Finding a dominant set of traffic demand matrices

S.E. Terblanche*, R. Wessäly†‡ and J.M. Hattingh*
*School of Computer, Statistical and Mathematical Sciences
North-West University (Potchefstroom)
Email: rkwset@puk.ac.za, rkwjmh@puk.ac.za
†Konrad-Zuse-Institute Berlin, Germany
‡atesio GmbH Berlin, Germany
Email: wessaely@{zib,atesio}.de

***Abstract*— Various approaches have been suggested for the robust design of communication networks. Among them is the design of a network such that a multitude of traffic demand matrices can be be satisfied non-simultaneously. This approach, however, tends to be computationally very time-consuming. To improve the running times we explore in this paper means to reduce the number of traffic demand matrices to a so-called dominant (sub-)set. That is, we try to identify those matrices which do not change the solution space of feasible network designs. To show the efficiency of this method, computational results are provided for traffic demand matrices taken from measurements in a live-network as well as some randomly generated data.**

I. Introduction

Traffic demand matrices play a vital role in the planning process of communication networks. Equipment is installed to satisfy capacity requirements which in turn is a function of a traffic demand matrix. A single traffic demand matrix may, however, be insufficient for representing varying traffic conditions in a network. For example, consider the noncoincident nature of business traffic vs. residential traffic. In the evenings a certain amount of traffic might be relocated from some part of the network to another as residential traffic increases and work related traffic decreases. One way of dealing with the problem is to have multiple traffic demand matrices where each matrix is a snapshot of the traffic requirements at a specific point in time. Not only may the matrices differ in volume, but also in load distribution. The source of multiple traffic demand matrices could be measurements from an existing network, forecast data, or simulation data generated according to some distribution. Network dimensioning with multiple demand matrices will consequently entail designing a minimal cost network with feasible routings that satisfy the traffic requirements for all of the multiple demand matrices (see e.g. [1], [2], [3], [4], [5]). From an algorithmic point of view the number of demand matrices that will be considered as part of the problem domain will contribute to the complexity of the problem. The network design problem with one traffic demand matrix is already considered to be a hard problem (see [6]). It is the objective of this paper to describe an algorithm that is able to identify traffic demand matrices which can be removed from the problem description without changing the cost of an optimal network design. The remaining sub-set of traffic demand matrices is referred to as a dominant set. The algorithm has been tested using traffic data measured from an operational network in Germany as well as data that has been generated randomly.

The remainder of this paper is subdivided into the following sections: in Section II an overview of the notation used in the paper is provided. In Section III the concept of domination is introduced with specific references to the work done in [7]. In this section the theory on domination is extended to multiple traffic demand matrices and an algorithmic framework is presented to do dominance checking for both dynamic and static routing models. In Section IV implementation details are provided and computational result are presented in Section V. Finally, a summary and conclusion is given in Section VI.

II. Notation and Model Assumptions

Let $G = (V, E)$ be a complete graph with undirected edges. The set $\delta(e) = \{i, j\}$ with $e \in E$ is used to represent the end nodes $i, j \in V$ associated with an edge $e \in E$. For ease of notation we use *demand vectors* to represent single traffic demand matrices. Each component of a demand vector, d_e with $e \in E$, represents a demand requirement for the communicating node pair $\delta(e)$. If no demand requirements exist for a node pair $\delta(e)$, then $d_e = 0$. The set $\mathcal{T} = \{1, 2, \ldots, T\}$ is used to index the set of demand vectors $D = \{d^1, d^2, \ldots, d^T\}$.

