A METHODOLOGY FOR NETWORK TOPOLOGY DESIGN USING FUZZY EVALUATIONS

Salman A. Khan[1] and Sadiq M. Sait[2]

1: Lecturer, Department of Computer Engineering, KFUPM, Saudi Arabia.
2: Professor, Department of Computer Engineering, KFUPM, Saudi Arabia.

P.O. Box 1652, KFUPM, Saudi Arabia, Fax: 9663 860-3059, Email: salmank@kfupm.edu.sa

ABSTRACT

The topology design of campus networks (CNs), a class of computer networks, is a NP-hard optimization problem. The design consists of three main steps and requires the optimization of several conflicting objectives such as minimization of cost, minimization of network delay, and minimization of maximum number of hops etc. Since some of the objectives are imprecise, fuzzy logic provides a suitable mathematical framework in such a situation. In this paper, we present a methodology to address design issues. This methodology is based on two algorithms, namely, fuzzy simulated evolution algorithm and the augmenting path algorithm. Test cases are used to evaluate the effectiveness of the methodology. Results suggest that the methodology is suitable to address the topology design problem.

Keywords: *Network Topology, Fuzzy Logic, Simulated Evolution, Combinatorial Optimization.*

الملخص

تصميم شبكات حرم الجامعة هي قسم من مشكلة تصميم الشبكات وهي مشكلة صعبة غير متعدد الحدود. يحتوي التصميم على ثلاث مراحل ويتطلب الإهتمام بأهداف متعددة، كالسعر و تأخر الشبكة وتقليل عدد القفزات، ألخ. وبما ان الأهداف غير دقيقة تم استخدام المنطق المبهم، وهو يوفر أداة رياضية مناسبة في مثل هذه الحالات. في هذه الورقة، نقدم طريقة للتصميم تحتوي على خوارزميان، الأول خوارزم محاكي الإرتقاء والثاني خوارزم الطريق المطول. تم استخدام أمثلة لفحص وتقييم فعالية الطريقة. النتائج اثبتت فعالية الطريقة المقترحة بمعالجة المشكلة.

1. INTRODUCTION

In a typical campus network, a number of components, such as mainframe computers, mini systems, workstations, PCs, user terminals, printers, etc. are connected together [Youssef et al, 1997]. Various devices such as switches, routers, hubs, etc., are used to interconnect and network these computers and peripherals. In the network topology design, several constraints need to be considered. Geographical constraints dictate the division of such internetworks into smaller parts or groups of nodes, where each group makes up what is called a LAN. Thus, we can define a campus network as a collection of interconnected LANs. Further, the nodes of a LAN may be subdivided into smaller parts, called *LAN segments*, to satisfy other constraints and objectives, for example, minimizing delay, cabling and equipment cost etc., [Youssef et al, 1997]. The topology design of LAN itself consists of two main issues: *segmentation*,

where formation of LAN segments takes place, and *design of actual topology*, which consists of interconnecting the individual segments. Topology design at LAN level requires interconnection of LAN segments via bridges and layer 2 switches [Elbaum, 1996][Gen et al, 1998]. Users (stations) are allocated to these segments based on some criteria. Only spanning tree topologies can be used as active LAN configurations [Elbaum, 1996] [Gen et al, 1998][Ersoy, 1993]. However, when the bridges support the spanning tree protocol, the actual physical topology does not have to be loop-free.

A good and structured campus network has three layers:

1. Local Access Layer, which provides workgroup access to the network.
2. Distribution Layer, which provides policy based connectivity among the workgroups. This layer is implemented with layer 3 switches, routers, and gateways. This is where packets manipulation takes place.
3. Backbone Layer, which provides high-speed optimal transport of data among local sites.

Thus, the design of such a structured campus network can be approached in three steps [Khan et al, 2002]:

1. Assignment of users/stations to LAN segments.
2. Design of the internal structure of each local site (i.e., in what topology the LAN segments of a local site are connected).
3. Backbone design, where the local sites are connected to the backbone.

