one individual from the current generation to the next generation. The probability of an individual to be selected is proportional to its fitness value. This operator is an artificial version of natural selection, i.e., the survival of the fittest solutions.
2) *Crossover:* Two individuals are mated in order to exchange genetic information. The chromosomes which represent the two solutions are broken at the same (random) place and combined together creating two new individuals.
3) *Mutation:* Mutation is a random change in the chromosome. One bit in the chromosome string is toggled. The original and the mutate individuals represent different solutions. The mutations operator gives the GA an opportunity to search in new corners of the solution space.

From the current population, we evolve a new population using those three operators. Individuals for the new generation are reproduced. Then the individuals are using the crossover operator to switch information. Mutations are randomly inserted in the new population. The fitness values of the new individuals are evaluated by the target function. The new generation replaces the current generation. From that point the algorithm can be repeated. The algorithm stops after a fixed number of generations, or when a chosen criterion, such as the best individual objective function value reaches some threshold.

1) Customizing the Topology Design Problem for GA's: As mentioned earlier, bridges connect the LAN clusters in a spanning tree configuration. Any spanning tree configuration is a valid network configuration, starting with a star configuration, ending with a line configuration, including all the configurations in between (see Fig. 1). The LAN network is assumed to consist of up to P clusters. Therefore, the spanning tree has up to P nodes. In fact, any desired spanning tree configuration can be obtained when a tree has the following two attributes: 1) Each node can have up to $P - 1$ sons, and 2) the tree depth can reach $\lfloor P/2 \rfloor$.

The network spanning tree must be represented by an appropriate data structure. Genetic operators, applied on such a structure representing a valid tree, must yield a valid tree. The data structure chosen for the representation of the spanning tree structure is based on a sparse Huffman tree.

Any node in a Huffman tree is associated with its unique label [19]. A node's label represents the path from the root to that node, where the length of the label is equal to the length

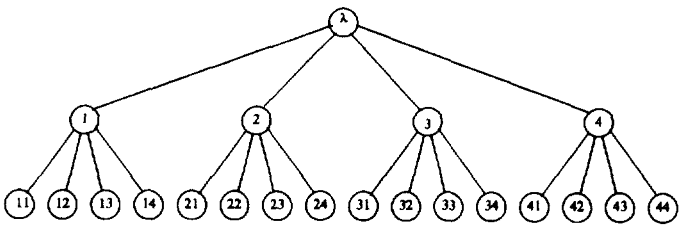


Fig. 2. Virtual Huffman tree where $P = 5$.

of that path. This path length is also called the node depth. The depth of the root is zero. The symbol λ stands for the null (zero length) label of the root. In Fig. 2 any intermediate node has four branches. The branches are associated with the characters $1, 2, 3, 4$ from the left to the right, respectively. So, in this example, the path from the root to the second from the right leaf is on branches 4 and 3; therefore, the label of that node is 43.

Consider a complete tree of depth $\lfloor P/2 \rfloor$ in which any node has $P - 1$ sons, and label it as an Huffman tree (Fig. 2). Summing the number of nodes in this tree gives a total of $\frac{(P-1)^{\lfloor P/2 \rfloor + 1} - 1}{P-2}$ nodes. This tree will be referred to as the "virtual Huffman tree." On this tree, any node can be identified by its label. Since the length of the label is equal to its depth, the longest label has $\lfloor P/2 \rfloor$ letters. Labeling of such a tree requires $P-1$ different letters. Consider the following example for $P = 5$. The depth of the virtual tree is two, and each node has four sons. Star topology can be obtained when the five label set$\{\lambda, 1, 2, 3, 4\}$ is chosen. Line topology can be obtained when the five label set $\{\lambda, 1, 11, 2, 23\}$ is chosen. Suppose that the five label set $\{\lambda, 11, 2, 32, 34\}$ is chosen (Fig. 3). The node labeled 11 is not connected to the root since the node labeled 1 hasn't been chosen. In order to get the right connectivity, the label 11 is forwarded toward the root. The least significant letter is omitted from the label 11 to form the label 1. Label 32 is changed in the same manner to 3, as where label 34 is untouched. Thus a fixed set of labels $\{\lambda, 1, 2, 3, 34\}$, replaces the original set. This fixed set represents a valid tree configuration.

In order to represent this virtual Huffman tree with the GA's chromosomes, it is required that all node labels will have the same lengths. For that purpose, an additional letter, called the "dummy-letter," is added to the alphabet. Dummy-letters are appended to the end of short node labels, enabling representation of any node label in a fixed length. The dummy-letter will be coded by the digit "0."

Using the virtual Huffman tree, any LAN topology configuration can be represented. A set of P virtual Huffman tree labels are selected randomly. This set of selected labels is fixed to form a valid tree. The fixing process is the same process described above. To ensure that in the obtained tree all the nodes have a path to the root, the set of the labels must obey to the following rule: Each prefix of a label in the set must also be a label in this set. A label in which one of its prefixes is not in the set is replaced by the missing prefix. For example, the prefixes of the label 2121 are 2120, 2100, and 2000. If the label 2121 is the only label in the set starting with 2, then it is replaced by the label 2000.

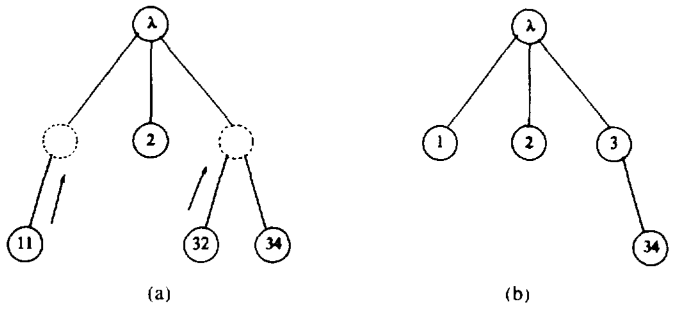


Fig. 3. Fixing the virtual Huffman tree. (a) The selected nodes and (b) the fixed tree.