Let $p \in \mathcal{P}$ be a set of edges that define a non-cyclic undirected path between a communicating node pair. The set $\mathcal{P}$ contains all possible paths for all possible node pairs and the subset $\mathcal{P}(e) \subseteq \mathcal{P}$ contains all paths that can route traffic between a node pair $\delta(e)$. For each edge $l \in E$ the subsets $\mathcal{P}(e, l) \subseteq \mathcal{P}(e)$ contain all paths for a communicating node pair $\delta(e)$ that traverse the edge l. The variables $r_p \in \mathbb{R}_+$ are introduced to define the fraction of flow for traffic on path $p \in \mathcal{P}$. For a given capacity vector $y \in \mathbb{R}_+^{|E|}$ and a single demand vector $d \in \mathbb{R}_+^{|E|}$ the following polyhedron $P(y, d)$ defines all feasible routings of the associated multi-commodity network flow problem:

$$P(y,d) = \Bigg\{ r \in \mathbb{R}_+^{|\mathcal{P}|} :$$

$$\sum_{p \in \mathcal{P}(e)} r_p = 1, \ \forall \ e \in E \quad (1)$$

$$\sum_{e \in E} \sum_{p \in \mathcal{P}(e,l)} r_p d_e \leq y_l, \ \forall \ l \in E \Bigg\} \quad (2)$$

For convenience the vector $r \in P(y,d)$ is referred to as a feasible routing.

If $P(y,d) \neq \emptyset$ we say the capacity vector y *supports* the demand vector d. The set $\mathcal{Y}(d)$ denotes the set of all capacity vectors supporting a demand vector d. Furthermore, let the set $\mathcal{YR}(d)$ be the set of all pairs y and r, where y supports d with a feasible routing $r \in P(y,d)$.

In order to extend the multi-commodity network flow problem for a set of demand vectors $D = \{d^1, d^2, \ldots, d^T\}$, the variables $r_p^t \in \mathbb{R}_+$ are introduced to define the fraction of flow for traffic on path $p \in \mathcal{P}(e)$, for a demand vector d^t, $t \in \mathcal{T}$. The associated polyhedron is the following:

$$P(y,D) = \Bigg\{ r \in \mathbb{R}_+^{|\mathcal{P}| \times T} :$$

$$\sum_{p \in \mathcal{P}(e)} r_p^t = 1, \ \forall \ e \in E, \ \forall \ t \in \mathcal{T}$$

$$\sum_{e \in E} \sum_{p \in \mathcal{P}(e,l)} r_p^t d_e \leq y_l, \ \forall \ l \in E, \ \forall \ t \in \mathcal{T} \Bigg\}$$

If $P(y,D) \neq \emptyset$ a set of feasible routings exist such that the demand vectors $d^1, d^2, \ldots, d^T$ can be routed non-simultaneously. In this case the capacity vector y *supports* the set of demand vectors D. The set $\mathcal{Y}(D)$ denotes the set of all capacity vectors supporting the set of demand vectors D.

The preceding definition of $P(y,D)$ implies a *dynamic routing* model since the demand vectors $d^1, d^2, \ldots, d^T$ can be routed independantly of each other according to the routings $r^1, r^2, \ldots, r^T$. In some cases, due to different planning requirements, *static routing* might be preferred where the same routing is applied to all demand vectors. To enforce static routing for the multi-commodity network flow problem, the index t can be discarded in the definition of $P(y,D)$ resulting in only one set of routing variables defining the same routing for all of the demand vectors.

III. DOMINATION

The decision to include or exclude a demand vector in the computation of a network design requires the notion of dominance. That is, if every capacity vector y that supports a demand vector d^1 also allows a feasible routing of demand vector d^2, then d^1 *dominates* d^2.

Definition 1: A demand vector d^1 dominates d^2 iff $\mathcal{Y}(d^1) \subseteq \mathcal{Y}(d^2)$ [7].

There is, however, a distinction to be made between dynamic and static routing. The preceding definition of domination imposes no restriction on routing and, therefore, implies that even if the same capacity vector y can support both d^1 and d^2, the two demand vectors can be routed independantly.

In order to extend the notion of domination for the case of static routing we say that if every capacity vector y that supports a demand vector d^1 also supports d^2, and if in addition $r^1 = r^2$, then d^1 *totally dominates* d^2.