Since the topological design of campus networks is a hard problem [Youssef et al, 1997], intelligent methods known as 'heuristics' are used to get near optimal solutions in reasonable amount of time. These heuristics are also effective for the design of a LAN, as this could also be classified as a NP-hard problem [Elbaum,1996][Gen et al, 1998][Ersoy, 1993]. There are two categories of heuristics: *constructive* and *iterative.* Some of the well known constructive algorithms for the constrained minimum spanning tree problem are Prim algorithm [Prim, 1957], and Esau-Williams algorithm [Esau, 1966].

Iterative heuristics attempt to improve a complete solution by making controlled stochastic moves [Sait, 1999]. The use of iterative heuristics for topological network design is reported in many research papers. The use of Genetic Algorithm (GA) for topological network design has been proposed in [Elbaum, 1996][Gen et al, 1998]. In [Elbaum,1996], Elbaum and Sidi have used GA based on Huffman tree to solve the topological design of LAN with a single criterion, which is to reduce the network delay. In [Gen et al, 1998], Gen et al. have used GA for topological network design with two criteria, which are the network delay and cost, based

on the weights of links. In [Kumar, 1995], Kumar et al. have developed a GA considering reliability to design computer networks. In [Altiparmak, 1997], Dengiz et al. focused on large backbone communication network design considering all-terminal network reliability and used a GA. Similarly; use of simulated annealing has been reported in [Ersoy, 1993] where Ersoy et al. used it for topological design of interconnected LAN/MAN. In this work, simulated evolution algorithm is proposed for topology design of structured campus networks based on several criteria, which are: monetary cost, maximum number of hops between any source-destination pair, and average network delay per packet. For assignment of segments to local sites, Augmenting Path algorithm is used. Since the backbone design and internal topology design are multi-objective combinatorial optimization problems, fuzzy logic is used to formulate the various objectives in the form of fuzzy rules that will guide the search algorithm toward solutions of desirable quality.

In Section 2, assumptions, terminology, and notation are given. Section 3 describes computation of objective values and constraints. Section 4 presents the proposed scheme. Section 5 describes the approach for assignment of LAN segments. In Section 6, internal topology design is discussed. We conclude in Section 7.

2. ASSUMPTIONS AND TERMINOLOGY

- The backbone is assumed to be running on Fast Ethernet using Fiber Optic. The Root node is a collapsed backbone with given Ethernet and Token Ring interfaces.
- Between two local sites, only fiber optic cable is used.
- Class C networks are assumed. Therefore, we limit the number of nodes per local site to at most 254.
- Maximum allowed utilization of a link is 60 %.

Notation

n	number of clusters
m	number of LAN segments in a cluster
T^b	n x n local site topology matrix where t_{ij}^b=1, if local sites i and j are connected and t_{ij}^b = 0 otherwise.
λ_i	traffic on link i.
$\lambda_{\max,i}$	capacity of link i.
D_{nd}	delay due to network devices.
g_i	maximum number of clusters which can be connected to cluster i.
γ_{ij}	external traffic between clusters i and j.
γ	overall external traffic

3. COST FUNCTION AND CONSTRAINTS

In this section, we present the computation of our objective values.

3.1. Monetary cost

The goal of monetary cost optimization is to find the topology with minimum possible cost, while meeting all the requirements and constraints. Since the cost of the cable and the cost of the network devices are the two main entities affecting the monetary cost, our function for monetary cost can mathematically be defined as:

$$\cos t = (s \times c_{cable}) + (c_{nd}) \qquad \textbf{(1)}$$

where s represents the total length of cable, c_{cable} represents the cost per unit of the cable used, c_{nd} and represents the combined costs of all the routers, switches, and hubs used.

3.2. Average Network Delay

Another important criterion to consider is the average network delay per packet. The goal, of course, is to minimize the delay as much as possible, while considering the constraints and requirements.

