AN APPROACH TO ROBUST NETWORK DESIGN IN TELECOMMUNICATIONS

GEORGIOS PETROU¹, CLAUDE LEMARÉCHAL² AND ADAM OUOROU¹

Abstract. In telecommunications network design, one of the most frequent problems is to adjust the capacity on the links of the network in order to satisfy a set of requirements. In the past, these requirements were demands based on historical data and/or demographic predictions. Nowadays, because of new technology development and customer movement due to competitiveness, the demands present considerable variability. Thus, network robustness w.r.t demand uncertainty is now regarded as a major consideration. In this work, we propose a min-max-min formulation and a methodology to cope with this uncertainty. We model the uncertainty as the convex hull of certain scenarios and show that cutting plane methods can be applied to solve the underlying problems. We will compare Kelley, Elzinga-Moore and bundle methods.

Keywords. Telecommunications network design, robust optimization, min-max-min problems, cutting plane methods.

Mathematics Subject Classification. 65K05, 90C26, 90B12.

1. INTRODUCTION

In the present competitive markets, robustness is regarded as a major consideration in telecommunication networks. The demand presents considerable variability

¹ France Télécom Division R&D, MCN-OTT, 38-40 rue du Général Leclerc, 92794 Issy-Les-Moulineaux Cedex 9, France; {georgios.petrou, adam.ouorou}@orange-ftgroup.com
² Inria, 655 avenue de l’Europe, Montbonnot, 38334 Saint Ismier, France;
Claude.Lemarechal@inrialpes.fr

© EDP Sciences, ROADEF, SMAI 2007

Article published by EDP Sciences and available at http://www.rairo-ro.org or http://dx.doi.org/10.1051/ro:2007033
because of customers movement, introduction of new services and product development. Historical data for predicting the future is inadequate since the area is constantly changing or because there is no history for new services. In this work, we study the problem of assigning capacity to the links of a network in order to satisfy a set of requirements. We consider that the demand is an uncertain parameter and propose a new formulation and a methodology to cope with this uncertainty. To model the uncertainty, the demand is assumed to belong to an uncertainty set and we seek for a capacity assignment of the links that is good enough for every possible demand in the uncertainty set. Thus, our approach fits in the framework of robust optimization as termed in [2,16]. We analyze the case where the uncertainty set is the convex hull of given demand scenarios. Our approach can be viewed as worst-case oriented since we consider a min-max-min criterion which will result in conservative decisions, with the purpose of contributing to the learning process of decision makers by seeking solutions that hedge against demand uncertainty.

The paper is organized as follows. After reviewing the literature in Section 2, we present our min-max-min formulation for the general case in Section 3. In Section 4, we introduce the resolution procedures when the uncertainty set is described by the convex hull of a reasonable number of scenarios. The computational results are reported in Section 5, and Section 6 concludes this study.

2. RELATED WORKS

Stochastic programming and robust optimization are mathematical tools to deal with uncertainty. They have been both used for telecommunication network design under uncertainty.

A stochastic approach for the network design problem with uncertain demand is presented in [23]. Sen et al. deal with the problem under the budget constrained approach. They formulate a two-stage stochastic linear program and they solve it with the use of stochastic decomposition [10]. This method is based on sampling which has asymptotic properties and a statistically motivated stopping rule. The first stage problem minimizes the expected unserved demands and provides the capacity of the graph, while the second stage problem strives for an efficient use of the capacity. The model is validated with the use of simulation, and it is also noted that the proposed solution is better than replacing the uncertain variables by their expected values.

Another stochastic optimization method is presented by Lisser et al. [19]. Scenario representation is used in order to formulate the uncertainty of the demand, and every scenario corresponds to a given probability for the possible realization. The resulting non-smooth problem is solved with the analytic center cutting plane method [9]. The subproblem is decomposed into multicommodity flow problems by demand scenarios. The authors take advantage of this formulation by using parallel computing.
Italiano et al. [12] are concerned with the problem of reserving capacity in a network in order to satisfy pairwise demands. To remove uncertainty of the demands they use terminal nodes and bounds for the traffic between the terminals. This model is known as hose model. The solution has to satisfy any traffic matrix for the demand, and the capacity vector should form a tree. Hence, the flows are not split, and for any traffic matrix the flow of every demand uses the same path.