The resulting LAN spanning tree configuration is described in a "configuration chromosome." This chromosome contains the P selected labels, of $\lfloor P/2 \rfloor$ letters each. Consider again the example in Fig. 3. The chosen five label set $\{\lambda, 11, 2, 32, 34\}$ is expressed in the chromosome 0 011 203 234. The value of this chromosome is fixed to 0 010 203 034 obtaining the set $\{\lambda, 1, 2, 3, 34\}$.

Given a LAN spanning tree configuration, $P!$ permutations of the P clusters can yield different LAN topologies. Therefore, the LAN configuration will be completely defined when each of the spanning tree nodes is matched to one of the P clusters. The order in which the clusters are assigned is expressed in the second chromosome.

The third chromosome describes the distribution of users into the clusters. This chromosome is treated as an array in which in the l^{th} place we find the cluster number of user l.

To summarize, any individual which is a solution for the topology design problem is represented by three chromosomes: C_{confi}, the configuration chromosome that describes the spanning tree configuration; C_{clust}, the clustering chromosome that describes the distribution of users into the clusters; C_{order}, the cluster order chromosome that describes the order of cluster assignment on the tree. The topology chromosome will be represented by the triplet $\{C_{\text{confi}}; C_{\text{clust}}; C_{\text{order}}\}$.

Genetic operators have to be applied on these chromosomes. After applying crossover and mutation operators on the configuration chromosome, it is possible that the resulting chromosome will not represent a valid tree unless it is fixed. The chromosome repairing process is as follows. The uniqueness of the labels in the chromosome is checked. If a label has multiple instances in the chromosome, all but one of those instances are changed. Then, each label is checked to have a father. A label which has no father is replaced by the missing father's label.

The Huffman tree structure was selected to represent the LAN configuration mainly since it significantly simplifies the problem of finding a path on the tree. Given a pair of clusters identified by their Huffman labels, we are interested in the path connecting them. This is done by removing the least significant bit from both the labels until we obtain their longest common prefix. The intermediate labels, together with the two original labels, define the desired path.

Since the clustering of the users is given, and the path between each pair of clusters can be found, the topology decision variables in (4) and (5) can be found. Therefore, the

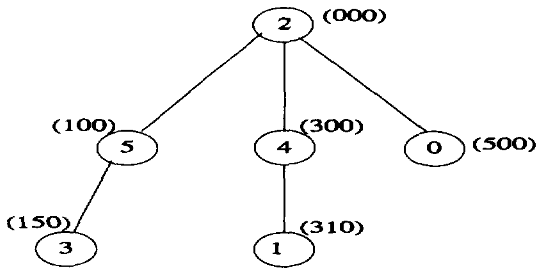


Fig. 4. The spanning tree represented by the Configuration chromosome 000 100 150 300 310 500, and the clusters order chromosome 253 410.

total traffic at each cluster (7) and the total traffic through any bridge (9) can be found. Putting those values in (3) gives us the average delay of the solution represented in the three chromosome individuals.

The average delay of a solution is used to set its fitness value. Since the lower the delay the better the solution is, the fitness value is taken as fitness value = 1/average delay.

Let us clarify the use of the discussed data structure with an example. Consider the user traffic matrix A given in Example 3 of Appendix C. An individual, which is a suggested solution to the topology design problem, is given by the following chromosomes:

$$C_{\text{confi}} = \underbrace{123\,450\,123\,450\,111\,111\,112\,222\,233\,333}_{\text{Clustering chromosome}}$$

$$C_{\text{clust}} = \underbrace{000\ 100\ 150\ 300\ 310\ 500}_{\text{Configuration chromosome}}$$

$$C_{\text{order}} = \underbrace{253\,410}_{\text{Clusters order chromosome}}.$$

The interpretation of the clustering chromosomes is as follows:

$$\begin{aligned}&\text{users}\{0,6,12\ldots19\} \in \text{cluster } 1, \text{ labeled with } 310.\\&\text{users}\{1,7,20\ldots24\} \in \text{cluster } 2, \text{ labeled with } 000.\\&\text{users}\{2,8,25\ldots29\} \in \text{cluster } 3, \text{ labeled with } 150.\\&\text{users}\{3,9\} \in \text{cluster } 4, \text{ labeled with } 300.\\&\text{users}\{4,10\} \in \text{cluster } 5, \text{ labeled with } 100.\\&\text{users}\{5,11\} \in \text{cluster } 0, \text{ labeled with } 500.\end{aligned}$$

The network spanning tree is shown in Fig. 4. The "cluster traffic matrix" obtained from this configuration is

$$S = \begin{pmatrix} 0 & 14 & 4 & 5 & 3 & 4 \\ 14 & 88 & 14 & 11 & 9 & 14 \\ 4 & 14 & 62 & 15 & 7 & 3 \\ 5 & 11 & 15 & 62 & 7 & 5 \\ 3 & 9 & 7 & 7 & 0 & 4 \\ 4 & 14 & 3 & 5 & 4 & 0 \end{pmatrix}.$$

The average delay of this individual solution is 0.129 956, when the clusters capacities are $C_k = 300$ and the bridge delay is $B = 0.1$ (all quantities units are given in next section).

VI. Numerical Examples

This section introduces some examples of topological design problems. These problems are solved using the GA described in the previous section. The best individual found by the GA is considered as the suggested solution to the problem. Lower bounds on the average delay of each example are found and compared with the GA suggested solution. These examples (excluding Example 4) are solved for the case of equal cluster capacities and equal bridges delays (this is common in many existing networks). The parameters for the GA are as follows: The mutation probability is 0.04, the crossover probability is 0.8, C is the cluster capacity in bits per second, B is the bridge delay in seconds per bit, P is the maximal number of clusters in the topology, N is the number of users, A is the user traffic matrix, and S is the cluster traffic matrix. The quantities $a_{i,j}$ and $s_{i,j}$ are given in bits per second. The algorithm terminates after a predefined number of generations. Examples 1 and 2 terminate after 50 generations and Example 3 and 4 terminate after 200 generations.

