Improving Communication Network Topologies Using Tabu Search

S. Pierre & A. Elgibaoui
LICEF Research Center
Télé-université, Université du Québec
C.P. 670, Succ. C, Montréal, Qué., Canada, H2L 4L5
E-mail: spierre@teluq.uquebec.ca

Abstract

This paper proposes a tabu search approach for improving communication network topologies. The problem consists of finding, for a given set of nodes and traffic requirements, a network topology that minimizes the communication cost taking into account some performance and reliability constraints. For this purpose, some moves or local transformations called perturbations are applied to a starting topology in order to reduce the communication cost or the average packet delay. Preliminary results demonstrate the capability of such an approach to improve the communication cost as well as some performance attributes of network topologies.

1. Introduction

The topological design of communication networks can be considered as part of the overall network planning. It consists of finding a network topology that minimizes the total communication cost taking into account some performance and reliability constraints [1], [3], [7]. This problem is well known as very difficult. In order to make easier its resolution, it is usually divided into three subproblems: topological configuration, routing strategy, and capacity assignment [3]. In general, even this division into subproblems does not allow us to obtain exact or optimal solutions in reasonable computation times.

For illustration purposes, consider the topological configuration subproblem. If n denotes the number of nodes, the maximum number of links is given by n(n-1)/2. Thus, the number of possible topological configurations that can be constructed is $2^{n(n-1)/2}$. For n=10, there are $2^{45} = 3.5184 \times 10^{13}$ possible configurations. A powerful computer which would be capable of generating 1000 configurations per second would spend about 1,116 years to explore the entire search space of candidate topological configurations. There is an obvious risk of combinatorial explosion.

The other related subproblems, such as routing and capacity assignment, are not less trivial. As a result, only heuristic approaches can be suitably dealt with this hard combinatorial optimization problem. In fact, many heuristic methods have been proposed for the design of distributed communication networks: Cut Saturation (CS), the Branch X-Change (BXC), Concave Branch Elimination (CBE) and MENTOR algorithm [3], [7]. Other methods based on Artificial Intelligence (AI) are also been used [8], as well as metaheuristics such as simulated annealing and genetic algorithms [9], [10], [11].

This paper proposes a *tabu search* approach for solving the topological design problem of packet switched networks. It is organized as follows: Section 2 presents basic concepts and background related to communication networks. Section 3 outlines the tabu search method. Finally, Section 4 gives some implementation details and results.

2. Basic Concepts and Background

A distributed communication network can be modeled by a graph G=(N,A), where N denotes the set of n switching nodes and A the set of m edges or communication links. The maximum number of links in such a graph is given by $m_{max}=n(n-1)/2$. We denote d(i) the degree of node i, that is, the number of edges incident to node i. Furthermore, each link is

0742-1303/97 $10.00 © 1997 IEEE

characterized by a set of attributes: the capacity C_k, the flow f_k, the utilization U_k, the length L_k and the cost D_k. The capacity C_k of link k represents the maximum data rate in bits per second (bps) carried by this link. The flow f_k of the link k is the effective data rate in bps on this link. The utilization U_k of link k is the ratio of flow to capacity of this link. The length L_k of link k represents the Euclidean distance between its end nodes. Thus, the fundamental characteristics of a topology are: its topological configuration represented by a characteristic vector $t=(t_k)$, $k=1, 2, ..., m_{max}$, with $t_k=1$ if link k exists and 0 otherwise, a link capacity vector $C=(C_k)$, and a flow vector $f=(f_k)$.

In a packet switched communication network, each message is broken at the source into small blocks called packets which travel independently through the network. The traffic requirements between each node pair (i, j) can be represented by a matrix $\Gamma=[\gamma_{ij}]$, $i, j=1, 2, ..., n$, $i\neq j$, where γ_{ij} denotes the number of packets per second exchanged between nodes i and j.

A vital performance measure for a communication computer network is the average packet delay, T, that is, the mean time that a packet takes to travel from a source node to a destination node in the network. It can be expressed as follows [3]:

$$T = \frac{1}{\gamma} \sum_{k=1}^{m} \frac{f_k}{C_k - f_k} \qquad (1)$$

where γ is the total number of packets exchanged through the network. The average packet delay T must be less than the maximum acceptable delay T_{max}.