To devise a suitable function for average network delay, we have used the model presented in [Elbaum, 1996], where an M/M/1 model is used to approximate the behavior of a link and network device. The delay per bit due to network device between local site i and j is $B_{i,j} = \mu b_{i,j}$, where $b_{i,j}$ is the delay per packet. If $\gamma_{i,j}$ is the total traffic through the network device between local sites i and j, then the average delay due to all network devices is:

$$D_{nd} = \frac{1}{\gamma} \sum_{i=1}^{d} \sum_{j=1}^{d} \gamma_{ij} B_{ij} \qquad \textbf{(2)}$$

Thus, total average network delay is composed of delays of links and network devices and is given by [Elbaum, 1996]

$$D = \frac{1}{\gamma} \sum_{i=1}^{m} \frac{\lambda_i}{\lambda_{\max,i} - \lambda_i} + \frac{1}{\gamma} \sum_{i=1}^{n} \sum_{j=1}^{n} \gamma_{ij} B_{ij} \qquad \textbf{(3)}$$

3.3. Maximum number of hops between any source-destination pair

The maximum number of hops between any source-destination pair is also another objective to be optimized. A hop is counted as the packet moves from one network device to another.

3.4. Constraints

Three important constraints are considered in this paper. The first set of constraints is due to the limitation of the capacity of the links. A good network would be one in which links are "reasonably" utilized, otherwise this would cause delays, congestion, and packet loss. Thus the traffic flow on any link *i* must never exceed a threshold value:

$$\lambda_i < \lambda_{\max,i} \qquad i = 1,2,....,e \qquad \textbf{(4)}$$

The second constraint is that the number of clusters attached to a network device G must not be more than the port capacity of that device. Mathematically, we can represent this constraint as:

$$\sum_{j=1}^{n} t_{ij}^{b} < g_i \qquad i= 1, 2, \ldots, n \quad \forall i \neq j \qquad \textbf{(5)}$$

The third possible constraint is that the designer might like to enforce certain hierarchies on the network devices. For example, he or she might not allow a hub to be the parent of a router or a switch to be the parent of a router. This constraint is dependent on the choice of the designer.

4. FUZZY SIMULATED EVOLUTION ALGORITHM FOR TOPOLOGY DESIGN

4.1. Simulated Evolution

Simulated Evolution (SE) is a general iterative heuristic proposed in [Kling, 1990]. SE iteratively operates a sequence of evaluation, selection and allocation (perturbation) on one solution. It starts with a randomly or constructively generated valid initial solution. The main loop of the algorithm consists of three steps: **evaluation**, **selection** and **allocation**. These steps are carried out repetitively in a main loop until some stopping condition is satisfied. Other than these three steps, some input parameters for the algorithm are initialized in an earlier step known as **initialization**.

4.2. Proposed Algorithm and Implementation Details

This section describes our proposals of fuzzification of different stages of the SE algorithm. The description is combined with the implementation details of the SE algorithm for topology design. We confine ourselves to tree design because they are minimal and provide unique path between every pair of local site.

4.2.1 Initialization

The initial topology, which is a spanning tree, is generated randomly, while keeping into account the constraints mentioned earlier. Some parameters are also initialized in this phase. These include the maximum number of iterations for which the algorithm has to be run, and the selection bias ***B.***

4.2.2 Proposed Fuzzy Evaluation Scheme

The **goodness** of each individual is computed as follows. In our case, an individual is a **link** that interconnects two local sites (at the backbone level) or two network devices (at the local site level). In the *fuzzy evaluation scheme*, monetary cost and optimum depth of a link (with respect to the root) are considered fuzzy variables. Then the goodness of a link is characterized by the following rule.

Rule 1: IF a link is *near optimum cost* AND *near optimum depth* **THEN** it has high *goodness*.

Here, *near optimum cost*, *near optimum depth*, and *high goodness* are linguistic values for the fuzzy variables cost, depth, and goodness. Using orlike compensatory operator, Rule 1 translates to the following equation for the fuzzy goodness measure of a link l_i.

$$g_{l_i} = \mu^e(x) = \alpha^e \times \min(\mu_1^2(x), \mu_2^2(x)) + (1-\alpha^e) \times \frac{1}{2}\sum_{i=1}^{2} \mu_i^e(x) \qquad \textbf{(6)}$$

The superscript *e* stands for **evaluation** and is used to distinguish similar notation in other fuzzy rules. In equation 6, $\mu^e(x)$ is the fuzzy set of *high goodness links,* g_{li} is goodness value, and α^e is a constant. The $\mu_1^e(x)$ and $\mu_2^e(x)$ represent memberships in the fuzzy sets *near optimum monetary cost* and *near optimum depth*.