Duffield et al. [6] have studied a similar problem, but they concentrate on Virtual Private Networks (VPN) based on the hose model. Under this concept they take into account only the ingress and the egress traffic for each endpoint of the VPN, which is more easily specified compared to the conventional point-to-point approach, and Steiner trees are employed in order to connect the VPN endpoints. However, they consider the capacity assignment on a pre-existing network, which is dynamically resized.

Another study in the framework of the hose model is done by Kumar et al. [17]. They propose two new methods for solving the problem of the VPN tree computation: the breath-first search algorithm (BFS) and a primal-dual algorithm. They compare these two methods in terms of cost and computational time, with the Steiner tree which is given as solution in [6]. The BFS algorithm was developed initially for the case of symmetric ingress and egress bandwidths and it computes the optimal tree in polynomial time. In the general case the problem is NP-hard, and both algorithms outperform the Steiner tree method.

Ben-Ameur and Kerivin [1] consider VPN networks with a polyhedral model for the demand uncertainty. Their mathematical formulation is built on the arc-path flow formulation and a cutting plane method is devised to deal with the large size of the problem. There are two procedures which identify violated inequalities for the master problem. These inequalities are based on unsatisfied demands and improvement of the value of the objective function by selecting sequentially different paths. The final solution specifies a capacity vector, the supporting paths of the demands, and the splitting coefficients which are independent of the possible values of the demands.

In [22] Ouorou proposes three models for a robust capacity assignment in telecommunication networks with demand uncertainty in the framework of robust optimization as defined by Kouvelis and Yu [16]. The algorithmic solutions are based on cutting plane methods. Some computational experiments indicate that the Elzinga-Moore cutting plane method [7] can be a more valuable choice when compared with Kelley’s method [13]. Since different possible uncertainty sets may exist in some circumstances, a generalization of these models is proposed in order to cope with a finite number of plausible uncertainty sets, and a weight is associated with each uncertainty set to determine its relative importance or worth.

3. Problem formulation

We follow the approach introduced in [20] by Lucertini and Paletta, who studied a similar problem in which the investment for installing the capacity is limited by
a fixed budget. Suppose that we have a telecommunication network represented by a graph \( G = (V,E) \) where \( V \) is the set of nodes (e.g. terminals) and \( E \) is the set of edges (e.g. optical links). We also have a set \( K \) of pairs of nodes, which are defined as origin-destination pairs (OD-pairs). The commodities that have to be transferred between the OD-pairs are not given but they belong to an uncertainty set \( \Omega \). Let \( \Gamma(x) \) be the set of all possible demand vectors which can be satisfied given a capacity vector \( x \). A robust decision can be obtained by minimizing a function \( \phi(\Omega, \Gamma(x)) \) with respect to the capacity vector \( x \), where \( \phi \) is an appropriate measure of the set of unsatisfied demands. In this study we suppose that the feasible capacities are continuous, as in [23]. If \( c \in \mathbb{R}^{|E|}_+ \) is the cost vector, then

\[
X := \left\{ x \in \mathbb{R}^{|E|}_+ : c^T x \leq B \right\}
\]

(1)

will denote the set of feasible capacities available for the network, under a given budget \( B \). The problem we consider is as follows:

\[
\min_{x \in X} \phi(\Omega, \Gamma(x)).
\]

To define the measure \( \phi \), we introduce a penalty function \( f(\omega, \gamma) \), which expresses the gap between a demand vector \( \omega \) that the graph has to support and a demand vector \( \gamma \) that the graph can support. For any capacity vector \( x \in X \), we set

\[
\phi(\Omega, \Gamma(x)) := \max_{\omega \in \Omega} \min_{\gamma \in \Gamma(x)} f(\omega, \gamma)
\]

and we formulate the problem as

\[
\min_{x \in X} \phi(\Omega, \Gamma(x)), \quad \text{i.e.} \quad \min_{x \in X} \max_{\omega \in \Omega} \min_{\gamma \in \Gamma(x)} f(\omega, \gamma).
\]