The cost D_k of a link k is generally a function of both the link length and capacity. Therefore, the total link cost D is given by:

$$D = \sum_{k=1}^{m} d_k \left(C_k \right. \qquad (2)$$

The flow assignment consists of determining the average number of packets λ_k carried by each link k. It is uniquely determined by the traffic matrix and the routing policy. In this paper, we adopt a fixed routing based on the Euclidean distance between nodes.

The availability of a computer network depends on the reliability of their materials components. For this reason, it is necessary to evaluate the overall network reliability by assuming a nonnull failure probability of network components. Various reliability measures have been proposed for computer communication networks. In this paper, we opt for the K-node-connectivity, with $2 \leq K \leq 5$.

Many different formulations of the design problem can be found in the literature [1],[3], [7], [11]. For sake of simplicity, we state the topological design of a distributed computer communication network as the search for a topology, denoted top= (t, f, C), which satisfies delay and connectivity constraints at the lowest possible cost. Tabu Search appears a suitable approach enabling to solve this problem according to a progressive improvement process.

3. Outline of the Tabu Search Approach

Tabu Search (TS) is an iterative improvement procedure that it starts from an initial feasible solution and attempts to determine a better solution in the manner of ordinary (descent) local method, until a local optimum is reached [2], [5]. This method can be used to guide any process that employs a set of moves for transforming one solution into another, and that provides an evaluation function for measuring the attractiveness of these moves [4], [6].

For a given combinatorial optimization problem, a solution space S may be defined as the set of all feasible solutions. TS associates with each feasible solution a numerical value that may be considered as the cost of the solution obtained by optimizing the cost function. For each solution s_i in the space solution S, there exists a subset of S, say $N(s_i)$, considered as a *neigbourhood* of s_i. This subset contains a set of feasible solutions that may be reached from s_i in one move. TS starts from an initial feasible solution s_i, and then moves to a solution s_j, which is also an element of $N(s_i)$. This process is repeated iteratively, and solutions that yield lower cost values than those previously encountered are recorded. The final cost recorded, when the search is interrupted, constitutes the overall optimum solution. Thus, TS can be viewed as a variable neighbourhood method: each step redefines the neighbourhood for which the next solution will be drawn [2].

A move from s_i to s_j is made on the basis that s_j ($s_i \neq s_j$) has the minimum cost among all the allowable solution in $N(s_i)$. Allowability is managed by a mechanism that involves historical information about moves made meanwhile the procedure progresses. TS is a high level procedure for solving optimization problems; it escapes the trap of local optimality by using short-term memory recording the most recently visited solutions [6]. The short-term memory constitutes a form of aggressive exploration that seeks to make the best move possible satisfying certain constraints. These constraints are designed to prevent repetition of some moves by rendering forbidden (tabu) the selected move attributes. Such attributes are maintained in a *tabu list* (LT).

On the other hand, TS permits backtracking to previous solutions which may ultimately lead, via a different direction, to better solutions. This flexibility element is implemented through a mechanism called *aspiration criteria* [6]. The goal of aspiration criteria is to increase the flexibility of the algorithm while preserving the basic features that allow the algorithm to escape local optima and avoid cyclic behaviour. Figure 1 presents a general version of TS [2].

```
Given
        $X$ : Solution feasible space
        $f$: Objective function, $X \rightarrow R$, defined on $X$
        $N(s)$ : Neighbourhood of $s \in X$
        $|LT_i|$ : Length of each tabu list $LT_i$
        $f^*$ : Lower bound of objective function
        nbmax : Maximum number of iterations between two improvements of $s^*$
Initialization
        Choose by any heuristic an initial solution $s \in X$
        $s^* := s$        ($s^*$ best solution obtained)
        nbiter := 0        (Iteratios counter)
        bestiter := 0   (iteration given the last $s^*$)
        Initialize $LT_i$ for each i
        $A_j(s, m) = +\infty$
While ($f(s) > f^*$) and (nbiter - bestiter < nbmax ) DO
          nbiter := nbiter +1
          generate a sample $V^* \subseteq N(s)$ of neighbour solutions
          if [one of the tabu conditions is violated, i.e. $t_i(s,m) \in LT_i$] or [at least one of
aspiration criteria conditions is verified, i.e. $a_j(s,m) < A_j(s,m)$ ] then
          Choose the best $s' \in V^*$ minimizing f on $V^*$ (by a heuristic)
          If $f(s') < f(s^*)$   Then    $s^* := s'$
                                  bestiter := nbiter
          Update of tabu lists $LT_i$
          Update of threshold values $A_j(s,m)$
        $s := s'$
Endwhile

Result  $s^*$ : best solution found after the execution of the procedure
```