In order to find the membership of a link with respect to *near optimum monetary cost*, we proceed in the following manner. From the cost matrix, which gives the costs of each possible link, we find the minimum cost of all the costs and maximum cost of all the costs. We take these minimum and maximum costs as the lower and upper bounds and call them "LCostMin" and "LCostMax" respectively and then find the membership of a link with respect to these bounds. Furthermore, in this work, we have normalized the monetary cost with respect to "LCostMax". The required membership function is represented as depicted in Figure 1(a).

In the same manner, we can find the membership of a link with respect to *near optimum depth*. The lower limit, which we call "LDepthMin" is taken to be a depth of 1 with respect to the root. This is logical to take as the lower bound since there can be no link which is connected to the root at a depth of zero. The upper bound, which we call "LDepthMax" is taken to be 1.5 times of the maximum depth generated in the initial solution or a maximum of 7. For example, if in the initial solution, the maximum depth turns out to be 4, then

"LDepthMax" for the depth membership function would be 6. This is done to give flexibility to links, which may have more depth than the one in the initial solution. If we take the initial solution maximum depth as "LDepthMax", then in the following iterations some links with higher depths will have a membership value of zero (with respect to depth membership function) and thus they will not be able to play any role as far as depth is concerned. However, due to technological limitations, we have limited the maximum possible depth to 7, in the case when " LDepthMax" turns out to be more than 4.

The reason for having the maximum depth of 7 is that the hop limit for routing information protocol (RIP) is 15. This means that if a maximum depth of 7 were taken, then in the worst case we would have a total of 14 hops from a source to a destination. The membership function with respect to *near optimum depth* can be represented as illustrated in Figure 1(b).

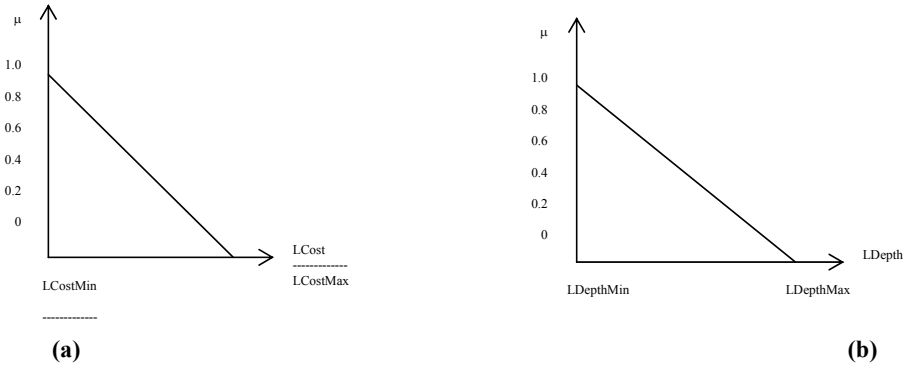

Figure 1: Membership function for (a) monetary cost of a link (b) depth of a link.

4.2.3 Selection

In this stage of the algorithm, for each link l_i, where i = 1,2,...,(n-1) currently present in the topology a random number in the range [0,1] is generated and compared with $g_{ci} + B$, where B is the selection bias. If the generated random number is greater than $g_{ci} + B$ then link l_i is selected for allocation and considered removed from the topology.

4.2.4 Proposed Fuzzy Allocation Scheme

During the **allocation** stage of the algorithm, the selected links are removed from the topology one at a time. For each removed link, new links are tried in such a way that they result in overall better solution. Before the allocation step starts, the selected links are sorted according to their goodness values, with the link with the worst goodness being the head-of-line in the queue.

In the *fuzzy allocation scheme*, the three criteria to be optimized are combined using fuzzy logic to characterize a good topology. The following rule and the subsequent equation are used for this purpose.