Definition 2: A demand vector d^1 totally dominates d^2 iff $\mathcal{YR}(d^1) \subseteq \mathcal{YR}(d^2)$ [7].

In the subsequent sections the basic theory of domination w.r.t. dynamic and static routing is extended to cater for multiple demand vectors.

A. Dominance Checking for Dynamic Routing

Theorem 1: A demand vector d^1 dominates d^2 iff $P(d^1, d^2) \neq \emptyset$. [7].

Proposition 1: Let $y \in \mathbb{R}_+^{|E|}$ be a capacity vector that supports both the demand vectors d^1 and d^2. Then y supports the convex combination $d^c = \lambda d^1 + (1-\lambda) d^2$, with $0 \leq \lambda \leq 1$.

Proof: For each edge $e \in E$ let $r_p^c = f_p^c / d_e^c$ with $f_p^c = \lambda r_p^1 d_e^1 + (1-\lambda) r_p^2 d_e^2$, $p \in \mathcal{P}(e)$, then

$$\begin{aligned}
\sum_{p \in \mathcal{P}(e)} r_p^c &= \sum_{p \in \mathcal{P}(e)} \frac{f_p^c}{d_e^c} \\
&= \sum_{p \in \mathcal{P}(e)} \left(\frac{\lambda r_p^1 d_e^1 + (1-\lambda) r_p^2 d_e^2}{d_e^c} \right) \\
&= \frac{\lambda d_e^1 \sum_{p \in \mathcal{P}(e)} r_p^1 + (1-\lambda) d_e^2 \sum_{p \in \mathcal{P}(e)} r_p^2}{d_e^c} \\
&= \frac{\lambda d_e^1 + (1-\lambda) d_e^2}{d_e^c} \\
&= 1
\end{aligned}$$

This satisfies (1) of $P(y, d^c)$. Furthermore, for each edge $l \in E$

$$\begin{aligned}
\sum_{e \in E} \sum_{p \in \mathcal{P}(e,l)} r_p^c d_e^c &= \sum_{e \in E} \sum_{p \in \mathcal{P}(e,l)} f_p^c \\
&= \sum_{e \in E} \sum_{p \in \mathcal{P}(e,l)} \left(\lambda r_p^1 d_e^1 + (1-\lambda) r_p^2 d_e^2 \right) \\
&= \lambda \sum_{e \in E} \sum_{p \in \mathcal{P}(e,l)} r_p^1 d_e^1 \\
&\quad + (1-\lambda) \sum_{e \in E} \sum_{p \in \mathcal{P}(e,l)} r_p^2 d_e^2 \\
&\leq \lambda y_l + (1-\lambda) y_l \\
&= y_l
\end{aligned}$$

This completes the proof since (2) of $P(y, d^c)$ is also satisfied. ■

Proposition 1 can be generalized such that the result is true for more than two demand vectors.

Definition 3: A set of demand vectors D dominates a demand vector d iff $\mathcal{Y}(D) \subseteq \mathcal{Y}(d)$.

Corollary 1: Let $d^c = \sum_{t\in\mathcal{T}} \lambda^t d^t$ with $\sum_{t\in\mathcal{T}} \lambda^t = 1$, $\lambda^t \geq 0$, dominate the demand vector d. Then the set $D = \{d^t, \forall t \in \mathcal{T}\}$ dominates d.