A. Examples, Set I

This set contains simple examples of LAN designing. The examples are simple enough so that the optimal solution (global minimum) can be found. The user traffic matrix of the examples in this set are given in Appendix C.

1) Example 1: In this example $N = 8$, $P = 4$, $C = 50$, $B = 0.1$.

Under these conditions, the GA has found the configuration given by the individual $\{22\,223\,333; 00102021; 0123\}$, with the average delay of $D_{ga} = 0.1933$. The resulting clusters traffic matrix is

$$S = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 36 & 4 \\ 0 & 0 & 4 & 36 \end{pmatrix}.$$

It can be seen that only two of the four clusters are being used. This reflects the implicit clustering in the users traffic matrix A. Both clusters are directly connected to each other.

For clustering the users into two clusters with capacity of $C = 50$, the trivial lower bound is $D_{tr} = 0.1$. The lower bound on the inter cluster traffic is $\alpha = 6$. The improved lower bound with $B = 0.1$ is $D_{\text{bound}} = 0.1611$.

Suppose now that cluster capacity is reduced to $C = 45$. In this new situation, the former clustering (into two clusters) can marginally supply the traffic demands. In this case, the GA yields a better solution which uses all four clusters $\{32\,230\,101; 00103033; 0132\}$, with the average delay of $D_{ga} = 0.2930$. The resulting cluster traffic matrix is

$$S = \begin{pmatrix} 6 & 12 & 1 & 1 \\ 12 & 6 & 1 & 1 \\ 1 & 1 & 6 & 12 \\ 1 & 1 & 12 & 6 \end{pmatrix}.$$

Obviously, the average delay of this configuration is higher by about 50% comparing to the former configuration.

On the other hand, when the cluster capacity is increased to $C = 200$, clustering all the users into one cluster is better than

TABLE I
GA RESULTS VERSUS LOWER BOUNDS

Ex	N	P_{max}	C	B	P_{ga}	D_{ga}	α	D_{tr}	D_{bound}	$D_{global\ min}$	$\frac{D_{ga}}{D_{bound}}$
1	8	4	50	0.1	2	0.1933	6	0.1	0.1611	0.1933	1.20
			45	0.1	4	0.293	8	0.04	0.0578	0.293	5.02
			200	0.1	1	0.0083	0	0.0083	0.0083	0.0083	1.00
2	10	4	50	0.1	2	0.0842	0	0.0333	0.0333	0.0842	2.53
			80	0.1	1	0.025	0	0.025	0.025	0.025	1.0
3	30	6	300	0.1	3	0.0329	30	0.0067	0.0143	0.0274	2.30

into two clusters since the dominant part in the delay is the bridge delay. The GA yielding this clustering resulted in an individual solution of $\{00\,000\,000; 00102030; 1203\}$, with an average delay of $D_{ga} = 0.0083$. The resulting cluster traffic matrix is

$$S = \begin{pmatrix} 80 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Note that for this example, a global minimum average delay solution has been found by the GA.

2) Example 2: This example shows a traffic which is equally distributed between all users, i.e., there is no implicit clustering. In this example $N = 10, P = 4, C = 50$, and $B = 0.1$.

The following individual solution has been found by the GA: $\{1\,100\,000\,111; 00101330; 1230\}$. The GA separates the users into two clusters with an average delay of $D_{ga} = 0.0842$. The resulting "cluster traffic matrix" is

$$S = \begin{pmatrix} 14 & 6 & 0 & 0 \\ 6 & 14 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

When the capacity has been increased to $C = 80$, the solution of the GA is clustering all the users into one cluster with the delay of $D_{ga} = 0.025$.

For clustering the users into two clusters with capacity of $C = 50$, the lower bound on the intercluster traffic is $\alpha = 0$. Both the trivial lower bound and the improved lower bound are $D_{tr} = D_{\text{bound}} = 0.0333$.

3) Example 3: In this example, $N = 30, P = 6, C = 300$, and $B = 0.1$. It can be seen that the user matrix is built of six groups of five users.

When the clusters capacity is $C_k = 300$, the resulting individual solution is

$$\{222\,223\,333\,311\,111\,111\,112\,222\,233\,333; \\ 000\,100\,150\,300\,310\,500; 140\,235\}$$

with the average delay of $D_{ga} = 0.0329$.

The resulting cluster traffic matrix is

$$S = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 130 & 10 & 10 & 0 & 0 \\ 0 & 10 & 120 & 20 & 0 & 0 \\ 0 & 10 & 20 & 120 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

This result is a local minima. The global minima is obtained when the clustering chromosome is 111111111122222222223333333333, with a delay of 0.0274. The local minima found by the GA is not so far from the global minima. In fact this solution identifies the clustering into five users groups.

For clustering the users into three clusters with capacity of $C = 300$, the trivial lower bound is $D_{tr} = 0.0067$. The lower bound on the inter cluster traffic is unchanged $\alpha = 30$. The improved lower bound is $D_{\text{bound}} = 0.0143$.

The above communication matrices have internal order, so that the grouping of the users can be observed easily. Obviously, the genetic algorithm does not need this ordering information. To show this, the order of the users in the communication matrix from Example 3 was changed randomly. Unsurprisingly, the genetic algorithm finds a similar solution to the former one.

Table I summarizes the results of the GA comparing to the lower bounds. In this table P_{max} is the maximal number of clusters that can be used by the GA. The lower bounds D_{tr}, D_{bound} were calculated for P_{ga}, the actual number of clusters that have been used in the GA solution. From this table we can observe that the lower bound and the GA results are of the same order.

4) Example 4, Unequal Capacities Case: The example was solved when the clusters had different capacities. The clusters capacities were: $\{C_0 = 400, C_1 = 350, C_2 = 340, C_3 = 330, C_4 = 300, C_5 = 100\}$. When the GA deals with different capacities, those capacities are associated with the tree configuration chromosome so that C_0 is associated with the first node label (the root), C_1 is associated with the second node label and so on.