(2)

There are several ways to choose the function \( f \). In this study, we consider that it is suitable to take as penalty function the weighted sum of unsatisfied commodities. So, if \( \pi_k \) is the penalty for not satisfying one unit of demand for the \( k \)th OD-pair, then we set \( f \) to be

\[
f(\omega, \gamma) := \sum_{k \in K} \pi_k (\omega_k - \gamma_k)^+,
\]

where \((y)^+ := \max(0,y)\). As for \( \Gamma(x) \), we introduce the following notation: \( P_k \) is the set of the possible paths considered to support the demand between the \( k \)th OD-pair, \( z_{k,p} \) is the flow of path \( p \in P_k \), \( \gamma \) is the vector of demands which can be carried through the graph given the capacity vector \( x \). Then,

\[
\Gamma(x) = \left\{ \gamma \in \mathbb{R}^{|K|}_+ : \exists z \in \mathbb{R}^{|P|}_+ : \sum_{k \in K} \sum_{p \in P_k} \sum_{j \in E} z_{k,p} \leq x_j, \forall j \in E, \sum_{p \in P_k} z_{k,p} = \gamma_k, \forall k \in K \right\}.
\]

The first constraint implies that the total flow on every path does not exceed the corresponding capacity, and the second constraint indicates that the vector of demands \( \gamma \) is supported from the flow \( z \).
Introducing $\delta_k := (\omega_k - \gamma_k)^+$ for every $k \in K$, problem (2) can be rewritten as:

$$\min_{x \in X} \max_{\omega \in \Omega} \min_{\gamma, \delta, z} \sum_{k \in K} \pi_k \delta_k$$

s.t. $$\begin{align*}
\sum_{k \in K} \sum_{p \in P_k} z_{k,p} &\leq x, j \in E, \\
\sum_{p \in P_k} z_{k,p} &= \gamma_k, k \in K, \\
\delta_k &\geq \omega_k - \gamma_k, k \in K, \\
\gamma, \delta, z &\geq 0,
\end{align*}$$

or in a compact form as

$$\min_{x \in X} \max_{\omega \in \Omega} \min_{\gamma, \delta, X} \pi^T \delta$$

s.t. $$\begin{align*}
Az &\leq x, \\
Fz &= \gamma, \\
\delta &\geq \omega - \gamma, \\
\gamma, \delta, z &\geq 0,
\end{align*}$$

(3)

where $A$ and $F$ are suitable 0-1 matrices. This formulation is defined as arc-path formulation.

**Remark.** Since we assume that the supporting paths for the commodities are not restricted, we consider in our implementation the following equivalent formulation:

$$\min_{x \in X} \max_{\omega \in \Omega} \min_{\gamma, \delta, X} \sum_{k \in K} \pi_k \delta_k$$

s.t. $$\begin{align*}
\sum_{k \in K} (X_{j}^{k+} + X_{j}^{k-}) &\leq x, j \in E, \\
A(X^{k+} - X^{k-}) &= \gamma_k l_k, k \in K, \\
\delta_k &\geq \omega_k - \gamma_k, k \in K, \\
\gamma, \delta, X &\geq 0,
\end{align*}$$

where $A$ is the node-arc incidence matrix of $G$, $X^{k+}$ and $X^{k-}$ are the “positive” and the “negative” flow vectors – respectively – which are used to satisfy commodity $k$, and $l_k$ is the vector of $\mathbb{R}^{|V|}$ defined by

$$(l_k)_i = \begin{cases} 
-1, & \text{if } i \text{ is the } k^{th} \text{ origin node,} \\
1, & \text{if } i \text{ is the } k^{th} \text{ destination node,} \\
0, & \text{otherwise.}
\end{cases}$$

This formulation is defined as node-arc formulation.