Rule 2: IF a solution X has *low monetary cost* AND *low average network delay* AND *low maximum number of hops between any source-destination pair* **THEN** it is a *good topology*.

$$\mu^a(x)_i = \beta^a \times \min(\mu_1^a(x), \mu_2^a(x), \mu_3^a(x)) + (1-\beta^a) \times \frac{1}{3}\sum_{i=1}^{3} \mu_i^a(x) \qquad \textbf{(7)}$$

where $\mu^a(x)$ is the membership value for solution x in the fuzzy set *good topology* and β^a is a constant in the range [0,1]. The superscript a stands for allocation. Here, $\mu_i^a(x)$ for $i = \{1,2,3\}$ represents the membership values of solution x in the fuzzy sets *low monetary cost*, *low average network delay*, and *low maximum number of hops between any source-destination pair* respectively. The solution that results in the maximum value for Equation 7 is reported as the best solution found by the SE algorithm.

Below we will see how to get the membership functions for the three criteria we have mentioned above.

4.2.5 Membership Function for Monetary Cost

First, we determine two extreme values for monetary cost, i.e., the minimum and maximum values. The minimum value, "TCostMin", is found by using the Esau-Williams algorithm, with all the constraints completely relaxed. This will surely give us the minimum possible monetary cost of the topology. The maximum value of monetary cost, "TCostMax", is taken to be the monetary cost generated in the initialization step. The monetary cost is normalized with respect to "TCostMax". The corresponding membership function is shown in Figure 2(a).

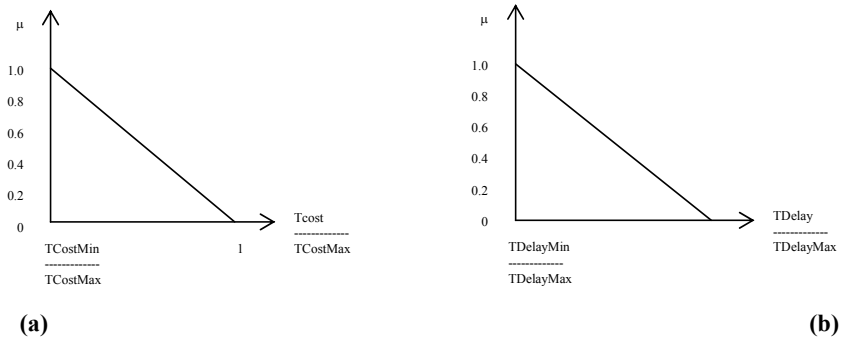

Figure 2: Membership function for (a) monetary cost (b) average network delay.

4.2.6 Membership Function for Average Network Delay

We determine two extreme values for average network delay. The minimum value, "TDelayMin", is found by connecting all the nodes to the root directly ignoring all the constraints and then calculating the average network delay using Equation 3. The maximum value of average delay, "TDelayMax", is taken to be the average delay generated in the initialization step. The average delay is normalized with respect to "TDelayMax". The membership function is shown in Figure 2(b).

4.2.7. Membership Function for Maximum Number of Hops

Again, two extreme values are determined. The minimum value, "THopsMin", is taken to be 1 hop, which will be the minimum possible in any case. The maximum value, "THopsMax", is taken to be the maximum number of hops between any source-destination pair generated in the initialization step. The membership function is shown in Figure 3.

In the proposed allocation scheme, all the selected links are removed one at a time and trial links are placed for each removed link. We start with the head-of-line link, i.e. the link with the worst goodness. We remove this link from the topology. This divides the topology into two disjoint topologies. After this, the placing of trial links begins. The approach adopted to place trial links is as follows. At most ten moves (i.e. trial links) are evaluated for each removed link. However, some moves may be invalid. Thus, we search for only four "valid"' moves. Whenever four valid moves are found, we stop; otherwise continue until a total of ten moves are evaluated (whether valid or invalid). The removal of a link involves two nodes *P* and *Q*, where *P* belongs to the subtree containing the root node and node *Q* belongs to the other subtree. For the ten moves we make, five are controlled and five are random. For controlled moves, we start with node *Q* and five *nearest* nodes in the other subtree are tried. For the random moves, we select any two nodes in the two subtrees and connect them.