Figure 1. General version of TS

4. Implementation Details and Results

Adaptation of TS to a specific problem essentially relies on the definition of moves that describe the neighbourhood to be explored. In the case of topological design of communication networks, some moves or local transformations called perturbations are applied to a starting topology in order to reduce its total link cost and/or to improve its average packet delay. These perturbations deal with *addition, removal and substitution* of links.

In order to characterize the network topologies to be modeled, a certain number of descriptors are required. In fact, given γ_{ij} the traffic between nodes i and j, and m_{max} the maximum number of links, the average traffic between the node pairs is given by $r_{aver} = \gamma / m_{max}$. Therefore, the index traffic of a link (i, j) can be defined as $I_{ij} = \gamma_{ij} / r_{aver}$.

On the other hand, the descriptor L_{max} represents the maximum Euclidean distance between all the node pairs of the network. The excess cost of a link is $E_i = D_i (C_i - f_i) / C_i = D_i(1-U_i)$. If $E = D(1-U)$ denotes the average excess cost of links, where D denotes the average cost of links and U the average rate of link utilization, we can define the excess index cost of a link i as $P_i = E_i / E = [D_i (1-U_i)] / [D(1-U)]$. Finally, the ratio of average delay T to maximum acceptable delay T_{max} refers to the normalized delay T_n.

The main goal of removal moves is to reduce the total link cost. If a starting topology contains the minimum number of links to preserve the K-connectivity, then it is impossible to apply to it a removal move because the connectivity constraint would be automatically violated. In our implementation, we defined three types of removal moves types :

M_1^R the removal of links k=(i, j) such that $L(k) > \alpha * L_{max}$, with $0 < \alpha \leq 1$, d(i) and d(j) remaining at least equal to the desired connectivity degree after the link removal;

M_2^R the removal of links k=(i, j) such that the excess index cost P_k is less than 1, d(i) and d(j) remaining at least equal to the desired connectivity degree after the link removal;

M_3^R the removal of links k=(i, j) such that the utilization rate U_k is less than α, with $0 < \alpha \leq 1$, d(i) and d(j) remaining at least equal to the desired connectivity degree after the link removal.

The addition moves cannot violate the connectivity constraint, but generally have negative effects on the cost. They are defined as follows:

M_1^A to each topology which have a normalized delay (T_n) greater than 1, add the links k=(i, j) that have an index traffic greater than 1;

M_2^A to each topology which have a normalized delay (T_n) less than 1, add the links k=(i,j) such that $L(k) < \alpha * R$, with $0 < \alpha \leq 1$.

The substitution moves correspond to a sequence of removal and addition of links. These moves which cannot guarantee the preservation of the connectivity degree are defined as follows:

M_1^S the substitution of all link k=(i, j) such that $L(k) > \alpha * L_{max}$, with $0 < \alpha \leq 1$, by two other links v=(i, p) and w=(j, q), with $L(v) \leq \alpha_1 L(k)$ and $L(w) \leq \alpha_2 L(k)$, $0 < \alpha_1 \leq 1$ and $0 < \alpha_2 \leq 1$;

M_2^S the substitution of all link k=(i, j) such that $U_k > \alpha$, with $0 < \alpha \leq 1$, by two other links v=(i, p) and w=(j, q) with $I_{ip} > 1$ and $I_{iq} > 1$.

