Integration of Equipment Constraints in the Network Topology Design Process

Mohamed Zied BEN HAMOUDA, Olivier BRUN and Jean-Marie GARCIA
CNRS; LAAS; 7 avenue du Colonel Roche, F-31077 Toulouse, France
Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077 Toulouse, France
Email: {zied.ben-hamouda, brun, 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 designed by installing equipments (routers, line cards, ...) on nodes as well as link facilities on arcs. This realistic problem finds its motivation in the rapidly developing field of telecommunication networks and the introduction of fiber-optic technology. It is, in particular, faced by the network designer whenever a new network is set up.**

To the best of our knowledge, this is the first work that addresses this capacitated network design problem where equipments, such as routers and lines cards, have to be settled on network nodes. We present in this paper a comprehensive model to our network design problem. One exact algorithm and two heuristic approaches are proposed to solve this problem. Numerical results are provided for randomly generated networks and networks coming from real-word applications.

I. Introduction

Network operators as well as many governmental organizations operating in strategic areas (e.g. defense, air control, energy, ...) 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.

A. Previous Work

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], [10], [15]). 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, ...) 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 became cheaper and cheaper. As a consequence, equipment costs have became a significant fraction of the total cost when designing a network. There is therefore an increasing need for an integrated approach of 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], [11], [12], [13] 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].

B. 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 [14].

This paper proposes a comprehensive model as well as solution approaches 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.

C. Paper Outline

The remainder of the paper is structured as follows. Section II presents a formal description and definition of the capacitated network design problem we deal with in this paper. In sectionIII we introduce a detailed mathematical model of our design problem. We propose in sections IV and V one exact and two heuristic algorithms to solve our design model. Computational results are reported in Section VI on typical telecommunication data. We conclude this paper by discussing some related open problems and suggesting further research.

II. 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 demands 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.

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.

III. Notation and Problem Formulation

A. Topological constraints

The above network design problem can be more formally stated as follows. We are given an undirected graph $G = (V, E)$. We let $A \subset V$ be the subset of mandatory nodes. The edges of E represent possible communication links between nodes of V.

For each potential link $e \in E$, let us define the following 0-1 decision variable,

$$x_e = \begin{cases} 1 & \text{if link } e \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$

A solution to the network design problem is then a vector $\mathbf{x} = (x_e)_{e \in E}$. In graph-theoretic notations, a solution can be represented as a subgraph $G_{\mathbf{x}} = (V_{\mathbf{x}}, E_{\mathbf{x}})$ of G where $E_{\mathbf{x}} = \{e \in E \ : \ x_e = 1\}$ and $V_{\mathbf{x}}$ is the set of vertices of V which are endpoint of an edge $e \in E_x$.

1) Connectivity constraints: A feasible solution has to satisfy the following connectivity constraints:

$$\sum_{u \in W} \sum_{v \in V - W} x_{uv} \geq 2 \qquad W \subset V, W \cap A \neq \emptyset, A \qquad (1)$$

$$\sum_{u \in W} \sum_{v \in V - z - W} x_{uv} \geq 1 \quad z \in V, W \subset V - z, W \cap A \neq \emptyset, A \qquad (2)$$

Inequalities (1) are called cut inequalities, while inequalities (2) are called node cut inequalities. Using Menger's theorem, it is easy to show that any solution $\mathbf{x}$ satisfying inequalities (1) guarantee that removing an edge preserves connectivity. If it also satisfies inequalities (2), then the nodes of A are in the same two-connected component of $G_{\mathbf{x}}$.