In this representation, the cluster order chromosome which determines the matching between the labels and the clusters, determines the cluster capacities as well. The GA solution in this case is

$$\{000\,003\,333\,322\,222\,111\,112\,222\,233\,333; \\ 000\,100\,200\,300\,400\,500; 302\,145\}.$$

The resulting cluster traffic matrix is

$$S = \begin{pmatrix} 180 & 0 & 0 & 0 & 0 & 0 \\ 0 & 80 & 0 & 0 & 0 & 0 \\ 0 & 0 & 160 & 0 & 0 & 0 \\ 0 & 0 & 0 & 200 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

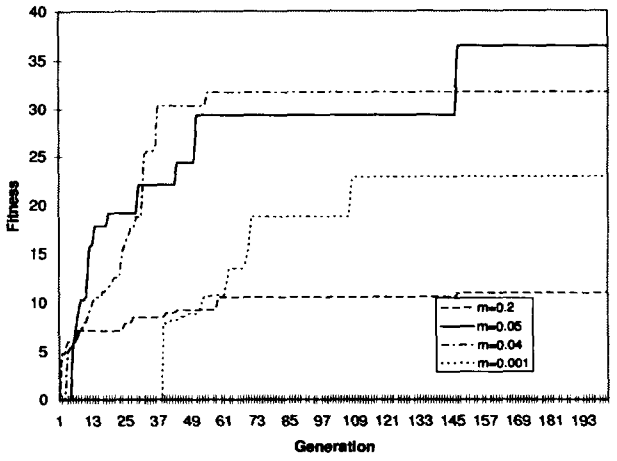


Fig. 5. Fitness evolution (m denotes the mutation probability).

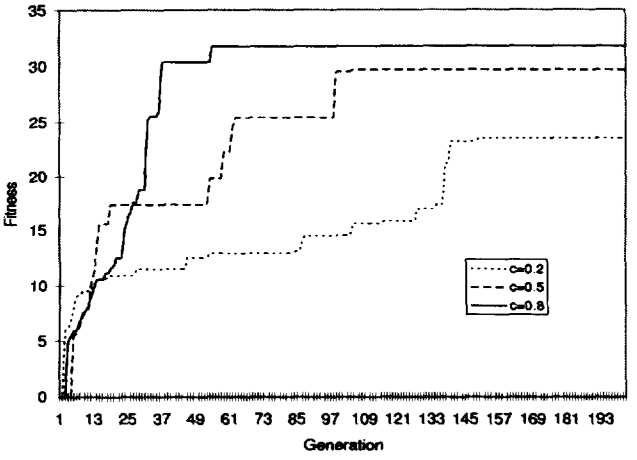


Fig. 6. Fitness evolution (c denotes the crossover probability).

The GA has assigned the cluster capacities according to their load. Cluster 3 gets the higher capacity (400), since its load is maximal (200). Cluster 0 gets the second higher capacity (350), since its load is 180, and so on. The average delay of the GA solution is 0.005 27. The lower bound on the average network delay has been calculated as described in Appendix B and its value is 0.004 438. The optimal traffic distribution between the clusters is $\{L_0 = 159.59, L_1 = 125.12, L_2 = 118.35, L_3 = 125.11, L_4 = 91.80, L_5 = 0\}$. This distribution is matching to the clusters capacities size. Intermediate result of the optimization process during the lower bound calculation shows that there is no point to use the smaller cluster, thus $L_5 = 0$. The ratio between the average delay of the GA solution and the lower bound is 1.187. In this example, as in the former ones, we can see that the GA solution was relatively close to the optimal solution.

5) Performance of the Algorithm: The performance of the algorithm is often evaluated by the speed of improving the fitness of its solution and by its complexity. The complexity of the algorithm that we have developed is $O(N^2+P^4log(P))$ for each individual of the population in any generation. In the following, we present the effect of the various parameters of the genetic algorithm using Example 3.

In Fig. 5, we present the effect of the mutation probability on the evolution of the fitness. We observe that when the mutation probability is small (0.001), the fitness starts to improve late (40th generation) and its value in the last generation is 22.8. When the mutation probability is large (0.2), the fitness starts to improve in the very first generations, but its value in the last generation is only 10.9. We found that, in general, a moderate value for the mutation probability (around 0.04–0.05) leads to the best performance. In this example, the mutation probability of 0.05 leads both to fast convergence and to the optimal fitness.

In Fig. 6, we present the effect of the crossover probability on the evolution of the fitness. We observe that the crossover probability should be kept high (around 0.8) in order to insure good performance of the algorithm.

For completeness, we include Fig. 7. It depicts the evolution of both the fitness of the best individual and the average fitness of the population. We observe the typical behavior of genetic algorithms, namely, the increasing trend of the average fitness of the population, with clear fluctuations between successive generations.

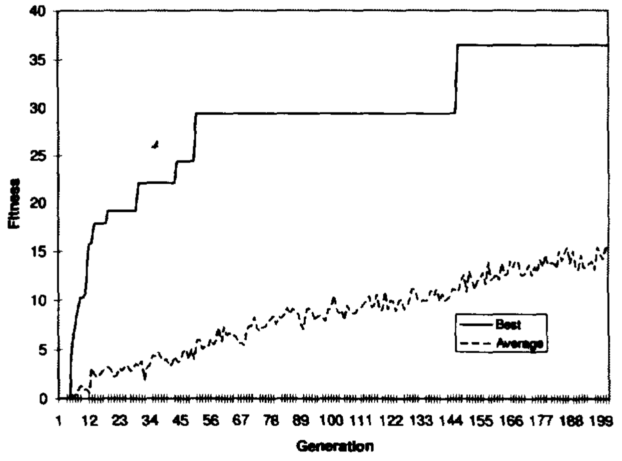


Fig. 7. Best and average fitness evolution.