**4. Solution methods**

Problem (3) is difficult because of non-convexity (due to the internal min) and nonsmoothness (due to the intermediate max). We refer to [5] for a discussion
of these difficulties. However, an algorithm can be constructed for the solution of (3) when Ω is a bounded polyhedron with a limited number of extreme points. Thus we limit our study to the case where Ω is defined as the convex hull of \(|S|\) scenarios \(ω^1, \ldots, ω^{|S|}\) of possible demands; then the extreme points of Ω will be among these scenarios.

For fixed \(x\) and \(ω\) in (3), the inner linear minimization problem is compactly written as

\[
\min_{δ, z} \pi^T δ \\
\text{s.t.} \quad Az \leq x, \\
\delta \geq ω - Fz, \\
δ, z \geq 0,
\]

whose dual problem is

\[
P(x, ω) := \max_{u, v} ω^T u - x^T v \\
\text{s.t.} \quad F^T u \leq A^T v, \\
u \leq π, \\
u, v \geq 0.
\]

We have substituted \(Fz\) for \(γ\) and eliminated the resulting constraint \(Fz \geq 0\) since \(z \) and \(F\) are positive by definition. By LP duality the above two programs have the same optimal value. Thus (3) is equivalent to solving

\[
\min_{x \in X} \max_{ω \in Ω} P(x, ω).
\]

Note that \(P(x, ω)\) is easily computed for every \((x, ω)\) through a linear maximization program.

Setting

\[Q(x) := \max_{ω ∈ Ω} P(x, ω)\]

we have to minimize \(Q\) over \(X\) and the next result gives crucial properties of function \(Q\).

**Theorem 1.** Function \(Q\) over \(X\) and we have

\[Q(x) = \max_{i=1,2,...,|S|} P_i(x), \quad \text{where} \quad P_i(x) := P(x, ω^i), \; i = 1, 2, \ldots, |S|.\]  

**Proof.** Let \(g_{u,v}: X × Ω → \mathbb{R}\) be the linear function \(g_{u,v}(x, ω) = u^T ω - v^T x\), where \((x, ω) \in X × Ω\) and \((u, v) \in U × V = \{(u, v) : F^T u \leq A^T v, \; u \leq π, \; u, v \geq 0\}\).

We have that \(X × Ω \neq \emptyset\) for every \((u, v) \in U × V\), so \(g_{u,v}\) is well-defined. Moreover \(g_{u,v}\) is convex as a linear function, and for an arbitrary \((x', ω') \in (X, Ω)\) we have:

\[
\max_{(u,v) \in U × V} g_{u,v}(x', ω') \leq \sum_{k \in K} π_k ω'_k < +∞.
\]
So, if we rewrite $P$ as

$$P = \max_{(u,v) \in U \times V} g_{u,v},$$

then we deduce that $P$ is convex as the maximum of a family of convex functions (see [11], Prop. IV.2.1.2).

Now, the convex function $P(x, \cdot)$ attains its maximum at some extreme point of the set $\Omega$ (see [11], Prop. III.2.4.6), and these extreme points are among the $|S|$ possible scenarios. Hence:

$$\max_{\omega \in \Omega} P(x, \omega) = \max \left\{ P(x, \omega^1), P(x, \omega^2), \ldots, P(x, \omega^{|S|}) \right\}.$$

This establishes (5), and also shows that $Q(x) < +\infty$ for all $x \in X$. Thus $Q$ is convex for the same reason as $P$. □

Then the convex nonsmooth problem

$$\min_{x \in X} Q(x)$$

is equivalent to (3).

We will analyze three cutting plane algorithms for the solution of (6). The basic idea underlying these algorithms can be described as follows. Having computed $t$ sets of $|S|$ values $P_i(x_k)$, $i = 1, \ldots, |S|$ and corresponding subgradients $g_{ki} = -v_{ki}$ (see Th. 2 below) for $i = 1, \ldots, |S|$ and $k = 1, \ldots, t$, the true function $Q$ is approximated (from below) by the polyhedral function

$$x \mapsto \max \left\{ P_i(x_k) + g_{ki}^T(x - x_k) : i = 1, \ldots, |S|, k = 1, \ldots, t \right\}.$$

(7)

A cutting plane method uses this function to select a new point $x_{t+1}$, so as to improve the current approximation (7). The way this point is chosen determines whether and how fast the algorithm converges.