2) Delay constraints: Moreover, a feasible solution has to satisfy additional delay constraints, i.e. path lengths have to be bounded by K_n and $K_f > K_n$ in nominal and failure states, respectively. Following the idea proposed in [3], these constraints can be formulated as linear constraints using a covering formulation. Let Π_{uv} (resp Π^z_{uv}) denotes the set of paths between u and v in G (resp. $G - z$) whose length is less than or equal to K_f (resp. K_n). Introducing the following binary variables,

$$y^\pi_{uv} = \begin{cases} 1 & \text{if all edges in path } \pi \text{ are included in } E_{\mathbf{x}} \\ 0 & \text{otherwise} \end{cases}$$

for all path π between u and v, the delay constraints can be formulated as follows,

$$y^\pi_{uv} \leq x_e \quad e \in \pi, \pi \in \Pi_{uv}, u, v \in A \qquad (3)$$

$$\sum_{\pi \in \Pi_{uv}} y^\pi_{uv} \geq 1 \qquad u, v \in A \qquad (4)$$

$$\sum_{\pi \in \Pi^z_{uv}} y^\pi_{uv} \geq 1 \quad z \in V, z \neq u, v, u, v \in A \qquad (5)$$

Inequalities (3) guarantee that a path is present in the solution only if its edges are present. Inequalities (4) state that the solution has to contain at least one path of Π_{uv} for any origin-destination pair $u, v \in A$. Finally, inequalities (5) states that the solution has to contain at least one path of Π^z_{uv} for any origin-destination pair $u, v \in A$, whatever the failed node $z \neq u, v$.

A feasible solution to our network design problem is a vector $\mathbf{x} = (x_e)_{e \in E}$ of 0-1 variables satisfying the connectivity constraints (1) and (2) and such that there exists binary

variables y^{π}_{uv} satisfying constraints (3)-(5). In the sequel, we let X denote the set of feasible solutions. We also let $\Pi_{uv}(\mathbf{x})$ denotes the set of path π between nodes u and v such that $y^{\pi}_{uv} = 1$.

B. Objective Function

Our network design problem is to find a minimum cost feasible solution $\mathbf{x} \in X$. The cost of a feasible solution is composed of link costs and equipment costs, as detailed below.

1) Link Load: Let us consider a feasible solution $\mathbf{x} \in X$. Let d_{uv} be the communication demand between nodes $u, v \in A$. We assume in our design model that this traffic is routed, along shortest paths, splitting flows at nodes where several outgoing links are on shortest paths to the destination. We define the length of a path $\pi \in \Pi_{uv}(\mathbf{x})$ as $|\pi| = \sum_{e \in \pi} x_e$, i.e. the length of a path present in the solution is its number of edges. Let $\overline{\Pi}_{uv}(\mathbf{x})$ be the set of shortest paths between u and v in the graph $G_{\mathbf{x}}$ and define for each $(i,j) \in E_{\mathbf{x}}$ and each $v \in A$,

$$\delta^v_{ij} = \begin{cases} 1 & \text{if there exists } \pi \in \overline{\Pi}_{iv}(\mathbf{x}) \text{ such that } (i,j) \in \pi \\ 0 & otherwise \end{cases}$$

The variables δ^v_{ij} indicates whether arc (i,j) is on a shortest path between nodes i and v. We also let $n^v_i = \sum_j \delta^v_{ij}$ be the number of shortest paths between these two nodes. The traffic received at node i for destination v can then be written as,

$$\gamma^v_i(\mathbf{x}) = d_{iv} + \sum_{j \neq v} \frac{\delta^v_{ji}}{n^v_j} \gamma^v_j$$

and the load of arc $(i,j) \in E_{\mathbf{x}}$ can be written as,

$$y_{i,j}(\mathbf{x}) = \sum_{v \in A} \frac{\delta^v_{ij}}{n^v_i} \gamma^v_i \tag{6}$$

2) Link costs: We assume that the cost of a link $(i,j) \in E_{\mathbf{x}}$ is an increasing function $F_{ij}()$ of its capacity. Link costs $F_{ij}()$ 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 $\beta^t_{ij}(\mathbf{x})$ be 1 if t is the minimum value such that $r_t \geq max(y_{i,j}(\mathbf{x}), y_{j,i}(\mathbf{x}))$, and 0 otherwise. The cost of link $(i,j) \in E_{\mathbf{x}}$ is then given by,

$$F_{ij}\left(\sum_{t=1}^{T} \beta^t_{ij}(\mathbf{x})\, r_t\right) \tag{7}$$

3) Equipment Cost: 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 ϕ_t be the cost of such a line card. Moreover, 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 $\overline{s}_r$ be the number of available slots, $\overline{T}_r$ be the maximum throughput of the router forwarding engine, and ψ_r be the cost of such a router.