TABLE II
SET II OF EXAMPLES: GA RESULTS VERSUS LOWER BOUNDS

Ex	Locality	P_{ga}	D_{ga}	D_{bound}	$\frac{D_{ga}}{D_{bound}}$
4	0.3	5	0.07038	0.0149	4.723
5	0.4	5	0.08006	0.0146	5.484
6	0.5	4	0.04639	0.0132	3.514
7	0.6	5	0.04572	0.0115	3.976
8	0.7	5	0.04176	0.0097	4.305
9	0.8	4	0.03598	0.0071	5.068

B. Examples, Set II

This set includes six examples of network design. The number of the users in the network is $N = 100$. The networks consists of ten groups of ten users each. The traffic between the users emulates a combination of four different traffic sources.

1) The first source emulates client–server relations between a user which is chosen as a server and the other users within its group.
2) The second source emulates client–server relations between a user which is chosen as a server and the other users in the network.

TABLE III
GA RESULTS VERSUS LOWER BOUNDS FOR VARIOUS VALUES OF BRIDGE DELAY

Locality	0.6	0.3
B	$\frac{D_{ga}}{D_{bound}}$	$\frac{D_{ga}}{D_{bound}}$
0.1	3.976	4.7
0.01	3.58	6.25
0.001	4.9	5.93
0.0001	2.4	4.79
0	2.72	4.92

TABLE IV
EE NETWORK: GA RESULTS VERSUS LOWER BOUNDS FOR VARIOUS VALUES OF BRIDGE DELAY

C	B	D_{ga}	D_{bound}	$\frac{D_{ga}}{D_{bound}}$
120	100.0	1048.7	13.45	77.97
	10.0	136.0	13.40	10.15
	1.0	41.3	13.397	3.08
	0.1	23.3	13.397	1.74
	0.0	19.78	13.396	1.48

TABLE V
EE NETWORK: GA RESULTS VERSUS LOWER BOUNDS FOR VARIOUS VALUES OF CLUSTER CAPACITIES

C	B	D_{ga}	D_{bound}	$\frac{D_{ga}}{D_{bound}}$
120	0.0	19.78	13.396	1.48
120	1.0	41.3	13.396	3.08
600	0.0	1.9173	1.8029	1.063
600	1.0	4.4014	1.8029	2.441
1,250	0.0	0.8985	0.8301	1.082
1,250	1.0	2.4190	0.8301	2.914

3) The third source emulates a session between a pair of users within a group.
4) The fourth source emulates a session between a pair of users in the network.

The following six examples combine traffic from all those four sources. The examples differ in their traffic locality index (see Section II-B). The total offered traffic in all these examples is $\Gamma = 20\,000$. The examples were checked for $P = 8$, $C = 15\,000$.

The task of finding the optimal solution of such large examples is not trivial at all. Therefore, the quality of the solution is evaluated against the lower bounds delay.

The delays of the GA solution and the lower bound for $B = 0.1$ are summarized in Table II. Table III shows the quality of the GA solution for various values of the bridge delay B.

In these Tables, we can see that the delays obtained from the GA solutions are on the same order as the lower bound. The ratio between the GA solution delay and the lower bound delay is 2.4–6.25. The traffic locality and the bridge delay have no clear effect on that ratio.

C. Examples, Set III

The last example is based on a traffic pattern which has been monitored at the Department of Electrical Engineering, Technion–Israel Institute of Technology Network. The monitoring has been done using several probes which were placed on Ethernet clusters at several laboratories and at the main department backbone. The number of monitored users was $N = 123$, and the total (average) monitored traffic volume was $\Gamma = 226\,770$ bytes/s.

The GA solutions were checked for various values of cluster capacities C and bridge delays B. For $C = 120$ KBytes/s when the bridge delay ranging between 0 s/byte and 0.1 s/byte, the delay results (in μs) of the GA and the lower bound are given in Table IV. Table V shows how the clusters capacity C affects the quality of the GA results, for two values of bridge delay ($B = 0$ s/byte and $B = 0.1$ s/byte).

The lower bound on the inter cluster traffic is $\alpha = 0.1256$ bytes/s. This number is smaller by several orders of magnitude from the inter cluster traffic obtained in the GA solutions (as one can observe from the following cluster traffic matrix). This difference grows as the bridge delay B increases. For $B = 0$ s/byte, the ratio between the delay resulted from the GA solution and the lower bound delay ranging between 1.063 and 1.48. That ratio remains under 3.08 as long as the bridge delay is smaller than $B = 0.001$ s/byte. When the bridge delay exceeds that value, the discussed ratio is increasing. For example, when $B = 0.1$ s/byte, we have that $\overline{trace}(S) = 2188$ while α remains as small as 0.1265. This indicates that there are situations in which the method described in Appendix A may lead to loose bounds on the traffic between clusters.

The resulting clusters traffic matrices for $C = 120$ Kbytes/s are

$$S_{B=0.0} = \begin{pmatrix} 10\,321 & 130 & 553 & 1061 & 80 \\ 365 & 36\,287 & 27 & 0 & 22 \\ 556 & 79 & 55\,094 & 0.1 & 2 \\ 3693 & 8 & 0.4 & 32\,475 & 0 \\ 172 & 9 & 4 & 0 & 85\,826 \end{pmatrix}$$

$$S_{B=0.001} = \begin{pmatrix} 18\,402 & 0 & 45 & 693 & 0.5 \\ 0 & 0 & 0 & 0 & 0.3 \\ 57 & 0 & 73\,006 & 581 & 2 \\ 1993 & 0 & 519 & 45\,354 & 102 \\ 0 & 0.2 & 12 & 172 & 85\,826 \end{pmatrix}$$

$$S_{B=0.1} = \begin{pmatrix} 18\,483 & 0.5 & 46 & 682 & 0 \\ 0 & 85\,921 & 21 & 108 & 0 \\ 67 & 7 & 73\,606 & 476 & 0 \\ 231 & 76 & 469 & 46\,572 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

