A Tabu Search Heuristic for Capacitated Network Design

Mohamed Zied Ben Hamouda  
LAAS-CNRS  
Université de Toulouse  
F-31077 Toulouse, France  
zied.ben-hamouda@laas.fr

Olivier Brun  
LAAS-CNRS  
Université de Toulouse  
F-31077 Toulouse, France  
brun@laas.fr

Jean-Marie Garcia  
LAAS-CNRS  
Université de Toulouse  
F-31077 Toulouse, France  
jmg@laas.fr

Abstract
This paper studies a topical capacitated network design problem that arises in the telecommunication industry. In this problem, given point-to-point demand between various pairs of nodes, a minimum cost survivable network must be met by installing capacitated equipments (routers, line cards, etc.) on nodes as well as link facilities on arcs. We present a Tabu Search heuristic to find a solution that minimizes the total network cost. Numerical results are provided for randomly generated networks and networks coming from real-word applications.

1 Introduction
In a world where information technology (IT) is becoming pervasive, communication networks appear more and more as strategic resources. Near-optimal design of these networks is a critical issue since network design greatly affects the long-term network performances and it determines most of the investment cost. Since this investment cost is typically huge and since the return on investment cannot be expected before years, it is crucial to ensure that it is properly minimized.

Network operators as well as many governmental organizations operating in strategic areas (e.g. defense, air control, energy, etc.) have always been aware of the critical importance of efficient network design approaches. But we are also witnessing a radical change in the strategy of many large companies with respect to the governance of their IT infrastructure. Disappointed by the lack of quality and flexibility of the VPN (Virtual Private Network) services offered by network operators, these companies are now investigating the opportunity to acquire their own communication infrastructure as well as the skills to design an optimized network architecture. Therefore there is a renewed interest in efficient network design methods.

1.1 Previous Works
In recent years, the topological design of networks has been fairly discussed in the literature and several design models and algorithms have been proposed.

The early literature in this area has mostly focused on uncapacitated problems (see e.g. [1], [11], [17]). Uncapacitated models deal only with topological aspects. They consider capacity-independent costs associated with the length of network links. In other words, uncapacitated models assume that link costs are independent of the type of communication line that will effectively be installed and that other equipment costs (routers, line cards, etc.) can be neglected in this first design phase. The rationale for this approximation is that the topological design of a wide-area network incurs capacity-independent fixed costs which are several orders of magnitude larger than the equipment costs. These “fixed” costs typically represent the costs of digging trenches for optic fibers, site opening costs, or even equipment installation and configuration costs.

More recently, with the massive deployment of optic fiber in all western countries, the cost of leasing transmission lines has become cheaper and cheaper. As a consequence, equipment costs have become a significant fraction of the total cost when designing a network. There is therefore an increasing need for an integrated approach for network design taking into account dimensioning of the equipments in the early stages of the design process. The first steps in this direction were performed in [4, 5, 6, 7, 13, 14, 15] where the authors study versions of the capacitated network design problem with facilities to be installed on the arcs. More recently, Frangioni and Gendron studied 0-1 reformulations of the multi-commodity capacitated network design problem in [2].
1.2 Our Contribution

Although researchers have successfully solved variations of the uncapacitated network design problem (see for example [1] and [3]), the general capacitated network design problem has proven to be considerably more difficult due to the complexity of the cost structure [16].

This paper proposes a Tabu Search heuristic for a realistic capacitated network design problem where a minimum cost survivable network topology has to be designed taking into account capacity-dependent link costs as well as all other equipment costs (routers, line cards). The motivation for this study comes from the fact that the problem we deal with arises in the telecommunication industry.

1.3 Paper Outline

The remainder of the paper is structured as follows. Section 2 presents a formal description and definition of the capacitated network design problem we deal with in this paper. In Sections 3 and 4, we discuss alternative solution strategies for the problem and provide motivation for our proposed solution approach. Computational results are reported in Section 5 on typical telecommunication data. We conclude this paper by discussing some related open problems and suggesting further research.