The minimum number $p^t_v(\mathbf{x})$ of cards of type t required to support links connected to node v is given by,

$$p^t_v(\mathbf{x}) = \left\lceil \frac{\sum_{i \in V} \beta^t_{iv}(\mathbf{x})}{p_t} \right\rceil$$

where for any real-valued z, $\lceil z \rceil$ denotes the lowest integer n such that $z \leq n$. Let $\mu^*(\mathbf{p}_v(\mathbf{x}))$ denotes the optimal equipment cost for node $v \in V_{\mathbf{x}}$ in the solution $\mathbf{x} \in X$, where $\mathbf{p}_v(\mathbf{x})$ is the vector $\left[p^1_v(\mathbf{x}), \ldots, p^T_v(\mathbf{x})\right]$. The optimal equipment cost for node v is given by,

$$\mu^*(\mathbf{p}_v(\mathbf{x})) = \sum_{t=1}^{T} p^t_v(\mathbf{x})\, \phi_t \;+\; min_r \left\{\psi_r : \mathbf{p}_v(\mathbf{x}) \in \Lambda_r\right\} \tag{8}$$

where,

$$\Lambda_r = \left\{ \mathbf{s} = (s_1, \ldots, s_T) : \sum_t s_t \leq \overline{s}_r,\; \sum_t s_t\, p_t\, r_t \leq \overline{T}_r \right\}$$

is the set of all card configurations that can be accommodated by a router of type $r = 1, \ldots, R$. Note that the function μ^* can be efficiently calculated before-hand for each possible card configuration, i.e. for each possible value of the vector $\mathbf{p}_v(\mathbf{x})$.

C. Problem statement

The problem to be solved can now be stated as follows,

$$\min_{\mathbf{x} \in X} \Gamma(x) = \sum_{e \in E} x_e\, F_e\Big(\sum_{t=1}^{T} \beta^t_e(\mathbf{x})\, r_t\Big) \;+\; \sum_{v \in V_{\mathbf{x}}} \mu^*\left(\mathbf{p}_v(\mathbf{x})\right) \tag{9}$$

IV. An Branch-and-Bound Algorithm

In this section, we describe the details of the implementation of a Branch-and-Bound (BB) algorithm for the design problem stated before. Our aim is not to describe in detail the general BB paradigm, but to emphasize the problem-specific aspects of our algorithm.

The recursive algorithm is based on a binary search tree where nodes represent partial solutions. The root node of the tree is the solution $\mathbf{x} = (1)_{e \in E}$, i.e., the fully-meshed network. The algorithm takes as input the current partial solution $\mathbf{x}$, the next potential link l to be considered as well as an upper bound ub on the optimal cost, and performs a tree search.

A. Branching

The branching operation consists of dividing the search tree associated with the parent node $\mathbf{x}$ into two descendants or subtrees: the right one and the left one. The right subtree is rooted at $\mathbf{x}^{right}$ where $\mathbf{x}^{right}$ is the solution obtained by removing the link l from $E_{\mathbf{x}}$. The left subtree is rooted at $\mathbf{x}^{left}$ where $\mathbf{x}^{left}$ is just $\mathbf{x}$, i.e. all the links of $\mathbf{x}$ are kept in $\mathbf{x}^{left}$.

Algorithm 1 Branch-and-Bound algorithm

1: **procedure** BC($\mathbf{x}$, l, ub)
2: **if** $\mathbf{x} \notin X$ **or** $l = |E| + 1$ **then**
3: **return** ∞
4: **end if**
5: **if** $lb(\mathbf{x}) \geq ub$ **then**
6: **return** $\Gamma(\mathbf{x})$
7: **end if**
8: **if** $\Gamma(\mathbf{x}) < ub$ **then**
9: $ub = \Gamma(\mathbf{x})$
10: **end if**
11: $\mathbf{x}^{right} = x$; $x_l^{right} = 0$
12: rightCost = $BC(\mathbf{x}^{right}, l + 1, ub)$
13: $\mathbf{x}^{left} = \mathbf{x}$
14: leftCost = $BC(\mathbf{x}^{left}, l + 1, ub)$
15: **return** min($\Gamma(\mathbf{x})$, leftCost, rightCost)

B. Convergence Test

The algorithm is recursively called for $\mathbf{x}^{left}$ and $\mathbf{x}^{right}$ to compute the minimum cost of feasible solutions in the subtree rooted at node $\mathbf{x}$. The recursion is stopped when either all links have been considered ($l = |E| + 1$) or the separation test is satisfied.