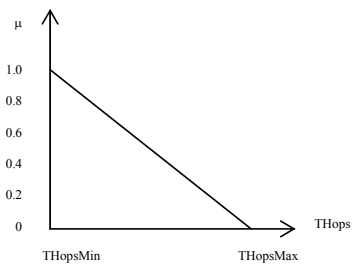

Figure 3: Membership function for maximum hops between a source-destination pair.

In the case where all ten moves are invalid, the original link is placed back in its position. The valid moves are evaluated based on Equation 7 and the best move among the ten moves is

made permanent. This procedure is repeated for all the links that are present in the set of selected links. As mentioned above, in the allocation phase, a number of moves are made for each link in the selection set and the best move is accepted, making the move (i.e., link) permanent.

4.2.8 Stopping Criterion

In our experiments, we have used a fixed number of iterations as a stopping criterion. We experimented with different values of iterations and found that for all the test cases, the SE algorithm converges within 4000 iterations or less.

4.3. Results

Experiments on five test cases were performed. Number of nodes in these test cases varied from 15 to 50 nodes. Traffic and bounds related to these test cases are given in Table 1. For each test case, a number of experiments were performed with different bias values, and the output values for Cost, delay, and hops were recorded. Table 2 gives these objective values for best bias in each test case. These values seem to be reasonable, for the corresponding test case.

Table 1: Characteristics of test cases used in our experiments. LCostMin, LCostMax, and TCostMin are in US$. TDelayMin is in milliseconds. Traffic is in Mbps.

Name	# of Local Sites	LCostMin	LCostMax	TCostMin	TDelayMin	Traffic
n15	15	1100	9400	325400	2.14296	24.63
n25	25	530	8655	469790	2.15059	74.12
n33	33	600	10925	624180	2.15444	117.81
n40	40	600	11560	754445	2.08757	144.76
n50	50	600	13840	928105	2.08965	164.12

Table 2: Results generated for different values for the test cases for best bias.

Case	SE			
	Bias	C	D	H
n15	0.2	314400	3.282	5
n25	0.2	509050	4.26	7
n33	0.0	687760	4.729	8
n40	0.3	866900	4.126	8
n50	0.3	1061900	5.32	9

5. ASSIGNMENT OF LAN SEGMENTS TO NETWORK DEVICES USING AUGMENTING PATH ALGORITHM

The algorithm has been used to solve the "terminal assignment problem" where the terminals (users) in a network are assigned to concentrators (e.g., hubs, or bridges) based on a cost function. The algorithm can be used to find an optimal solution [Kershanbaum, 1989]. The algorithm is based on the following observations:

1. Ideally, every terminal would be assigned to the nearest concentrator. The only reason for not doing so is that the capacity constraint (of the concentrator) prevents this [Kershanbaum, 1989].
2. If a terminal is already assigned to a nearby concentrator, the only reason for moving it to one that is farther away is to make room for another terminal that would have to detour even farther. Thus, it is only necessary to move terminals on concentrators that are full, and then only to make room for another terminal [Kershanbaum, 1989].
3. Given an optimal partial solution with k terminals assigned (i.e., these k terminals are assigned to concentrators at a minimum possible total cost), an optimal partial solution with k + 1 terminals can be found by finding the least expensive way of adding the k + 1st terminal to the k terminal solution. Note that this may involve reassigning some of the first k terminals [Kershanbaum, 1989].

Stated this way, it is clear that with sufficient effort it is possible to find an optimal solution, since an entirely new solution can be found by reassigning as many terminals as desired. Clearly, any solution, including the optimal one, is obtainable this way.

In this work, we have assumed that terminals have already been assigned to local concentrators, which make up a "segment". The AP algorithm is also suitable to assign the segments (hubs) to upper level concentrators (workgroup switch), which when interconnected will create a communication path between any two stations in the same or different work areas.