For $C = 600$ Kbytes/s, $B = 0.01$, the resulting cluster traffic matrix is

$$S = \begin{pmatrix} 221 & 0 & 85 & 0 & 0 \\ 0 & 8 & 66 & 2 & 0 \\ 16 & 98 & 122\,546 & 120 & 0 \\ 0 & 0 & 140 & 103\,463 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

For C = 1,250 Kbytes/s, B = 0.0, the resulting cluster traffic matrix is

$$S = \begin{pmatrix} 55\,464 & 1 & 509 & 118 & 0 \\ 2 & 253 & 82 & 1 & 0 \\ 443 & 28 & 22\,855 & 957 & 0 \\ 93 & 5 & 2514 & 143\,439 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

It can be seen that in all those different cases, the GA finds a clustering so that the local traffic inside the clusters is higher at least by two orders from the inter cluster traffic. One can observe that when the clusters capacities increase, or when the bridges delay grow, only four out of the five available clusters were used. In these cases, the significance of the intercluster traffic grows, and the traffic is reduced by reducing the number of clusters.

An example of the resulting topology in the case when $C = 120\,000$, $B = 0.001$ is given in Fig. 8.

It is easy to see that the GA solution for that case is not the optimal solution. The union of clusters 1 and 4 is a simple move which can reduce the average delay in the network. It is very possible that the GA could have found this move in one of the next generations, but since the terminating criteria which was used is reaching predefined number of generations, the GA terminated before that move.

VII. Conclusion

The LAN topological design problem has been addressed. We described an heuristic based on genetic algorithm. This heuristic finds LAN configurations (i.e., partition of the users into clusters and routing between the clusters) with low average delay. This heuristic has been applied on several numerical examples. The average delay of the obtained designs has been compared with the network average delay lower bounds that were developed. The numerical examples give us a hint about the potential of GA's for solving complicated optimization problems in general, and in particular for solving the topological design problem.

Appendix A
Finding a Lower Bound on the Traffic Between Clusters

We need to find the solution for the following optimization problem:

$$\min \sum_{i=1}^{P} -\lambda_i m_i$$

subject to

$$\sum_{i=1}^{P} m_i = N$$

where $m_i \geq 0$ is the number of users in cluster i

$$L_p \leq C_p \; ; \; 1 \leq p \leq P$$

where A is the user traffic matrix, and $0 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N$ are the eigenvalues of $A - U$ introduced in Section III-B.

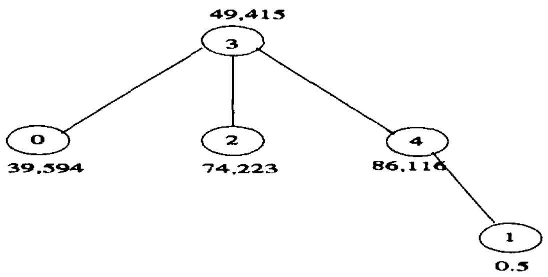


Fig. 8. Resulting configuration for $C = 120\,000$, $B = 0.001$.

Without the second constraint, which states that m_p is the number of users in cluster p, the solution is simple: $m_1 = N$; $m_2 = \cdots = m_N = 0$. Since the second constraint exists, the minimal solution will be obtained when m_1 is taken to be as large as possible, then m_2 is taken to be as large as possible, and so on.

Let $\{m_p\}_1^N$ be a set of natural numbers, so that m_p is a bound on the number of users in cluster p, as discussed in the former paragraph. Let $l_i = \sum_{j=1}^{N} a_{i,j} \quad 1 \leq i \leq N$, and let $\{q_k\}_1^N$ be the sorted set of $\{l_i\}_1^N$ so that $q_k = l_i$; $q_k \leq q_{k+1}$. Define

$$\text{arg\_index}(k) = \{i \;\text{ where }\; l_i = q_k\}.$$

Also, define the $N \times N$ matrix G by

$$g_{i,j} = \begin{cases} 1, & \text{if } j = \text{arg\_index}(i) \\ 0, & \text{otherwise.} \end{cases}$$

Let Y be the matrix which resulted from the linear transformation $Y = GAG^T$. This matrix is an equivalent traffic matrix to A and differ from A only in the users ordering. The order of Y is according to the users traffic volume. The user whose traffic volume is the smallest will appear first, followed by the other users sorted by increasing traffic volume. Let

$$s_p = \sum_{i=1}^{p} m_i \quad ; \; 1 \leq p \leq N \; ; \; s_0 = 0.$$

Obviously, $m_j = s_j - s_{j-1} \;\; ; \; 1 \leq j \leq N$. Since only P clusters exist, let us define $C_{P+1} = C_{P+2} = \cdots = C_N = 0$. The traffic volume at cluster p in one hop network is

$$L_p = \sum_{i=s_{p-1}+1}^{s_p} \sum_{j=1}^{N} (y_{i,j} + y_{j,i}) - \sum_{i=s_{p-1}+1}^{s_p} \sum_{j=s_{p-1}+1}^{s_p} y_{i,j}.$$

Any cluster must satisfy $L_p \leq C_p$. Therefore, the group $\{m_p\}_1^N$ can be obtained from the following constraints:

$$\sum_{i=s_{p-1}+1}^{s_p} \sum_{j=1}^{N} (y_{i,j} + y_{j,i}) - \sum_{i=s_{p-1}+1}^{s_p} \sum_{j=s_{p-1}+1}^{s_p} y_{i,j} \leq C_p$$

$$\sum_{i=s_{p-1}+1}^{s_p+1} \sum_{j=1}^{N} (y_{i,j} + y_{j,i}) - \sum_{i=s_{p-1}+1}^{s_p+1} \sum_{j=s_{p-1}+1}^{s_p+1} y_{i,j} > C_p$$

$$1 \leq p \leq N.$$

The solution of the constraint yields the set $\{m_p\}_1^N$, where $m_1 \geq m_2 \geq \cdots \geq m_s : m_{s+1} = m_{s+2} = \cdots = m_n = 0$. This set gives a lower bound for the minimization problem solution.