The neighbourhood of a current topology s is a finite set N(s) of topologies which are said to be feasible. In our implementation, this neighbourhood cannot contain more than 6 topologies, each of these is obtained from the current topology by applying one move. Our adaptation of TS has been implemented in *Turbo-Pascal version 7.0* on a *Pentium IBM compatible, 60Mhz.*

In order to evaluate the effectiveness and efficiency of our approach, we have considered a set of 20 nodes defined by the Cartesian coordinates given in Table 1. The capacity options are given in Table 2, and the traffic matrix is provided by Table 3. The maximum acceptable delay is T_{max}=250 ms, the degree of connectivity is equal to 3 and the average packet length 1000 bits. The maximum number of iterations between two successive improvements of the best solution is equal to 15. Figure 2 shows the resulting initial topological configuration.

To this initial topology, the application of the third removal move links M_3^R, during the first perturbation cycle has generated, by deletion of links 1-14, 2-5, 10-15, the topology shown in Figure 3. This topology has the lowest cost among the generated topologies. Thus, it becomes the solution of the first perturbation cycle, with a total link cost D=127,890.26 $/month. The first perturbation cycle did not lead to a solution with a cost lower than the cost of the initial topology of Figure 2. By contrast, we observe a diminution of the average packet delay.

From the starting topology shown in Figure 2, the final topology of Figure 4 has been reached during the seventh perturbation cycle after 22 iterations. The cost of the initial topology is passed from 124,222.98 $/month to 102,419.91 $/month, say a cost improvement of 17.6%. By contrast, the average delay is worsened from 81.84 ms to 133.68 ms, that is again less than the maximum acceptable delay T_{max} =250 ms. Since we are looking for a solution that minimizes the total link cost, taking into consideration some performance constraints, this solution becomes the best solution found. Figures 5 and 6 shows the evolution of cost and delay respectively during the execution process of the TS algorithm.

Table 1. Node coordinates

Node	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Abscissa	63	22	41	33	32	40	52	80	19	15
Ordinate	8	72	45	10	84	78	52	33	35	81
Node	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
Abscissa	27	27	70	48	4	10	56	82	9	95
Ordinate	3	16	96	4	73	71	27	47	54	61

Table 2. Capacity options and costs

Capacity **(Kbps)**	Fixed cost **($/month)**	Variable cost **($/month/Km)**
9.60	650.00	0.40
19.20	850.00	2.50
50.00	850.00	7.50
100.00	1700.00	10.00
230.40	2350.00	30.00
460.80	4700.00	60.00
921.60	9400.00	120.00
1843.20	18800.00	240.00

Table 3. Traffic matrix

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	0	4	7	8	6	9	5	5	9	3	9	3	6	4	3	4	1	7	7	10
2	4	0	9	3	2	8	3	1	2	9	8	10	6	10	3	6	10	2	6	3
3	7	9	0	7	10	10	6	5	5	10	6	8	8	9	1	2	4	4	10	9
4	8	3	7	0	9	6	5	4	7	3	7	6	8	5	9	4	2	10	3	5
5	6	2	10	9	0	8	1	3	8	6	6	7	6	3	3	1	6	5	4	2
6	9	8	10	6	8	0	10	9	6	7	1	5	1	5	9	6	10	5	4	2
7	5	3	6	5	1	10	0	9	7	4	7	6	2	8	5	3	8	8	3	4
8	5	1	5	4	3	9	9	0	9	4	4	4	1	5	10	9	7	7	10	3
9	9	2	5	7	8	6	7	9	0	6	7	1	4	5	10	6	6	8	8	3
10	3	9	10	3	6	7	4	4	6	0	10	2	6	5	7	2	4	8	7	8
11	9	8	6	7	6	1	7	4	7	10	0	5	4	6	10	9	8	5	6	2
12	3	10	8	6	7	5	6	4	1	2	5	0	3	6	5	6	6	3	7	7
13	6	6	8	8	6	1	2	1	4	6	4	3	0	3	9	3	4	7	1	9
14	4	10	9	5	3	5	8	5	5	5	6	6	3	0	3	4	3	4	2	4
15	3	3	1	9	3	9	5	10	10	7	10	5	9	3	0	8	5	1	9	9
16	4	6	2	4	1	6	3	9	6	2	9	6	3	4	8	0	2	1	4	7
17	1	10	4	2	6	10	8	7	6	4	8	6	4	3	5	2	0	10	10	9
18	7	2	4	10	5	5	8	7	8	8	5	3	7	4	1	1	10	0	10	6
19	7	6	10	3	4	4	3	10	8	7	6	7	1	2	9	4	10	10	0	8
20	10	3	9	5	2	2	4	3	3	8	2	7	9	4	9	7	9	6	8	0