We conducted and experiment to show how the AP algorithm works. Figure 4 depicts a typical local site, where S denotes the segments, and C denotes the concentrators. There are fifteen segments and four concentrators located at four different places within the local site. Each concentrator (a hub or a switch in this case) has 4 ports. The task is to assign these fifteen segments to the four concentrators. When the AP algorithm is given the input (the distance of each segment from each concentrator, as shown in Table 3, the topology in Figure 5 is achieved. Table 3 also shows the concentrator to which the segment has been assigned.

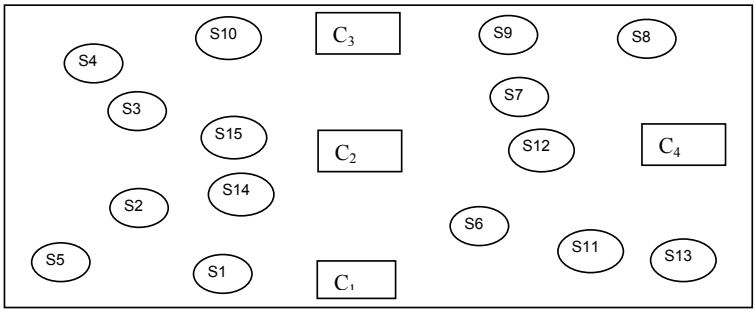

Figure 4: Segments and concentrators before assignment

6. DESIGN OF THE INTERNAL TOPOLOGY OF A LOCAL SITE

When the segments are assigned to the concentrators, the next step is to join these concentrators in a minimum spanning tree topology. For this purpose, we have used the proposed SE algorithm. We conducted an experiment for this. In this experiment, the only difference is the cost of equipment.

Since we have assumed that we are using Category 5 cable within a local site, we will include the cost of this instead of cost of fiber optic cable. Costs of the networking devices are the same as before. For this experiment, we have assumed that the local site is running on 10baseT Ethernet, and the maximum allowed traffic on a link should not exceed 6 Mbps (or 60%). When we apply the SE algorithm to the local site considered in Figure 5, we get the topology as shown in Figure 6. In this topology, we have:

- Total traffic = 4.2 Mbps.
- Traffic on link 1,2 = 2 Mbps.
- Traffic on link 1,3 = 2.6 Mbps.
- Traffic on link 3,4 = 2.7 Mbps.
- Monetary Cost = 80065 dollars.
- End-to-end average delay per packet = 2.894 milliseconds.
- Maximum number of hops = 3.

Note that these statistics include the cost, delay, and hops only of the topology that includes the concentrators C_1 to C_4 (and not the segments).

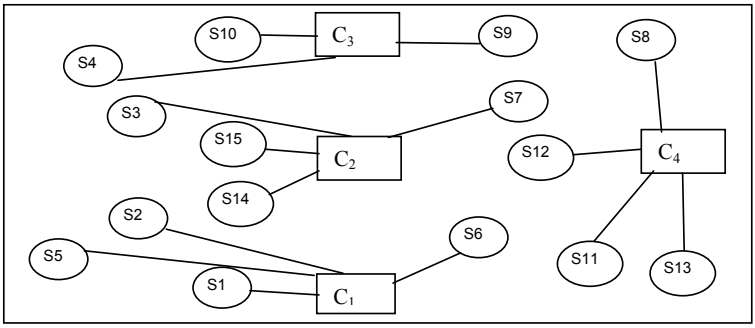

Figure 5: Segments and concentrators after assignment

Table 3: Segment-concentrator and concentrator-concentrator distances in meters. The concentrator to which the segment has been assigned is also shown.

	C_1	C_2	C_3	C_4	Segment assigned to
S_1	30	38	54	74	C_1
S_2	46	46	52	84	C_1
S_3	50	36	38	78	C_2
S_4	64	54	52	96	C_3
S_5	56	60	70	98	C_1
S_6	24	24	42	26	C_1
S_7	40	22	26	24	C_2
S_8	60	46	38	28	C_4
S_9	52	30	20	50	C_3
S_{10}	56	36	28	72	C_3
S_{11}	32	40	54	22	C_4
S_{12}	36	30	40	16	C_4
S_{13}	50	54	66	22	C_4
S_{14}	22	20	46	60	C_2
S_{15}	30	24	36	44	C_2
C_1		28	52	50	
C_2			28	46	
C_3				50	