**Theorem 2.** For fixed $x_0 \in X$ and $i \in \{1, \ldots, |S|\}$, let $(u_{0i}, v_{0i})$ be an optimal solution of $P_i(x_0)$. Then $-v_{0i} \in \partial P_i(x_0)$.

**Proof.** The theorem results from elementary convex calculus, but a simple direct proof can be given:

If $(u_{0i}, v_{0i})$ is an optimal solution of the mathematical programming problem $P(x_0, \omega^i)$ of (4), then $P_i(x_0) = u_{0i}^T \omega^i - v_{0i}^T x_0$. Let $(u_{si}, v_{si})$ be the optimal solution of $P_i(x)$ for an arbitrary $x \in X$. Since the constraints of (4) do not depend on $x$, $(u_{0i}, v_{0i})$ is feasible for problem $P(x, \omega^i)$ and

$$P_i(x) = u_{si}^T \omega^i - v_{si}^T x \geq u_{0i}^T \omega^i - v_{0i}^T x_0$$

$$= P_i(x_0) - v_{0i}^T(x - x_0), \quad \forall x, x_0 \in X, i \in \{1, \ldots, |S|\}.$$  

Hence $-v_{0i} \in \partial P_i(x_0)$. □

We consider and compare the three following cutting plane methods, which differ in the way they construct the trial points $x_k$. They all use an oracle which,
at a given $x$, solves (4) $|S|$ times to compute the $P_i(x)$’s and then $Q(x)$, as well as the optimal $v_i$’s to provide the subgradients stipulated in Theorem 2. The optimal $v_i$’s computed at $x = x_k$ will be denoted by $v_{ki}$ for $i = 1, \ldots, |S|$.

4.1. Kelley cutting plane method

This is one of the first cutting plane methods proposed in the literature. The next trial point $x_{t+1}$ is obtained by solving the following relaxed problem (Master Problem) of (6):

$$
\begin{align*}
\min_{x, \bar{Z}} & \quad \bar{Z} \\
\text{s.t.} & \quad \bar{Z} \geq P_i(x_k) - v_{ki}^T(x - x_k), \ i = 1, 2, \ldots, |S|, \ k = 0, 1, \ldots, t, \\
& \quad c^T x \leq B, \\
& \quad x \geq 0.
\end{align*}
$$

The method iterates until a “good” approximation of the solution is found, and the resulting algorithm can be stated as follows:

**Kelley Algorithm (KA)**

1. **Step 0** Initialize $x_0 = 0$, $\varepsilon > 0$, $\bar{Z} = +\infty$ and $t = 0$.
2. **Step 1** Call the oracle at $x_t$ to obtain $P_i(x_t)$ and $v_{ti}$ for $i = 1, \ldots, |S|$, and $Q(x_t)$.
3. **Step 2** Let $\bar{Z} = \min \{\bar{Z}, Q(x_t)\}$.
4. **Step 3** Solve (8) and let $(x_{t+1}, \bar{Z})$ be its optimal solution.
5. **Step 4** If $\bar{Z} - \bar{Z} \leq \varepsilon (1 + |\bar{Z}|)$, then stop. Otherwise set $t = t + 1$ and loop to Step 1.

This procedure converges, and the proof is given in [13]. Note that $X$ is bounded, so (8) has always an optimal solution. This substantially simplifies the issue (see [11], Th. XII.4.2.3). However, if convergence is slow, then the Master Problem can be uncomfortably large. For further discussion see Sections 5 and 6.

4.2. Elzinga-Moore cutting plane method

This method is based on the following result by Nemhauser and Widhelm [21].

**Theorem 3.** Given a bounded, non-empty polyhedron described from a set of linear inequalities,

$$
\alpha_j^T x \geq \beta_j, \ j = 1, \ldots, p,
$$

the optimal solutions in $\sigma$ and $x$ of the problem

$$
\max_{\sigma, x} \sigma \\
\text{s.t.} \quad \alpha_j^T x - ||\alpha_j|| \sigma \geq \beta_j, \ j = 1, \ldots, p,
$$

are respectively the radius and the center of the largest sphere inscribed in the polyhedron.
Proof. See [21].