C. Separation Test

A search subtree is pruned when it cannot lead to a solution that will be better than the best solution found so far. A search tree rooted at $\mathbf{x}$ is pruned if one of the following statements is true:

- $\mathbf{x}$ is not feasible. (It is clear that if a solution $\mathbf{x}$ is not feasible, all the solutions in the subtree rooted at $\mathbf{x}$ are not feasible too).
- the lower bound of $\mathbf{x}$ exceeds the current upper bound.

D. Upper and Lower Bounds Generation

Upper and a lower bounds are used to prune the solution tree. The upper bound is initialized in our algorithm with the cost returned by the Greedy heuristic to be described later, and it is updated each time an improving feasible solution is discovered.

Let us consider a feasible solution $\mathbf{x}$ of the search tree obtained by deciding the values of the x_e for $e < l$. Let $Subtree(\mathbf{x})$ be the set of feasible solutions in the subtree rooted at $\mathbf{x}$. Let $F = \{e < l : x_e = 1\}$ be the set of links already selected and $F' = \{e > l\}$ be the set of potential links that may be selected (i.e., links not yet considered). It is clear that for any solution $\mathbf{x}' \in Subtree(\mathbf{x})$, we will have $F \subset E_{\mathbf{x}'}$. Let $G[F]$ be the subgraph of $G_{\mathbf{x}}$ induced by the links of F. Clearly, if $G[F]$ does not satisfy the connectivity constraints (1)-(2), then some links of F' will also have to be selected, i.e. $F' \cap E_{\mathbf{x}'} \neq \emptyset$.

A lower bound on the cost of any solution in $Subtree(\mathbf{x})$ has therefore three components:

- a lower bound on the cost of the links in F. Observe that solution $\mathbf{x}'$ is obtained by removing some links in $G_{\mathbf{x}}$. Hence, the load of a link belonging to F can only increase with respect to its load in solution $\mathbf{x}$. A lower bound on the costs of the links in F is thus obtained by computing the cost of these links using their loads in $\mathbf{x}$.
- If $G[F]$ does not satisfy the connectivity constraints (1)-(2), then some links of F' have to be selected. The minimum number n of edges to be added can easily be computed using the algorithm proposed in [16]. A lower bound on the cost of these links is thus obtained by considering the n cheapest links in F' such that a) the endpoints of these links do not belong to the same biconnected component of $G_{\mathbf{x}}$ and b) none of the endpoints correspond to a cut-node of $G_{\mathbf{x}}$. The cost of these links is computed using their loads in $\mathbf{x}$.
- a lower bound on the cost of the line cards and routers to be settled with solution $\mathbf{x}'$. Such a lower bound is obtained by taking into account only the links of F, with the capacity they have in solution $\mathbf{x}$.

E. Value Ordering and Variable Ordering

Let us consider a partial solution $\mathbf{x}$. The order in which the subtrees rooted at $\mathbf{x}$ are considered is important since an efficient value-ordering scheme can enable to improve the upper bound faster and thus to prune the search tree much more.

As the cost of $\mathbf{x}$ and $\mathbf{x}^{right}$ are different, a natural ordering scheme is to consider the subtree rooted at $\mathbf{x}^{right}$ before that rooted at $\mathbf{x}^{left}$.

V. Heuristic Approaches

In this section, we describe two heuristics used to generate feasible solutions.

A. Tabu Search Heuristic

Tabu Search (TS) is a well known technique for combinatorial optimization [8][9]. It has proved its effectiveness in many combinatorial optimization problems, and in particular for the design of survivable networks [3]. In the following, we describe how we adapted the ideas of TS to our network design problem. A complete description of the TS algorithm is given in algorithm 2. We describe below the details of this algorithm.