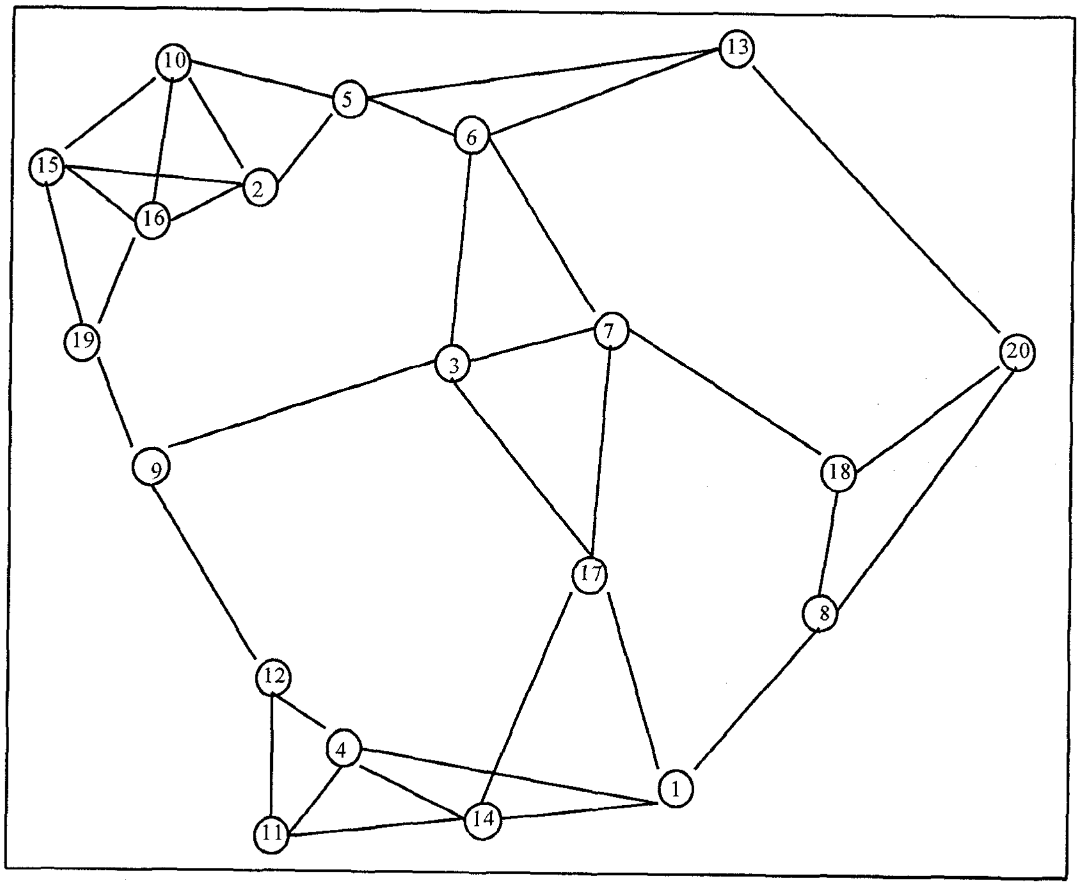

Figure 2. Initial topological configuration
(D = 124,222.98 $/ month, T = 81.84 ms)