Appendix B
Lower Bounds for the Nonequal Capacities Case

For a given total traffic γ, we have to find the cluster loads vector $\overline{L}^T = (L_1, L_2, \cdots, L_P)$ that solves the following optimization problem:

$$\text{minimize } D(\overline{L}) = \frac{1}{\Gamma}\left[\sum_{i=1}^{P} \frac{L_i}{C_i - L_i}\right] \tag{22}$$

subject to

$$\sum_{i=1}^{P} L_i - \gamma = 0 \tag{23}$$

$$L_i \geq 0 \quad \forall 1 \leq i \leq P. \tag{24}$$

$D(\overline{L})$ is sum of convex function; therefore it is convex. Let

$$\varphi(\overline{L}) = \begin{pmatrix} [\sum_{i=1}^{P} L_i - \gamma] \\ -[\sum_{i=1}^{P} L_i - \gamma] \\ -L_1 \\ -L_2 \\ \vdots \\ -L_P \end{pmatrix} \quad \overline{x} = \begin{pmatrix} z_1 \\ z_2 \\ y_1 \\ y_2 \\ \vdots \\ y_P \end{pmatrix}.$$

Equations (23) and (24) can be replaced by $\varphi(\overline{L}) \leq 0$. The Kuhn–Tucker (KKT) condition for optimal solution states that for such an optimal solution there exists $\overline{x} \geq 0$ such that

$$x_i \varphi_i(\overline{L}) = 0 \quad \forall 1 \leq i \leq P + 2 \tag{25}$$

$$\nabla D(\overline{L}) + \sum_{i=1}^{P+2} \overline{x} \nabla \varphi(\overline{L}) = 0. \tag{26}$$

Define $z = z_1 - z_2$; then, the KKT condition yields the following set of equations:

$$\frac{1}{\Gamma}\frac{C_i}{(C_i - L_i)^2} + z - y_i = 0 \quad \forall 1 \leq i \leq P \tag{27}$$

$$L_i y_i = 0 \quad \forall 1 \leq i \leq P\,; \quad z \in R. \tag{28}$$

Solving this quadric equation for L_i under the constraint that $L_i < C_i$, gives

$$L_i = C_i - \sqrt{\frac{-C_i}{\Gamma(z - y_i)}}. \tag{29}$$

Putting these solutions into (23) gives

$$\sum_{i=1}^{P}\left[C_i - \sqrt{\frac{-C_i}{\Gamma(z - y_i)}}\right] - \gamma = 0. \tag{30}$$

From (29) and (28), we can conclude that

$$y_i = \begin{cases} 0, & \text{if } L_i \neq 0 \\ \frac{1}{\Gamma C_i} + z, & \text{if } L_i = 0. \end{cases} \tag{31}$$

It is not clear yet whether $L_k = 0$ for some k. Let us observe at two clusters k, l with capacities C_k, C_l. Assume that $L_k = 0$ and $L_l \neq 0$. This case can be optimal only if

$$\left.\frac{\partial D(\overline{L})}{\partial L_k}\right|_{L_k=0} > \frac{\partial D(\overline{L})}{\partial L_l} \tag{32}$$

which yields

$$\frac{1}{\Gamma}\frac{C_k}{(C_k - 0)^2} > \frac{1}{\Gamma}\frac{C_l}{(C_l - L_l)^2}. \tag{33}$$

Inserting L_l from (29) gives

$$\frac{1}{C_k} > -\Gamma z. \tag{34}$$

Since (34) is not dependent at C_l or L_l, we can conclude that any cluster k satisfies condition (34) must have $L_k = 0$.

Assume now that $L_i > 0 \;\; \forall 1 \leq i \leq P$, then it follows that $y_i = 0 \;\; \forall 1 \leq i \leq P$ and (30) becomes one variable equation

$$\sum_{i=1}^{P}\left[C_i - \sqrt{\frac{-C_i}{\Gamma z}}\right] - \gamma = 0. \tag{35}$$

Let z be the solution of (35); then the assumption that $L_i > 0$ must be checked for any $i = 1, \cdots, P$. If condition (34) is not satisfied for any cluster, then $\overline{L}$ obtained by (29) are the solution for the optimization problem. Otherwise, the load at any cluster satisfies condition (34) is set to zero and new value for z is obtained from (30). The process of checking L_i and resolving (30) is repeated until no cluster satisfies condition (34).

After $\overline{L}$ is known, the average delay lower bounds can be found. For the trivial lower bound $\gamma = \Gamma$, and for the improved lower bound $\gamma = \Gamma + \alpha$. Putting the resulting traffic values $\overline{L}$ in (21) gives us the lower bound for the nonequal capacities case.

Appendix C
Example Matrices

In this Appendix, the users traffic matrix of the examples illustrated in Section Vi are given. Note that for the ease of the reader, $0's$ were replaced by "$\cdot$".

1) Example 1:

$$A = \begin{pmatrix} \cdot & 3 & 3 & 3 & \cdot & \cdot & \cdot & 1 \\ 3 & \cdot & 3 & 3 & \cdot & \cdot & 1 & \cdot \\ 3 & 3 & \cdot & 3 & \cdot & 1 & \cdot & \cdot \\ 3 & 3 & 3 & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & 3 & 3 & 3 \\ \cdot & \cdot & 1 & \cdot & 3 & \cdot & 3 & 3 \\ \cdot & 1 & \cdot & \cdot & 3 & 3 & \cdot & 3 \\ 1 & \cdot & \cdot & \cdot & 3 & 3 & 3 & \cdot \end{pmatrix}.$$

2) Example 2:

$$A = \begin{pmatrix} \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 \\ 1 & \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & 1 \\ 1 & 1 & \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & 1 & \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & 1 & \cdot & 1 & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & 1 & \cdot & 1 & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & 1 & \cdot & 1 & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & \cdot & 1 & 1 \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & \cdot & 1 \\ 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & \cdot \end{pmatrix}.$$