2 Problem Definition

The design problem we consider in this paper is that of finding a communication network with minimum cost given:

1. a set of mandatory terminal nodes and potential transit nodes. A Potential transit node is an optional node which may not be part of the network at all. Such a node does not generate demands but does only transit the flows between the terminal nodes,
2. traffic demand between each pair of terminal nodes,
3. permissible communication delays between each pair of mandatory nodes,
4. a set of permissible link models, router models and line card models. Routers and line cards are to be settled on the network nodes (i.e., terminal and transit nodes).

The objective function to be minimized represents the sum of acquisition and installation costs of all the links and equipments (i.e., routers and line cards) required to establish the network (see section 4).

The resulting network should be survivable: after failure of a single node or link, the network still allows communication between all non-faulty mandatory nodes. This implies constraints on the connectivity of the network.

Furthermore, the network must guarantee that the delay required to transmit packets between mandatory nodes is limited even in case of link/node failure. In other words, the failure of a single node or link should not increase the distance between two terminal nodes beyond a given threshold. This property is a desirable feature since in case of a network failure, the traffic is rerouted with a limited alteration of the communication delay.

Figure 1 illustrates an instance of the network design problem we deal with. The goal is to find the minimum cost network that connects the mandatory nodes (represented in the figure by squares). Four transit nodes (represented by squares) could be used. Sub-figure 1(b) shows a feasible solution to the network design problem. We see that only three transit nodes are used and several types of links and routers are used.

![Figure 1. Example of Feasible Solutions to our Network Design Problem](image-url)
satisfy the following property: each path separating a pair of nodes of $M$ is bounded by $K_n$ and $K_f \geq K_n$ in nominal and failure states respectively. For a more detailed formulation for the network design problem stated in this section, we refer the reader to [12].

The problem stated above is NP-hard, as it contains the fixed charge network design problem (and thus, the Steiner tree problem) as a special case. It thus provides opportunities for the application of heuristic methods. In the next section, a Tabu Search heuristic is proposed to address this problem.

3 Solution Procedure

The nature of the design problem we address in this paper implies that if an exhaustive search is employed, one must scan through a large number of alternatives to find the optimal solution. Exact approaches, such as integer programming and Branch-and-Cut, attempt to solve the problem to optimality. Figure 2 shows the computing times as a function of the total number of nodes obtained with an exact Branch-and-Cut algorithm we developed for this problem. The algorithm is detailed in [12]. It can be noticed from this figure that exact approaches, such as Branch-and-Cut, are applicable to networks with a small number of nodes. However, as the number of nodes in practice is important, we expect that computing times will rise beyond a tolerable range for large networks.

![Figure 2. Computing times of the Branch-and-Cut algorithm](image)

To solve our design problem the emphasis is on heuristic algorithms with lower computational complexities than the exact ones and producing solutions which are close to the optimal solution.

In recent years, Tabu Search (TS) has been applied with a high degree of success to a variety of NP-hard problems [8], [9] and [10]. It has also proved its effectiveness in many network design problems (see e.g., [3]). TS is basically an iterative neighborhood search strategy that explores the solution space by moving from a solution to the solution with the best cost in its neighborhood at each iteration even in the case that this might cause the deterioration of the objective. Through such moves, the method can escape from bad local optima. To avoid cycling, a short term memory, known as the tabu list, stores previously visited solution or components of previously visited solutions. It is then forbidden or tabu to come back to these solutions for a certain number of iterations. While central to TS, tabus are sometimes too powerful: they may prohibit attractive moves, even when there is no danger of cycling, or they may lead to an overall stagnation of the searching process. It is thus necessary to use algorithmic devices that will allow one to revoke (cancel) tabus. These are called aspiration criteria.

Many other ingredients (often problem-specific) may appear in an effective TS algorithm. Nevertheless, basic tools that were presented in this section are sufficient in the scope of this work. In the following, we describe how we adapted the ideas of TS to solve our network design problem.