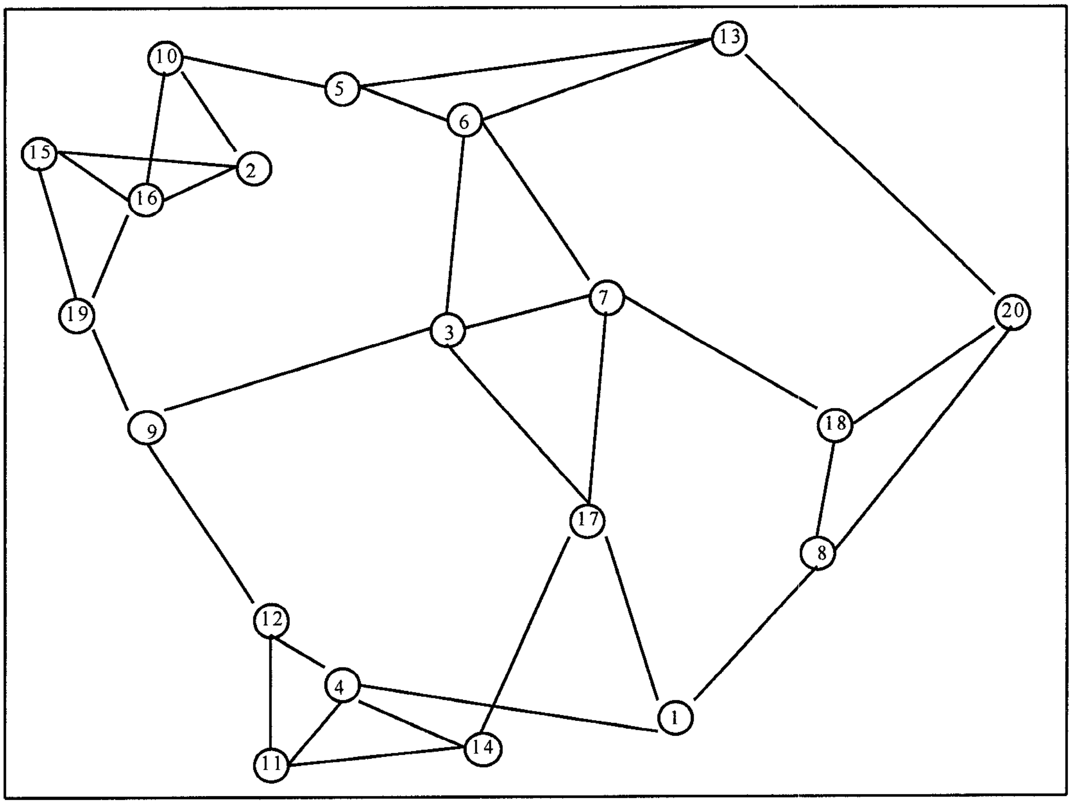

Figure 3. Best topology found during the first perturbation cycle
(D = 127,890.26 $/ month, T = 73.78 ms)