Proof: According to Proposition 1 each capacity vector y that supports D will also support the convex combination d^c. Furthermore, if d^c dominates d then $\mathcal{Y}(d^c) \subseteq \mathcal{Y}(d)$, therefore $\mathcal{Y}(D) \subseteq \mathcal{Y}(d^c) \subseteq \mathcal{Y}(d)$. ■

The implication of Corollary 1 is that if we can find a demand vector $d^c \in C$ that dominates d, where $C = conv(D)$ is the convex hull of demand vectors D, then d can be discarded in the network design process. The following linear program, called the Dynamic Routing Dominance Checker (DRDC), is employed to assist in finding such a d^c.

$$\text{(DRDC)}: \quad \min \ \alpha$$

$$\sum_{p\in\mathcal{P}(e)} r_p = 1 \quad \forall\ e \in E \tag{3}$$

$$\sum_{e\in E}\sum_{p\in\mathcal{P}(e,l)} r_p d_e - \alpha - d^c_l \leq 0 \quad \forall\ l \in E \tag{4}$$

$$d^c_e - \sum_{t\in\mathcal{T}} \lambda^t d^t_e = 0 \quad \forall\ e \in E \tag{5}$$

$$\sum_{t\in\mathcal{T}} \lambda^t = 1 \tag{6}$$

$$\lambda^t \geq 0 \quad \forall\ t \in \mathcal{T}$$

$$r_p \geq 0 \quad \forall\ e \in E, \quad \forall\ p \in \mathcal{P}(e)$$

The constraint sets (5) and (6) are responsible for expressing d^c as a point in C.

If the optimum yields $\alpha \leq 0$, it can be concluded that a point in C could be found that provides enough “capacity” that allows a feasible routing of d. Therefore, an optimum point given by the solution d^c will dominate d. If on the other hand, $\alpha > 0$, it is determined that no point in C exists that dominates d.

B. Domination Checking for Static Routing

The domination approach outlined in the previous section is only applicable to dynamic routing since the routings provided by solving the problem DRDC for each of the demand vectors in D could differ from each other.

The approach for doing dominance checking w.r.t. static routing relies on the definition of total dominance (already stated in the introductory part of Section III).

Theorem 2: A demand vector d^1 totally dominates a demand vector d^2 iff $d^1_e \geq d^2_e$ for all $e \in E$ [7].

Proposition 2: Let $y \in \mathbb{R}^{|E|}_+$ be a capacity vector that supports both the demand vectors d^1 and d^2 with routing r. Then y supports the convex combination $d^c = \lambda d^1 + (1-\lambda)d^2$, $0 \leq \lambda \leq 1$ with routing r.

Proof: Let $d^c = \lambda d^1 + (1-\lambda)d^2$ with $0 \leq \lambda \leq 1$, then for each edge $l \in E$

$$\begin{aligned}
\sum_{e\in E}\sum_{p\in\mathcal{P}(e,l)} r_p d^c_e &= \sum_{e\in E}\sum_{p\in\mathcal{P}(e,l)} \left(\lambda r_p d^1_e + (1-\lambda) r_p d^2_e\right) \\
&= \lambda \sum_{e\in E}\sum_{p\in\mathcal{P}(e,l)} r_p d^1_e \\
&\quad + (1-\lambda) \sum_{e\in E}\sum_{p\in\mathcal{P}(e,l)} r_p d^2_e \\
&\leq \lambda y_l + (1-\lambda) y_l \\
&= y_l
\end{aligned}$$

This completes the proof. ■

Proposition 2 can be generalized such that the result is true for more than two demand vectors.

Definition 4: A set of demand vectors D totally dominates a demand vector d iff $\cap_{t\in\mathcal{T}} \mathcal{YR}(d^t) \subseteq \mathcal{YR}(d)$.

Corollary 2: Let $d^c = \sum_{t\in\mathcal{T}} \lambda^t d^t$ with $\sum_{t\in\mathcal{T}} \lambda^t = 1$, $\lambda^t \geq 0$ totally dominate the demand vector d. Then the set $D = \{d^t, \forall t \in \mathcal{T}\}$ totally dominates d.