3.1 Initialization

Our TS algorithm starts from an initial feasible solution and tries to reach a near optimum solution by means of moves. An initial solution can be obtained using for instance the Greedy method described in Algorithm 1. Such a descent algorithm is inspired from the uncapacitated one introduced by Fortz et al. in [3]. It belongs to local search methods. The algorithm is based on a greedy removal of edges. It starts with an initial topology. At each iteration the edges are checked, and we select the edge which removal gives the best decrease in the total network cost; then we remove that edge. This procedure is repeated until no improvement can be achieved and while preserving feasibility.

In the algorithm, the function $getCost()$ computes the overall network cost while taking into account the cost of equipments and capacitated links.
Algorithm 1 Capacitated Greedy Algorithm

Require: a feasible solution $S_0$ and a network graph $G_0$ that models $S_0$
1: $G_{\text{best}} \leftarrow G_0$, $G_i \leftarrow (V_i, E_i) \leftarrow G_0$
2: $\text{best}_\text{cost} \leftarrow \text{getCost}(G_0)$
3: repeat
4: \hspace{1em} iteration\_cost $\leftarrow \infty$
5: \hspace{1em} for all $e \in E_i$ do
6: \hspace{2em} $G \leftarrow (V_i, E_i \setminus \{e\})$
7: \hspace{2em} cost $\leftarrow \text{getCost}(G)$
8: \hspace{2em} if cost < iteration\_cost then
9: \hspace{3em} $G_i \leftarrow G$
10: \hspace{3em} iteration\_cost $\leftarrow$ cost
11: \hspace{2em} end if
12: \hspace{1em} end for
13: \hspace{1em} if iteration\_cost < best\_cost then
14: \hspace{2em} best\_cost $\leftarrow$ iteration\_cost and $G_{\text{best}} \leftarrow G_i$
15: \hspace{2em} end if
16: until iteration\_cost $\geq$ best\_cost

3.2 Search space and Neighborhood structure

The neighborhood structure is the most important issue in the development of a tabu search heuristic. Consider a minimal feasible solution $S$. We mean by the term “minimal” that the removal of any edge from $S$ makes it not feasible or increases its total cost. If a better solution $S'$ exists, it is easy to see that $S'$ contains at least one potential node or one edge which is not in $S$. Based on this observation, we define two sets of neighbors of $S$: $N_1(S)$ and $N_2(S)$.

$N_1(S)$ contains all the feasible solutions that can be obtained from $S$ by a simple cut-and-paste operation on the graph $G_S$ representing $S$, that keeps the topology feasible, namely:

1. add an arc to $G_S$, and then,
2. cut a set of arcs from the graph just formed using the Greedy algorithm.

To define the neighborhood $N_2(S)$, we consider two types of moves on $S$:

1. add a potential node to $G_S$ and connect it to the other used nodes, and then
2. apply the Greedy algorithm.

To compute the cost of a potential solution $S' \in N_1(S) \cup N_2(S)$, one should check whether $G_S'$ is feasible, find the shortest paths on $G_S'$, propagate the traffic along these paths, and finally compute the best link and equipment models to be used to carry the traffic on each link. In Subsection 4, a method to compute the best link and equipment models to be installed is presented.

3.3 Tabu List

Once the best solution in the neighborhood is chosen, we give a tabu status to the links that were added for a number of iterations randomly chosen. These links cannot be removed for that number of iterations. By doing so, many moves may be forbidden and thus, many new solutions may be overlooked to reduce the risks of cycling. However, a tabu move may still be applied if it leads to a solution that is better than the best solution visited thus far. This is the only aspiration criterion we use. Note that trying to first remove links which are not tabu leads to a neighborhood structure strongly related to the current tabu set, therefore enforcing diversification.

3.4 Stopping Criteria of the TS Algorithm

Within our TS algorithm, the maximum number of iterations is set to a fixed value. The algorithm is stopped once this maximum number of iterations has been reached or after a number of consecutive iterations without improving the current solution.

3.5 TS Algorithm

In this section, we sketch in Algorithm 2 the general algorithmic structure of our TS heuristic. In this algorithm, the variables $S_i$ and $S^*$ are used to store the solution at iteration $i$ and the best overall solution, respectively. We denote also by $\Gamma(S)$ the cost of solution $S$.

4 Network Cost

The cost of a feasible solution is composed of link costs and equipment costs, as detailed below.