3) Example 3:

$$A = \begin{pmatrix} A_3 & B & 0 & B & 0 & B \\ B & A_3 & B & 0 & B & 0 \\ 0 & B & A_3 & B & 0 & B \\ B & 0 & B & A_3 & B & 0 \\ 0 & B & 0 & B & A_3 & B \\ B & 0 & B & 0 & B & A_3 \end{pmatrix}$$

where 0 is the zero matrix and

$$A_3 = \begin{pmatrix} \cdot & 3 & 3 & 3 & 3 \\ 3 & \cdot & 3 & 3 & 3 \\ 3 & 3 & \cdot & 3 & 3 \\ 3 & 3 & 3 & \cdot & 3 \\ 3 & 3 & 3 & 3 & \cdot \end{pmatrix}; B = \begin{pmatrix} \cdot & \cdot & \cdot & \cdot & 1 \\ \cdot & \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot \end{pmatrix}.$$

4) Example 4:

$$A = \begin{pmatrix} A_9 & 0 & 0 & 0 & 0 & 0 \\ 0 & A_7 & 0 & 0 & 0 & 0 \\ 0 & 0 & A_5 & 0 & 0 & 0 \\ 0 & 0 & 0 & A_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & A_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & A_3 \end{pmatrix}$$

where 0 is the zero matrix and

$$A_i = \begin{pmatrix} \cdot & i & i & i & i \\ i & \cdot & i & i & i \\ i & i & \cdot & i & i \\ i & i & i & \cdot & i \\ i & i & i & i & \cdot \end{pmatrix}.$$

REFERENCES

[1] IEEE Project 802, "Local and metropolitan area networks," Standard 802.1 part D: MAC Bridges, in *IEEE Standards Press*, 1989.

[2] F. Backes, "Transparent bridges for interconnection of IEEE 802 LAN's," *IEEE Networks*, pp. 5-9, Jun. 1988.

[3] E. D. Sykas and G. L. Lyberopoulos, "Performance analysis of interconnected CSMA/CD local area networks," in *IEE PROC.-I*, Apr. 1992, pp. 181-197.

[4] W. W. Ho and B. Mukherjee, "A Multiple-Partition Token Ring Network," in *IEEE INFOCOM* '90, 1990, pp. 982-988.

[5] R. Zambre, "Design considerations for extended local area networks," in *ICCC'90*, 1990, pp. 432-442.

[6] K. M. Khalil and P. A. Spencer, "A systematic approach for planning, tuning and upgrading local area networks," in *GLOBECOM'91*, 1991, pp. 658-663.

[7] 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.

[8] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall 1992.

[9] M. R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: Freeman, 1979.

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

[11] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *IEEE Computer*, pp. 17-26, Jun. 1994.

[12] T. Routen, "Genetic algorithms and neural network approaches to local access network design," in *MASCOTS'94*, 1994.

[13] M. C. Sinclair, "The application of genetic algorithm to trunk network routing table optimization," in *Proc. 10th U.K. Teletraffic Symp.—Performance Eng. Telecommun. Networks*, 1993.

[14] W. D. Potter, R. Pittes, P. Gillis, J. Young, and J. Caramadre, "IDA-NET: an intelligent decision aid for battlefield communications network configuration," in *Proc. 8th Conf. Artificial Intelligence Applications*, 1992, pp. 247-253.

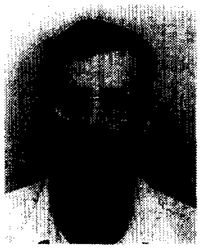
[15] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high speed networks," *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, pp. 968-981, Aug. 1991.

[16] A. J. Hoffman and W. E. Donath, "Lower Bounds for the Partitioning of Graphs," *IBM J. Res. Develop.*, pp. 420-425, 1973.

[17] C. L. Liu, *Introduction to Combinatorial Mathematics*. New York: McGraw-Hill, 1968.

[18] A. Nijenhuis, *Combinatorial Algorithms*. New York: Academic, 1975.

[19] S. Even, *Graph Algorithms*. Rockville, MD: Computer Science Press, 1979.



Reuven Elbaum was born in Tel-Aviv, Israel, on May 20, 1962. He received the B.Sc. and M.Sc. degrees from the Technion–Israel Institute of Technology, Haifa, Israel, both in electrical engineering, in 1988 and 1995, respectively.

From 1988 to 1992, he worked as a Research and Development Engineer at Intel Israel (74) LTD. From 1992 to 1995, he worked as a Teaching Assistant at the Technion, Israel. Currently, he is working as a Research and Development Engineer at PowerSpectrum Technology LTD, Israel. His interests include computer networks and computer architecture.

Moshe Sidi (S'77–M'82–SM'87) received the B.Sc., M.Sc., and the D.Sc. degrees from the Technion–Israel Institute of Technology, Haifa, Israel, in 1975, 1979, and 1982, respectively, all in electrical engineering.

In 1982, he joined the faculty of the Electrical Engineering Department at the Technion. During the academic year 1983–1984, he was a Post-Doctoral Associate at the Electrical Engineering and Computer Science Department of the Massachusetts Institute of Technology, Cambridge, MA. During 1986–1987, he was a Visiting Scientist at IBM, Thomas J. Watson Research Center, Yorktown Heights, NY. His research interests are in wireless networks and multiple-access protocols, traffic characterization and guaranteed grade of service in high-speed networks, queueing modeling and performance evaluation of computer communication networks. He co-authored the book *Multiple Access Protocols: Performance and Analysis* (New York: Springer Verlag, 1990). He served as the Editor for Communication Networks in the IEEE TRANSACTIONS ON COMMUNICATIONS from 1989 until 1993, and as the Associate Editor for Communication Networks and Computer Networks in the IEEE TRANSACTIONS ON INFORMATION THEORY from 1991 until 1994. Currently, he serves as an Editor in the IEEE/ACM TRANSACTIONS ON NETWORKING and as an Editor in the *Wireless Journal*.