Proof: According to Proposition 2 each capacity vector y that supports $D = \{d^t, \forall t \in \mathcal{T}\}$ with a routing r will also support the convex combination d^c with routing r. Furthermore, if d^c totally dominates d then $\mathcal{YR}(d^c) \subseteq \mathcal{YR}(d)$, therefore $\cap_{t\in\mathcal{T}} \mathcal{YR}(d^t) \subseteq Y(d^c) \subseteq \mathcal{YR}(d)$. ■

The implication of Corollary 2 is that if we can find values for $\lambda^t \geq 0$, for all $t \in \mathcal{T}$ such that

$$d_e \leq \sum_{t\in\mathcal{T}} \lambda^t d^t_e \ \forall\ e \in E$$

with $\sum_{t\in\mathcal{T}} \lambda^t = 1$, then d can be discarded in the network design process since the set D totally dominates d. The following linear program, called the Static Routing Dominance Checker (SRDC), solves the problem of finding values for $\lambda^t \geq 0$ by introducing a slack variable $\alpha \in \mathbb{R}$ that needs to be minimized:

$$\begin{aligned}
\min \ \alpha & \\
\sum_{t\in\mathcal{T}} \lambda^t d^t_e + \alpha &\geq d_e \quad \forall\ e \in E \\
\sum_{t\in\mathcal{T}} \lambda^t &= 1 \\
\lambda^t &\geq 0 \quad \forall\ t \in \mathcal{T}
\end{aligned}$$

The objective function is to minimize the error found in writing d as a convex combination of the demand vectors D. If the optimum to the linear program yields $\alpha \leq 0$, then a point was found in C that totally dominates d, and therefore d can be discarded. If $\alpha > 0$ then no dominating point in C exists and d will have to be included in the network design process.

IV. Implementation Details

For purposes of applying our theory on dominance, we implemented code to do dominance checking in the case of both dynamic and static routing. The implementation is based on the network design tool DISCNET (see [8] and [9]). The code has been implemented using C++ and the commercial mathematical programming software CPLEX is used for solving the linear programming problems.

The implementation of the dominance checker that assumes dynamic routing is more complex compared to the static routing case since a column generation approach is followed in order to manage the exponential number of path variables in the linear program.

Let the dual variables $\pi \in \mathbb{R}^{|E|}$, $\mu \in \mathbb{R}_+^{|E|}$, $v \in \mathbb{R}^{|E|}$, and $w \in \mathbb{R}$ be associated with the constraints (3), (4), (5) and (6) in the formulation of DRDC, respectively. Furthermore, let $E(p)$ be the set of edges being traversed by a path $p \in \mathcal{P}$, then the dual of the DRDC problem is the following:

$$\max w + \sum_{e \in E} \pi_e$$

$$\sum_{e \in E} \mu_e = 1 \tag{7}$$

$$\mu_e + v_e = 0 \quad \forall\, e \in E \tag{8}$$

$$w - \sum_{e \in E} d_e^t v_e \leq 0 \quad \forall\, t \in \mathcal{T}, \tag{9}$$

$$\pi_e - \sum_{l \in E(p)} d_e \mu_l \leq 0 \quad \forall\, e \in E, \quad \forall\, p \in \mathcal{P}(e) \tag{10}$$

The constraint set (10) of the dual problem states that each variable π_e, for a communicating node pair $\delta(e)$, is less or equal to the minimum of the expression $\sum_{l \in E(p)} d_e \mu_l$ for all paths $p \in \mathcal{P}$. This is equivalent in saying that each π_e is the shortest path length for a communicating node pair $\delta(e)$ calculated with edge weights $d_e \mu_l$, for all edges on the shortest path. It is, therefore, not necessary to include all possible paths in the formulation. The pricing operation required is simply to calculate shortest paths using the values for $d_e \mu_l$ for each of the communicating node pairs. Whenever it is found that a shortest path has been calculated that is less than any of the π_e's, this new shortest path is added to the formulation.