4.1 Link Costs

Let consider a feasible solution $S$ and $G_S = (V_S, E_S)$ the graph of $S$. We assume that the cost of a link $(i, j) \in E_S$ is an increasing function $F_{ij}(\cdot)$ of its capacity,
and thus of its load \( y_{ij} \). Link costs \( F_{ij}(\cdot) \) are also typically piecewise linear increasing functions of the euclidean distance between the end nodes, as usual with tariff systems used by providers of leased lines.

The capacity of link \((i, j)\) has to be selected among a set of modular link capacities. Let \( T \) be the number of link/port types and \( r_t \) be the bandwidth of type \( t = 1, \ldots, T \) (sorted in the order of increasing capacity).

Let the binary variable \( \beta_{ij}^t \) be 1 if \( t \) is the minimum value such that \( r_t \geq \max(y_{ij}, y_{ji}) \), and 0 otherwise. The global link cost of solution \( S \) is given by,

\[
\sum_{e \in E_S} x_e F_e(\sum_{t=1}^T \beta_{ij}^t r_t)
\]  

(1)

where the binary variable \( x_e \) is defined as follows:

\[
x_e = \begin{cases} 
1 & \text{if link } e \text{ is used in } S \\
0 & \text{otherwise} 
\end{cases}
\]

### 4.2 Equipment Costs

The optimal cost of the equipments to be settled at node \( v \) in solution \( S \) just depends on the number of line cards of each type that have to be used for this node.

A link of type \( t = 1, \ldots, T \) needs to be connected to an interface card. We assume that there are \( T \) types of line cards, so that there is a one-to-one correspondence between link types and line card types. Let \( p_t \) be the number of available ports on a line card of type \( t \), and let \( \phi_t \) be the cost of such a line card. A router on which line cards will be plugged has to be selected among \( R \) router types. For router model \( r = 1, \ldots, R \), let \( \bar{\tau}_r \) be the number of available slots, \( \bar{T}_r \) be the maximum throughput of the router forwarding engine, and \( \bar{\psi}_r \) be the cost of such a router.

Let \( \delta(v) \) denotes the set of links originating from \( v \). Let also the binary variable \( e_t^v \) be 1 if the model of line \( e \) is \( t \). The minimum number \( p_t^v(S) \) of cards of type \( t \) required to support links connected to node \( v \) is given by,

\[
p_t^v(S) = \left[ \frac{\sum_{e \in \delta(v)} e_t^v}{p_t} \right]
\]

where for any real-valued \( z \), \( \lceil z \rceil \) denotes the lowest integer \( n \) such that \( z \leq n \).

Let \( \mu^*(p_v(S)) \) denotes the optimal equipment cost for node \( v \) in the solution \( S \), where \( p_v(S) \) is the vector \([p_1^v(S), \ldots, p_T^v(S)]\). The optimal equipment cost for node \( v \) is given by,

\[
\mu^*(p_v(S)) = \sum_{t=1}^T p_t^v(S) \phi_t + \min \{ \bar{\psi}_r : p_v(S) \in \Lambda_r \}
\]

where,

\[
\Lambda_r = \left\{ s = (s_1, \ldots, s_T) : \sum_t s_t \bar{\tau}_r, \sum_t s_t p_t r_t \leq \bar{T}_r \right\}
\]

is the set of all card configurations that can be accommodated by a router of type \( r = 1, \ldots, R \).

Note that the function \( \mu^* \) can be efficiently computed beforehand for each possible card configuration, i.e. for each possible value of the vector \( p_v(S) \).

## 5 Computational Results

In order to evaluate our TS heuristic, we performed computational experiments on randomly generated networks with characteristics frequently observed in practice. We first introduce the cost data and network topologies used for the experiments. Then, we present numerical
<table>
<thead>
<tr>
<th>Router Model</th>
<th># Slots</th>
<th>Throughput</th>
<th>Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>12</td>
<td>3 Gbps</td>
<td>5,000</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>5 Gbps</td>
<td>10,000</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>10 Gbps</td>
<td>15,000</td>
</tr>
</tbody>
</table>

Table 1. Router models.