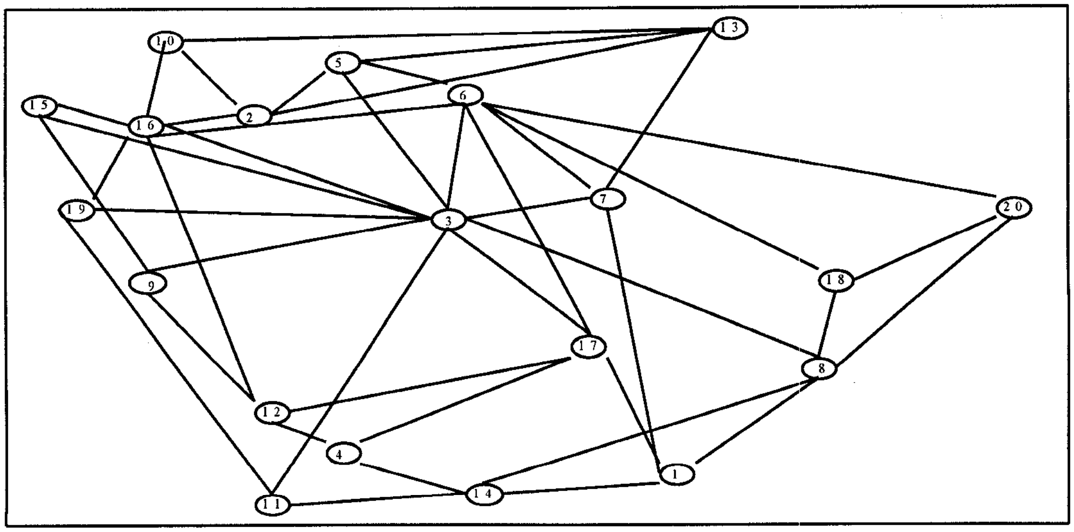

Figure 4. Topological configuration of the (last) final solution
(D = 102,419.91 $/month, T = 133.68 ms)