The overall procedure for doing dominance checking is the same for both the static and the dynamic case. Let DOM_CHECK(d) denote the sub-procedure that solves either the DRDC or the SRDC problem for a demand vector d depending on whether dynamic or static routing is considered. The optimal objective value obtained by calling the sub-procedure is z^* and the vector λ^* is the optimal solution for the variables λ^t, for all $t \in \mathcal{T}$ which is present in both the problems DRDC and SRDC. If it is found that a demand vector d is being dominated, it is added to the set $\mathcal{A}$. Initially this set is empty and will eventually contain the set of dominated demand vectors. The main algorithm that calls the sub-procedure DOM_CHECK(d) is listed in Algorithm 1.

Algorithm 1 Dominance Checking Algorithm

```
A = ∅
for all t ∈ T do
  for all k ∈ K do
    Set d_k = d_k^t
  end for
  Set variable bounds λ^t = 0
  Call sub-procedure DOM_CHECK( d )
  if z* < 0 then
    Add d^t to the set A.
  else
    Set variable bounds λ^t ≥ 0
  end if
end for
```

V. Computational results

The empirical results of this paper are based on 2 types of problem instances. For the first type of problem instances data were collected through measurements from an operational network called G-WIN (German Research Network). For the second type of problem instances data were generated randomly. For both types of problem instances a demand graph consisting out of 32 nodes were considered.

A. G-WIN data

Two sets of data where collected through measurements. The first data set consisting out of 12 demand matrices was measured over a time period of a year, one demand matrix per month. The second data set consisting out of 28 demand matrices, one demand matrix per day, was measured in the month of February. The total number of directed commodities that was recorded for the 32 node demand graph is $996 = (\frac{32 \times 31}{2}) \times 2$. Some preprocessing was done on the raw data that resulted in a reduced number of commodities. A minimum cut-off value of 1Mbps was applied and demands in opposite directions for the same communicating node pairs were aggregated to comply with an undirected commodity model. The result was that the number of commodities considered for the monthly demand matrices was reduced to 236 and the number of commodities considered for the daily matrices was reduced to 328.

The results obtained by doing dominance checking for the G-WIN data are presented in Table I. The column labeled *Random* indicates whether the demand matrices where ordered descendingly prior to doing dominance checking, or selected randomly. The ordering criteria applied is simply the sum over all demands in the demand matrix. The columns labeled *Percent Dominated* and *Execute Time(s)* gives the number of demand matrices that were found to be dominated and the execution time of the algorithm respectively. These two columns exist for both the dynamic and static routing cases.

In the case of static routing no demand matrices were found that are being dominated in either of the 2 data sets. Consequently, all demand matrices will have to be considered if a network is designed that assumes static routing.

Data Set	Ordering	Dynamic Routing Percent. Dominated	Dynamic Routing Execute. Time(s)	Static Routing Percent. Dominated	Static Routing Execute. Time(s)
daily(28)	Random	25.0	161	0	0
	Descend	25.0	162	0	0
monthly(12)	Random	16.7	64	0	0
	Descend	16.7	44	0	0

TABLE I

COMPUTATIONAL RESULTS - G-WIN DATA

Scenarios	Ordering	Dynamic Routing Percent. Dominated	Dynamic Routing Execute. Time(s)	Static Routing Percent. Dominated	Static Routing Execute. Time(s)
100	Random	84.0	258	82.0	0
	Descend	84.0	176	82.0	0
200	Random	90.0	282	84.0	1
	Descend	90.0	184	84.0	1
400	Random	90.3	373	83.8	4
	Descend	90.3	243	83.8	3
800	Random	89.8	2077	84.8	14
	Descend	89.8	1105	84.8	11
1600	Random	90.1	5524	88.4	49
	Descend	90.1	3226	88.4	49

TABLE II

COMPUTATIONAL RESULTS - RANDOM DATA

B. *Random Data*

The random data are based on a worst case demand vector w that was created from the 12 monthly matrices measured from G-WIN. This was done by taking only the maximum demand values for each component over the 12 demand matrices. Demand vectors were then created randomly by using the following:

$$d^i_k = \rho^i \gamma^k w_k$$

where d^i_k is the k-th demand for a demand vector d^i generated for a scenario i. The parameter ρ^i is for introducing variation across all the scenarios and the parameter γ^k is to randomize demands within a demand vector. The two parameters are sampled out of the uniform distributions $U(0.5, 1.5)$ and $U(0.2, 1, 2)$ respectively. The ranges for the two distributions were selected arbitrarily.