1) Initialization: Our TS algorithm starts from an initial feasible solution $\mathbf{x_0}$ and generate a sequence $\mathbf{x_0}, \mathbf{x_1}, \ldots$ of feasible solutions. The initial solution $\mathbf{x_0}$ can be obtained, for instance, with the Greedy algorithm described in [3]. This algorithm belongs to the class of local search methods and 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 while preserving feasibility.

2) Search space and Neighborhood structure: The neighborhood structure is the most important issue in the development of a TS heuristic. Let us consider a minimal feasible solution $\mathbf{x_i}$ at iteration i. We mean by the term "minimal" that the removal of any edge from $\mathbf{x_i}$ makes it not feasible or increases its total cost. If a better minimal solution $\mathbf{x_{i+1}}$ exists, it is easy to see that $\mathbf{x_{i+1}}$ contains at least one edge which is not in $\mathbf{x_i}$.

Based on this observation, we define the neighborhood $\mathcal{N}(\mathbf{x_i})$ of $\mathbf{x_i}$ as follows. $\mathcal{N}(\mathbf{x_i})$ contains all the feasible solutions that can be obtained from $\mathbf{x_i}$ by a simple cut-and-paste operation on the graph $G_{\mathbf{x_i}}$, that keeps the topology feasible, namely: a) add an arc e to $G_{\mathbf{x_i}}$ between endpoints which are not isolated, and then b) cut a set of arcs $f \neq e$ from the graph just formed using the greedy algorithm.

3) Diversification: As its name suggests, a diversification strategy is designed to drive the search into new regions and to generate solutions that differ in various significant ways from those seen before. Diversification strategy is particularly helpful when better solutions can be reached only by crossing "barriers" in the solution space. In particular, it can be used when the search is trapped in a local minimum.

The concept of Diversification can easily be adapted to our problem. Let $\mathbf{x_i}$ be the best solution at iteration i and assume $\mathbf{x}_i$ is a local minimum with respect to the neighborhood structure $\mathcal{N}$. Based on the observation that, in practice, a potential transit node is generally used in a telecommunication network if and only if it reduces the total network cost, we define a new neighborhood $\mathcal{N}'(\mathbf{x_i})$ of $\mathbf{x_i}$ by applying the two following operations: a) add a potential node to $G_{\mathbf{x_i}}$ and connect it to the other used nodes, and then b) apply the greedy algorithm.

At iteration i, if no solution improving $\mathbf{x_i}$ is found in $\mathcal{N}(\mathbf{x_i})$, the algorithm diversifies the search space by exploring solutions in $\mathcal{N}'(\mathbf{x_i})$.

4) Tabu List: To avoid a risk of cycling, we used a memory (a tabu list T) to forbid moves that might lead to recently visited solutions. We give a tabu status to the arcs that were added to define the next iterate, and we first try to remove arcs that are not tabu in the greedy step. If an arc which is tabu was removed, the solution is accepted only if it improves the network cost. This is the only aspiration criterion we use.

B. LDS Heuristic

The concept of *Limited Discrepancy Search* (LDS) was initially proposed by Harvey and Ginsberg for Constraint Satisfaction Problem (CSP) [17]. Assume that we know an efficient heuristic to solve a CSP. The heuristic will lead directly to a solution most of the time, but it may also fails. The intuition behind LDS is that, when the heuristic fails, the heuristic probably would have lead to a solution if and only if it had not made one or two "wrong" decisions that got it off track. The idea is then to systematically follow the heuristic at all but a limited number of decision points (which are called "discrepancies").