Naturally, this result has a meaning only for a bounded polyhedron. Now consider in the \((x, Z)\)-space the polyhedron defined by the constraints in (8) – i.e. the epigraph of the current approximation (7). It is unbounded but it can be truncated from above by appending the constraint \(Z \leq \overline{Z}\). The polyhedron thus obtained clearly contains any optimal solution \((x^*, Q(x^*))\) of (6): the center revealed by Theorem 3 may be deemed an adequate approximation of such an optimal solution. This is the rationale for Elzinga-Moore cutting plane method, which computes the next trial point by solving

\[
\begin{align*}
\max_{x, \sigma, Z} & \quad \sigma \\
\text{s.t.} & \quad Z + \sigma \leq \overline{Z} \\
& \quad \mathcal{P}_i(x_k) - v_{ki}^T(x - x_k) - Z + (\|v_{ki}\|^2 + 1)^{1/2} \sigma \leq 0, \quad i = 1, 2, \ldots, |S|, \quad k = 0, 1, \ldots, t, \\
& \quad c^T x \leq B, \\
& \quad x \geq 0.
\end{align*}
\]

Note again that this linear program has a non-empty bounded feasible set, and therefore an optimal solution \((x_{t+1}, \sigma_{t+1}, Z_{t+1})\) (with \(\sigma_{t+1} > 0\)). This method has little usage in the literature even though it is not much harder to be implemented than Kelley’s method. It was shown to be more efficient than Kelley’s for some min-max problems, see [22]. Elzinga-Moore cutting plane algorithm for the solution of (6) is as follows:

**Elzinga-Moore Algorithm (EMA)**

1. **Step 0** Initialize \(x_0 = 0, \varepsilon > 0, \overline{Z} = +\infty\) and \(t = 0\).
2. **Step 1** Call the oracle at \(x_t\) to obtain \(\mathcal{P}_i(x_t)\) and \(v_{ti}\) for \(i = 1, \ldots, |S|\), and \(Q(x_t)\).
3. **Step 2** Let \(\overline{Z} = \min \{\overline{Z}, Q(x_t)\}\).
4. **Step 3** Solve (9) to get \(x_{t+1}\) and \(\sigma_{t+1}\).
5. **Step 4** If \(\sigma_{t+1} \leq \varepsilon\), then stop. Otherwise set \(t = t + 1\) and loop to Step 1.

For further details about this procedure and the proof of its convergence, see [7].

4.3. **Bundle method**

Kelley cutting plane method is known to be unstable. Bundle methods [11,14, 18] aim at overcoming this instability by computing the Moreau-Yosida regularization of the piecewise linear approximation of (7). More precisely, the master problem to consider is obtained by adding a quadratic term to (8) as follows:

\[
\begin{align*}
\min_{x, Z} & \quad Z + \frac{\mu_t}{2} \|x - \hat{x}_t\|^2 \\
\text{s.t.} & \quad Z \geq \mathcal{P}_i(x_k) - v_{ki}^T(x - x_k), \quad i = 1, 2, \ldots, |S|, \quad k = 0, 1, \ldots, t, \\
& \quad c^T x \leq B, \\
& \quad x \geq 0.
\end{align*}
\]
where \( \hat{x}_t \) is the stability center: a point which is known to be relatively good; \( \mu_t \) is a positive parameter which controls the tradeoff between minimizing \( Z \) and staying close to \( \hat{x}_t \). The stability center is updated if \( x_{t+1} \) is significantly better than \( \hat{x}_t \) in the sense that

\[
Q(x_{t+1}) \leq Q(\hat{x}_t) - \kappa(Q(\hat{x}_t) - Z_{t+1}),
\]

(11)

where \((x_{t+1}, Z_{t+1})\) is the solution of (10), and \( \kappa \in [0, 0.5] \). If (11) holds, then we have a descent step and set \( \hat{x}_{t+1} = x_{t+1} \), otherwise we have a null step and the stability center is left as it is. The resulting cutting planes are added to (10) in both cases. The bundle algorithm (BA) stops when

\[
Q(\hat{x}_t) - Z_{t+1} \leq \varepsilon (1 + |Q(\hat{x}_t)|),
\]

(12)

where \( \varepsilon \) is the desired accuracy.