The results obtained by doing dominance checking with randomly generated data are presented in Table II. The number of scenarios that we considered was 100, 200, 400, 800, and 1600.

From the results an important observation is that in all of the scenarios considered for the dynamic routing case, the ordering of demand matrices had a significant influence on the performance of the implementation. For the problem instance considering 800 scenarios a decrease of up to 53 % in execution time was observed.

Furthermore, the results show that for all of the problem instances the percentage of demand matrices found to be dominated in the case of dynamic routing exceeds the percentage of demand matrices found to be dominated in the case of static routing. This is expected since for static routing additional constraints on the routing are being imposed.

VI. SUMMARY AND CONCLUSION

In this paper we gave an overview of dominance theory in the context of multiple traffic demand matrices. An algorithm for doing dominance checking for both the dynamic and static routing cases were presented. The computational results are positive considering we could do dominance checking in the case of dynamic routing for a scenario of 1600 demand matrices in just $1\frac{1}{2}$ hours on a regular sized desktop computer. In the case of static routing, the algorithm terminated in just 49 seconds for the same data set. This is expected since the problem size for doing dominance checking in the case of static routing is moderate and does not involve an approach that requires column or row generation.

A combined approach could however be followed for doing dominance checking in case of dynamic routing by considering the fact that total dominance is a special case of domination. That is, a first stage is to reduce the number of demand matrices using the computationally less expensive SRDC algorithm (i.e. dominance checking with static routing). The final stage is then to apply the DRDC algorithm on the remaining demand matrices to check for dominance in the case of dynamic routing.

The percentage reduction in the number of traffic demand matrices based on domination is significant for several problem instances that we considered. The best results obtained for the real life data is a 25% reduction for the problem involving demand matrices measured on daily intervals (28 of them for the month of February). For the data generated randomly, a reduction of up to 90.3% was recorded.

REFERENCES

[1] A. Dutta, "Capacity planning of private networks using DCS under multibusy-hour traffic," *IEEE Transactions on Communications*, vol. 42, no. 7, pp. 2371–2374, 1994.

[2] D. Medhi and D. Tipper, "Some approaches to solving a multi-hour broadband network capacity design problem with single-path routing," *Telecommunication Systems*, vol. 13, no. 2-4, pp. 269–291, 2000.

[3] A. Amiri and H. Pirkul, "Routing and capacity assignment in backbone communication networks under time varying traffic conditions," *European Journal of Operational Research*, vol. 117, pp. 15–29, 1999.

[4] M. Riis and K.A. Andersen, "Capacitated network design with uncertain demand," *INFORMS Journal on Computing*, vol. 14, no. 3, pp. 247–260, 2002.

[5] S.E. Terblanche, R. Wessäly, and J.M. Hattingh, "Solutions for multi-hour survivable network design problems," in *Proceedings of the South African Telecommuniations and Networks Applications Conference - SATNAC 2005*.

[6] "SNDlib 1.0 – Survivable network design data library," http://sndlib.zib.de, 2005.

[7] G. Oriolo, "Domination between traffic matrices," http://www.optimization-online.org, 2004.

[8] R. Wessäly, *DImensioning Survivable Capacitated NETworks*, Phd thesis, Technische Universität Berlin, 2000.

[9] DISCNET, www.atesio.de, atesio GmbH, Berlin, Germany.