Algorithm 2 Tabu Search Algorithm

Require: an initial feasible solution $\mathbf{x_0}$
1: $\mathbf{x_i} = \mathbf{x_0}$; $\mathbf{x}^* = \mathbf{x_0}$; $tabuList = \emptyset$
2: **while** the stopping criterion is not satisfied **do**
3: select the best solution $x \in \mathcal{N}(\mathbf{x_i})$ such that x in not tabu or x satisfies the aspiration criterion
4: **if** x does not improve $\mathbf{x_i}$ **then**
5: select the best solution $x' \in \mathcal{N}'(\mathbf{x_i})$ such that x' in not tabu or x' satisfies the aspiration criterion
6: **end if**
7: $\mathbf{x_{i+1}} = min(x, x')$
8: update $tabuList$
9: **if** $\Gamma(\mathbf{x_{i+1}}) < \Gamma(\mathbf{x}^*)$ **then**
10: $\mathbf{x}^* = \mathbf{x_{i+1}}$
11: **end if**
12: **end while**

Router Model	# Slots	Throughput (Gbps)	Cost (K€)
A	12	3	5
B	10	5	10
C	8	10	15

TABLE I
ROUTER MODELS.

The concept of LDS can easily be adapted to our problem. Consider a feasible solution $\overline{\mathbf{x}} \in X$. Such a solution can be obtained, for example, by the Greedy heuristic or the TS heuristic described earlier. Let $\mathbf{x} \neq \overline{\mathbf{x}}$ be another feasible solution. We define the distance between $\mathbf{x}$ and $\overline{\mathbf{x}}$ as follows,

$$d(\mathbf{x}, \overline{\mathbf{x}}) = \sum_{e \in E} |x_e - \overline{x}_e| \tag{10}$$

For a given integer parameter β, let $X_\beta(\overline{\mathbf{x}})$ be the subset of feasible solutions $\mathbf{x} \in X$ such that $d(\mathbf{x}, \overline{\mathbf{x}}) \leq \beta$. The proposed heuristic is a slight modification of algorithm 1. As algorithm 1, this algorithm performs a tree search, but instead of exploring the whole solution space X, it explores only the region $X_\beta(\overline{\mathbf{x}})$. A subtree is not explored if it is rooted at a partial solution $\mathbf{x}$ such that $d(\mathbf{x}, \overline{\mathbf{x}}) > \beta$.

VI. COMPUTATIONAL RESULTS

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 I, II and III. Tests were made for random problems, with vertices uniformly generated in a square of size 200 $Km \times 200$ Km. Traffic demand between each pair of mandatory nodes varies from 1 to 20 $Mbps$.

Figure 1 shows the relative gaps in terms of network cost between the optimal solution on the one hand and TS, TS+LDS and Greedy heuristics on the other hand. Figure 2 portrays the computing times we obtained.

Line Card Model	# Ports	Throughput (Mbps)	Cost (K€)
1	12	12 × 20	2
2	8	8 × 50	3
3	4	4 × 100	5
4	2	2 × 1000	8

TABLE II
LINE CARD MODELS.

Capacity (Mbps)	20	50	100	1000
Cost/Km per year (K€)	0.3	0.4	0.5	0.6

TABLE III
LINK COSTS.

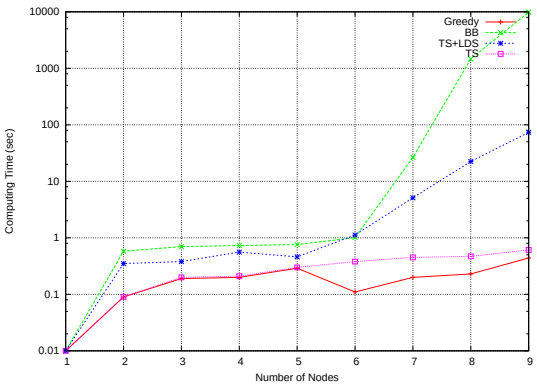

Fig. 2. Computing Times

It can be noticed that, the BB algorithm is applicable only to small problem instances. This is due to the fact that the nature of the design problem we address in this paper is NP-hard, as it contains the fixed charge network design problem (and thus, the Steiner tree problem) as a special case. As a consequence, we can solve optimally only small size instances of the problem, and the design of efficient heuristics is of paramount importance for dealing with problems of larger sizes.

The TS heuristic allows to keep reasonable computing times, while providing near-optimal solutions. These solutions are always improved by LDS heuristic to obtain a near-optimal design. These results underline the benefits associated with the mechanisms at the core of the TS and LDS algorithms introduced in this paper, namely their abilities to design networks coming from real-word applications.