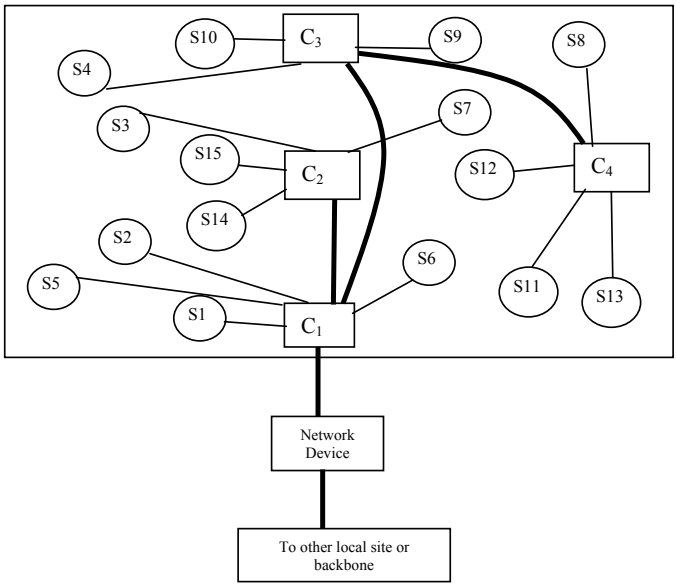

Figure 6: Topology generated for concentrators

7. CONCLUSION

In this paper we have presented a novel approach for topology design of campus networks based on fuzzy simulated evolution algorithm. Results obtained for test cases considered suggest that fuzzy simulated evolution algorithm performs better than Esau-Williams algorithm, which is a widely used algorithm for centralized network design.

ACKNOWLEDGEMENTS

Authors acknowledge King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, for support.

REFERENCES

1. Altiparmak, D. F. and Smith, A., 1997, "Local Search Genetic Algorithm for Optimal Design of Reliable Network", I*EEE Transactions on Evolutionary Computation*, pp 179-188.
2. Elbaum, R. and Sidi, M., 1996, "Topological Design of Local-Area Networks Using Genetic Algorithm", *IEEE/ACM Transactions on Networking*, pp 766-778, October 1996.

3. Ersoy, C. and Panwar, S., 1993, "Topological Design of Interconnected LAN/MAN Networks", *IEEE Journal on Selected Area in Communications*, pp 1172-1182, October 1993.
4. Esau. L.R. and Williams. K.C, 1966, "On teleprocessing system design. A method for approximating the optimal network", *IBM System Journal*, 5, pp 142-147.
5. Gen, M., K. Ida, and J. Kim, 1998, "A Spanning Tree-Based Genetic Algorithm for Bicriteria Topological Network Design", *Proceedings, IEEE International Conference on Evolutionary Computation*, p.164.
6. Kershenbaum, A., 1989, Telecommunications Network Design Algorithms, McGraw-Hill, USA.
7. Kling , Ralph Michael, 1990, "Optimization by Simulated Evolution and its Application to cell placement", Ph.D. Thesis, University of Illinois, Urbana, USA.
8. Kumar, M. P., and Gupta, Y., 1995 "Genetic Algorithm-Based Reliability Optimization for Computer Network Expansion", *IEEE Transactions on Reliability*, 24, pp 63-72.
9. Prim., R. C., 1957, "Shortest connection networks and some generalizations". *BSTJ*, 36, pp 1389-1401.
10. Sait, S. M., and Youssef, H., 1999, Iterative Computer Algorithms and their Application to Engineering, IEEE Computer Science Press, USA.
11. Youssef, H., Sait, S.M., and Issa, O., 1997, "Computer-Aided Design of Structured Backbones", *Proceedings, 15th National Computer Conference and Exhibition*, p. 593.
12. Youssef, H., Sait, S.M., and. Khan, S. A., 2002, "Topology Design of Switched Enterprise Networks Using a Fuzzy Simulated Evolution Algorithm", Accepted, Engineering Applications of Artificial Intelligence, Elsevier Publications.

Top