**Bundle Algorithm (BA)**

- **Step 0** Initialize \( x_0 = \hat{x}_0 = 0, \varepsilon > 0, \kappa \in [0, 0.5] \) and \( t = 0 \). Call the oracle at \( x_0 \) to obtain \( P_i(x_0) \) and \( v_0 \) for \( i = 1, \ldots, |S| \), and \( Q(x_0) \).

- **Step 1** Solve (10) and let \((x_{t+1}, Z_{t+1})\) be its optimal solution.

- **Step 2** Call the oracle at \( x_{t+1} \) to obtain \( P_i(x_{t+1}) \) and \( v_{t+1} \) for \( i = 1, \ldots, |S| \), and \( Q(x_{t+1}) \).

- **Step 3** If (11) is true then set \( \hat{x}_{t+1} = x_{t+1} \), otherwise set \( \hat{x}_{t+1} = \hat{x}_t \).

- **Step 4** Compute \( \mu_{t+1} \).

- **Step 5** If (12) is true then stop. Otherwise set \( t = t + 1 \) and loop to Step 1.

**Remark.** Step 3 is defined as weight updating and it is crucial for the speed of convergence of the algorithm. We use a procedure described in [14], where the weight \( \mu_t \) is decreased if the approximation of \( Q \) is close to \( Q \) at \( x_{t+1} \), and it is increased if the errors of the new linearizations are greater than a variation estimate of \( Q(\hat{x}_t) \). See [14] for further details.

5. **Results and numerical experiments**

We consider the above three methods for the following reasons. Kelley’s method is one of the most methods used in nonsmooth optimization while Elzinga-Moore algorithm has been, to the best of our knowledge, used rarely even if it is not much more hard to implement than the former. It outperformed Kelley’s method on some min-max problems arising in robust capacity planning problems, see [22]. Our first aim was to confirm this efficiency. We also consider the proximal bundle method because it is known as one of the best methods for nonsmooth optimization and we think it is extremely informative to compare it with the two above methods. Its subproblem is a quadratic problem which is considered to be more difficult than the linear subproblems of Kelley and Elzinga-Moore methods.

We have written an experimental code based on the above development, under Eclipse Platform 2.1.1; we have used Java 1.4.2 and ILOG Cplex 9.0 for the solution
of the subproblems (4), (8), (9) and (10). All the runs are performed on a two-
processor Xeon Intel server of 2.4 GHz CPU speed and 1.5 GB of RAM memory,
running under Linux.

The networks of our experiments derive from actual networks with given nominal
demands. Their sizes are given in Table 1, as well as the numbers of OD-pairs
and the number $|S|$ of scenarios considered in each case. Those scenarios have
been generated in two ways as follows:

1. From the available nominal vector of demands $\omega$ and a positive parameter
$\rho$, we have generated random $\epsilon^s_k \in [-\rho, \rho]$ for $s = 1, \ldots, |S|$ and $k = 1, \ldots, |K|$ and set

$$\omega^s_k = \epsilon^s_k \omega_k, \quad k = 1, 2, \ldots, |K|.$$

2. From the nominal vector of demands $\omega$ we have generated random $\epsilon^s_k \in [0, 1]$ for $s = 1, \ldots, |S|$ and $k = 1, \ldots, |K|$, and set

$$\omega^s_k = \epsilon^s_k \frac{\sum_j \omega_j}{\sum_j \epsilon_j^s}, \quad k = 1, 2, \ldots, |K|.$$

The first case results in scenarios that are variations around the nominal demand,
and we use it for graphs 1, 2, 3, 4 and 6. In the second case the sum of the
demands is the same for every scenario, and we use it for graphs 5 and 6. Finally,
the nominal demand is included in the set of scenarios.