VII. CONCLUSION

We propose in this paper a comprehensive model 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). A BB algorithm as well as TS and LDS heuristics have been proposed to solve this problem. Extensive numerical experiments are reported. Numerical results show that the BB algorithm is applicable only to small problem instances. TS heuristic has produced near-optimal solutions, while keeping reasonable computing times. These results underline the benefits associated with the mechanisms at the core of the TS algorithm introduced in this paper, namely its ability to design networks coming from real-word applications.

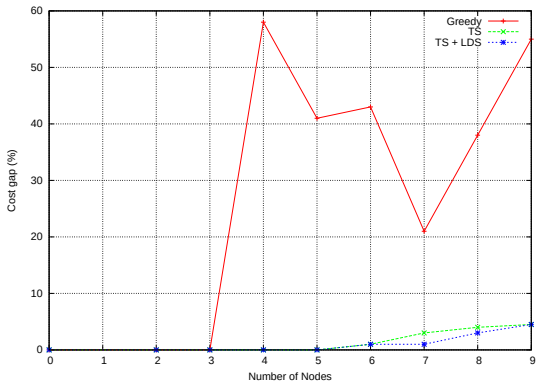

Fig. 1. Gaps Between the BB algorithm and the heuristic algorithms

REFERENCES

[1] A. Balakrishnan, T. L. Magnanti, A. Shulman and R. T. Wong (1989), *A Dual-Ascent Procedure for Large-Scale Uncapacitated Network Design*, Operation Research.

[2] A. Frangioni, B. Gendron (2008) *0-1 Reformulations of the Multicommodity Capacitated Network Design Problem*, Discrete Applied Mathematics.

[3] B. Fortz and M. Labbé and F. Maffioli (2000), *Solving the Two-Connected Network with Bounded Meshes Problem*, Operations Research.

[4] B. Gavish and A. Altinkemer (1990), *Backbone Network Design Tools with Economic Tradeoffs*, ORSA J. on Computing.

[5] B. Gendrom, T.G. Crainic and A. Frangioni (1999), *Multicommodity Capacited Network Design*, in: P. Soriano, B. Sanso (Eds.), Telecommunications Network Planning, Kluwer Academics Publisher.

[6] D. Bienstock and O. Gunluk (1996), *Capacitated Network Design-Polyhedral Structure and Computation*, INFORMS J. on Computing.

[7] F. Barahona (1996), *Network Design Using Cut Inequalities*, SIAM J. on Optimization.

[8] F. Glover (1989), *Tabu Search–Part I*, ORSA J. on Computing.

[9] F. Glover (1990), *Tabu Search–Part II*, ORSA J. on Computing.

[10] M. Minoux (1989), *Network Synthesis and Optimum Network Design Problems: Models, Solution Methods and Applications*, Networks.

[11] T. L. Magnanti and P. Mirchandani (1993), *Shortest Paths, Single Origin-Destination Network Design, and Associated Polyhedra*, Networks.

[12] T.L. Magnanti, P. Mirchandani and R. Vachani (1993), *The Convex Hull of Two Core Capacitated Network Design Problems*, Mathematical Programming.

[13] T.L. Magnanti, P. Mirchandani and R. Vachani (1995), *Modeling and Solving the Two-Facility Capacitated Network Loading Problem*, Operations Research.

[14] T.L. Magnanti, P. Mirchandani and R. Vachani (1991), *Modeling and Solving the Capacitated Network Loading Problem*, Operations Research.

[15] T.L. Magnanti and R.T. Wong (1984), *Network Design and Transportation Planning: Models and Algorithms*, Transportation Science.

[16] T. Mashima et al. (2006), *Bi-Connectivity Augmentation for Specified Vertices of a Graph with Upper Bounds on Vertex-Degree Increase*, IEICE-Transactions on Information and Systems.

[17] W. D. Harvey, M. L. Ginsberg (1995), *Limited Discrepancy Search*. The 14th Int. Conf. on Artificial Intelligence.