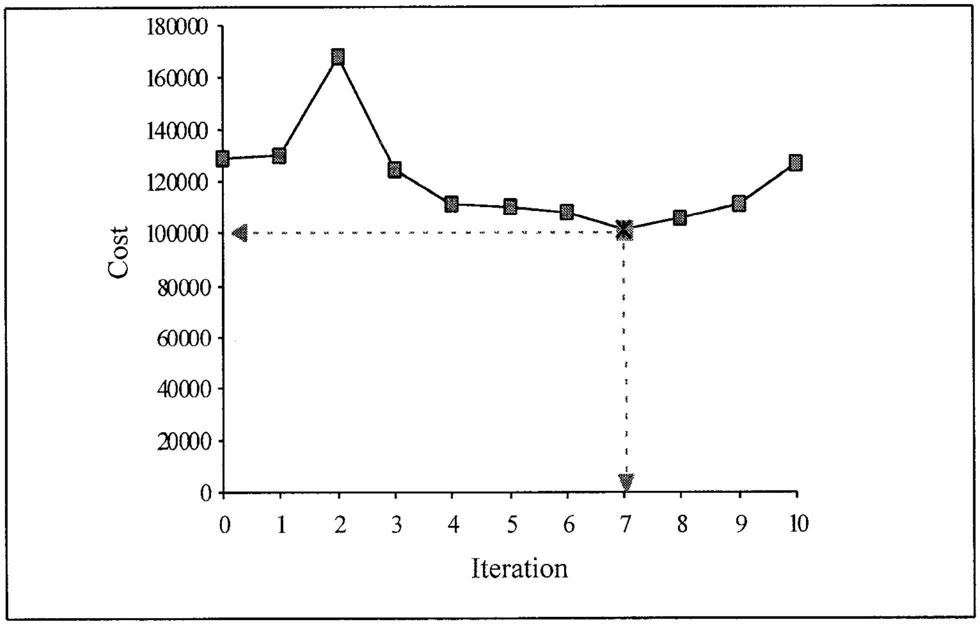

Figure 5. Evolution of cost versus iteration

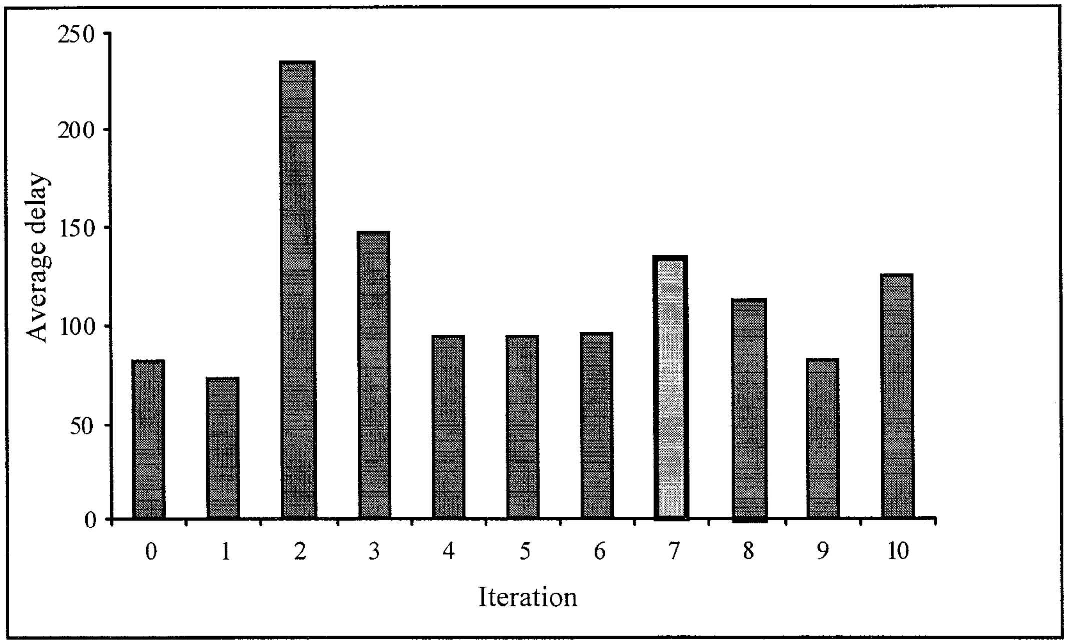

Figure 6. Evolution of delay versus iteration

5. Conclusion

This paper has been dealt with topological design of computer networks considered as a part of the overall network planning. More specificaly, it proposed an heuristic method called Tabu Search to reduce the search of candidate topologies and provide suboptimal solutions. Simulation results confirm the efficiency and the robustness of this method for designing fault-tolerant and survivable backbone networks. Finally, our implementation of TS generally provides better solutions than alternative heuristics such as: Cut Saturation, Simulated Annealing and Genetic Algorithm.

6. References

[1] Dutta, A., and Mitra, S. 1993. «Integrating Heuristic Knowledge and Optimization Models for Communication Network Design». *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, no. 6, pp. 999-1017.

[2] de Werra, D., Glover, F. and Taillard, E. 1993. «A user's guide to tabu search». *Annals of Operations research*, vol. 41, pp. 3-28.

[3] Gerla, M., and Kleinrock. L. 1977. «On the Topological Design of Distributed Computer Networks». *IEEE Transactions on communications*, vol. COM-25, no.1, pp. 48-60.

[4] Glover, F. 1994. «Tabu Search: Improved Solution Alternatives for Real World Problems». in *Mathematical Programming: State of the Art*, Birge & Murty, eds., University of Michigan Press, pp. 64-92.

[5] Glover, F. 1995. «Tabu Thresholding: Improved Search by Nonmonotonic Trajectories». *INFORMS Journal on Computing*, vol. 7, pp. 426-442.

[6] Glover, F. and Laguna, M. 1993. «Tabu Search». in *Modern Heuristic Techniques for Combinatorial Problems*, C.Reeves, Ed., Blackwell Scientific Publishing, pp.70-141.

[7] Kershenbaum, A., Kermani, P. and Grover, G. A. 1991. «MENTOR: An Algorithm for Mesh Network Topological Optimization and Routing». *IEEE Transactions on communications*, vol. 39, no. 4, pp. 503-513.

[8] Pierre, S. 1993 «Application of Artificial Intelligence Techniques to Computer Network Topology Design». *Eng. Applications of Artificial Intelligence*, vol. 6, no. 5, pp. 465-472.

[9] Pierre, S. and Legault, G. 1996. «An Evolutionary Approach for Configuring Economical Packet-Switched Computer Networks». *Artificial Intelligence in Engineering*, vol. 10, no. 2, pp. 127-134.

[10] Pierre, S., Hyppolite, M.-A., Bourjolly, J.-M. and Dioume, O. 1995. «Topological Design of Computer Communications Networks Using Simulated Annealing». *Eng. Applications of Artificial Intelligence*, vol. 8, no. 1, pp. 61-69.

[11] Rose, C. 1992. «Low Mean Internodal Distance Network Topologies and Simulated Annealing». *IEEE Transactions on communications*, vol. 40, no.8, pp. 1319-1326.