In Table 2 we report the results obtained by the three algorithms presented
in Section 4. The column headed “Budget” specifies the parameter $B$ in each
test problem, see (1). This budget is derived arbitrarily, having as indicators
the solutions of the multicommodity flow problems, formed by every graph and
its corresponding scenarios. Different values for the budget are tested, but it is
always requested that finally at least one objective value equals to 0 (zero) and
various positive solutions for every graph are acquired. Then, for every method
we report the value of the objective function, the number of oracle calls and the
CPU time in seconds. Graph 6 appears twice because both scenario generation
methods 1 and 2 were used for it (the notations 6a and 6b correspond to the first
and second method respectively).
<table>
<thead>
<tr>
<th>Budget</th>
<th>BA</th>
<th>VA</th>
<th>KA</th>
</tr>
</thead>
<tbody>
<tr>
<td>#calls</td>
<td>CPU time</td>
<td>obj. value</td>
<td>CPU time</td>
</tr>
<tr>
<td>422</td>
<td>9657.57</td>
<td>0.00</td>
<td>330</td>
</tr>
<tr>
<td>330</td>
<td>9657.57</td>
<td>0.00</td>
<td>330</td>
</tr>
<tr>
<td>630</td>
<td>9657.57</td>
<td>0.00</td>
<td>330</td>
</tr>
<tr>
<td>930</td>
<td>9657.57</td>
<td>0.00</td>
<td>330</td>
</tr>
<tr>
<td>1230</td>
<td>9657.57</td>
<td>0.00</td>
<td>330</td>
</tr>
<tr>
<td>1530</td>
<td>9657.57</td>
<td>0.00</td>
<td>330</td>
</tr>
</tbody>
</table>

Table 2. Results obtained by the different algorithms.
It appears from these results that, in terms of number of calls to the oracle, the bundle algorithm outperforms the other two methods: it divides the number of Kelley’s [resp. Elzinga-Moore] oracle calls by a factor of 4.2 to 12.7 [resp. 2.6 to 11.1]. However, this reduction is not reflected in the CPU times: in particular for large problems, the bundle algorithm can become substantially slower than Elzinga-Moore. The reason is that the quadratic subproblem in our implementation becomes time consuming when the sizes of the problem and of the bundle increase. This behaviour contradicts the experiments of [4], which clearly show that quadratic programming is hardly more expensive than linear programming, and thus demonstrates the usefulness of specialized algorithms such as [8, 15]. On the other hand, Elzinga-Moore method behaves efficiently when compared to Kelley’s, as reported in [22]. We hope that the readers will be encouraged to test this method when only a linear solver is at their disposal. Figure 1 illustrates in another way the above observations, where we plot the number of oracle calls vs different budget for the first four graphs using other 20 traffic scenarios in each case.

Using the first four graphs, we conduct some numerical experiments with the proximal bundle algorithm, to analyse the model in terms of number of scenarios and budget. The number of scenarios to be considered in the model is itself a difficult issue which must be considered at a first stage before applying the model. Interesting studies about the demand modeling exist in the literature, (see for instance [3]) and can be used to simulate the demand and provide the scenarios.
Table 3. Results with different number of scenarios.

<table>
<thead>
<tr>
<th>Budget</th>
<th>obj. value</th>
<th>#oracle</th>
<th>CPU time</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 scenarios</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 scenarios</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 scenarios</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 scenarios</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The greater the number of scenarios is, the better the uncertainty set minimize the forecast error. However, some traffic scenario may dominates others. One of the main features of the proximal bundle method is its small number of oracle calls that do not depend on the number of scenarios in hand.

6. Conclusions

We have proposed a new approach for the network design problem in telecommunications under demand uncertainty. The uncertainty is modelled as a convex set of a moderate number of demand’s scenarios, which results in a convex non-smooth problem. For its solution, we considered and compared three algorithms: the proximal bundle method and the cutting plane algorithms by Kelley and Elzinga-Moore. The proximal bundle method appears to be the most efficient in terms of number of calls to the oracle, but is impeded by the use of a general-purpose quadratic solver. An extension of our proposed methodology to the general case of a polyhedral uncertainty set is currently under study.

References


To access this journal online:
www.edpsciences.org