<table>
<thead>
<tr>
<th>Line Card Model</th>
<th># Ports</th>
<th>Throughput (Mbps)</th>
<th>Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12</td>
<td>12 × 20</td>
<td>2,000</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>8 × 50</td>
<td>3,000</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4 × 100</td>
<td>5,000</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2 × 1000</td>
<td>8,000</td>
</tr>
</tbody>
</table>

Table 2. Line card models.

results obtained with TS heuristic and compare them to the optimum and the results obtained by the Greedy heuristic.

5.1 Cost data

The results below have been produced for a period of one year and using the router models, line card models and link costs presented in Tables 1, 2 and 3. Tests were made for random problems, with vertices uniformly generated in a square of size 200 km × 200 km. Traffic demand between each pair of mandatory nodes varies from 1 to 20 Mbps.

5.2 Experiments

In this section, the computational performances of our TS heuristic are assessed and compared to the optimum for scenarios with a small number of nodes. We also compare the TS heuristic with the greedy heuristic for problem instances of practical interest (10 to 50 nodes).

Figure 3 shows the relative gaps in terms of network cost between the optimal solution on the one hand and TS and Greedy heuristics on the other hand. Figure 4 portrays the computing times we obtained for an exact Branch-and-Cut algorithm (see [12]), for the TS heuristic and for the Greedy heuristic.

Table 3. Link costs.

<table>
<thead>
<tr>
<th>Capacity</th>
<th>Cost/Km per year ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 Mbps</td>
<td>300</td>
</tr>
<tr>
<td>50 Mbps</td>
<td>400</td>
</tr>
<tr>
<td>100 Mbps</td>
<td>500</td>
</tr>
<tr>
<td>1 Gbps</td>
<td>600</td>
</tr>
</tbody>
</table>

Table 3. Link costs.

Figure 3. Gaps Between the Optimal Solution and TS Heuristic and the Greedy one Respectively

Figure 4. Computing Times

It can be noticed on these figures that the TS heuristic allows to keep reasonable computing times, while providing near-optimal solutions. The quality of solutions obtained with the Greedy algorithm is relatively low when compared to that of the TS heuristic.

Table 4 compares the total network costs as well as the computing times we obtained under the TS and Greedy heuristics. Random problems with 10 to 50 nodes were generated, and we tested several instances of each size.
<table>
<thead>
<tr>
<th># Nodes</th>
<th>Greedy Heuristic</th>
<th>TS Heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cost(K)</td>
<td>Time(s)</td>
</tr>
<tr>
<td>10</td>
<td>968</td>
<td>1.05</td>
</tr>
<tr>
<td>20</td>
<td>1058</td>
<td>15.5</td>
</tr>
<tr>
<td>30</td>
<td>1352</td>
<td>65.5</td>
</tr>
<tr>
<td>40</td>
<td>1438</td>
<td>175.1</td>
</tr>
<tr>
<td>50</td>
<td>1574</td>
<td>289.5</td>
</tr>
</tbody>
</table>

Table 4. TS heuristic v.s. Greedy heuristic

The results indicate that the TS heuristic outperforms the greedy heuristic with respect to solution quality. In particular, the TS heuristic finds better solution than the greedy heuristic, even though it takes more computing time to find the best solution. This behavior is due to the fact that TS explores a more restricted neighborhood. However, computing time of the proposed TS heuristic remains quite reasonable even for problem instances of practical interest. Clearly this demonstrates the usefulness of our TS heuristic.

6 Conclusion

We propose in this paper a Tabu Search heuristic for a realistic capacitated network design problem where a minimum cost survivable network topology has to be designed taking into account capacity-dependent link costs as well as all other equipment costs (routers, line cards). The motivation for this study comes from the fact that the problem we deal with arises in the telecommunication industry.

Extensive numerical experiments are reported. TS heuristic has produced near-optimal solutions, while keeping reasonable computing times. TS has also outperformed a Greedy heuristic based on local search for problems with realistic sizes. This result underlines the benefits associated with the mechanisms at the core of TS to escape from local optima.

References


