Hybrid Demand Oblivious Routing: Hyper-cubic Partitions and Theoretical Upper Bounds

Gábor Németh and Gábor Rétvári[*]

Dept. of Telecommunications and Media Informatics
Budapest University of Technology and Economics
Magyar tudósok körútja 2., Budapest, Hungary, H-1117
`{nemethgab,retvari}@tmit.bme.hu`

Summary. Traditionally, network routing was optimized with respect to an expected traffic matrix, which left the network in a suboptimal state if user traffic did not match expectations. A demand-oblivious routing is, contrarily, optimized with respect to all possible traffic matrices, obviating the need for traffic matrix estimation. Oblivious routing is a fundamentally distributed scheme, so it can be implemented easily. Unfortunately, in certain cases it may cause unwanted link over-utilization. Recently, we have introduced a hybrid centralized-distributed method to mitigate this shortcoming. However, our scheme did not provide a theoretical upper bound for the link over-utilization. In this paper, we tackle the problem again from a different perspective. Based on a novel hyper-cubic partition of the demand space, we construct a new algorithm that readily delivers the theoretical bounds. Simulation results show the theoretical and practical significance of our algorithm.

Keywords: oblivious ratio, demand-oblivious routing, hyper-cubic region.

1 Introduction

Traffic Engineering (TE) [1] has become the key tool used in the majority of autonomous systems, whose task is to map user traffic to the physical network efficiently. This is important given the high cost of the elemental network infrastructure and the highly competitive nature of the ISP market. Most of the TE methods are offline methods: forwarding paths are optimized with respect to some measured and/or expected traffic matrices over some period of time, and over-provisioning of network capacity ensures that unpredictable traffic spikes do not cause violation of link capacities [2, 3]. In a more dynamically changing environment, this routing strategy has become more and more inadequate. Thus, several proposals have surfaced to reduce the significance of traffic matrices (e.g., [4–7]) or even eliminate them (e.g., [8–13]) in intra-domain traffic engineering.

[*] The second author was supported by the Janos Bolyai Fellowship of the Hungarian Academy of Sciences.

A practical method to deal with unpredictable traffic matrices is called *(demand) oblivious routing* [14–22]. Here, the basic idea is to handle *all* legitimate traffic matrices simultaneously. Demand-oblivious routing is a fundamentally distributed scheme, meaning that the amount of traffic sent to a forwarding path by a router only depends on information available locally at that router. This ensures simplicity and scalability. Unfortunately, not all traffic matrices can be routed equally efficiently in oblivious routing, therefore, the routing might underperform for the everyday traffic scenario and consequently the network will operate in a suboptimal state in the majority of the time. Moreover, there is no theoretical upper bound on the capacity oversubscription, and hence, congestion demand-oblivious routing might cause [16].

Based on this insight, hybrid methods, combining the advantages of oblivious routing with some minimal central knowledge, have drawn the attention of the research community lately. The first hybrid algorithm was introduced in [23]. The main idea is to split the set of all legitimate traffic matrices (the so-called throughput polytope [24]) into multiple hyper-cubic regions, and assign a separate routing function to all of these regions. The individual routing functions are distributed, because the amount of traffic sent to a path at a source node only depends on the actual demand at that node, and it is independent of the demands at other nodes. A central controller, meanwhile, periodically observes the traffic matrix, decides in which operating region the network is, and installs the corresponding traffic splitting rations at the routers. This is exactly why hyper-cubic regions are a key concept in the algorithm: a hyper-cubic region provides the easiest way to decide whether the actual throughput is part of it. Thus, this architecture only needs a limited amount of central control. Although it has been shown that using the algorithm in [23] a significant improvement in the link over-subscription can be achieved by only a few cuts, it is still unclear whether the maximal link over-utilization converges to a minimum value.

In this paper, we analyze the properties of the maximal link over-utilization over hyper-cubic regions. In particular, we answer the questions

- how to create a finite hyper-cubic partition of the throughput region and
- what is the maximum size of each hyper-cubic region when fixing the maximal link over-utilization at a previously given value.

The rest of the paper is organized as follows. In Section 2, notations and definitions are introduced. In Section III, we introduce the mathematical backgrounds of our new algorithm. Simulation results are discussed in Section 4, related work is assessed in Section 5 and finally, Section 6 concludes the paper.

2 Notations and Definitions

Before introducing our main theorem, we need to summarize the main ideas of the geometric framework described in [23]. We need some basic terms and definitions from convex geometry also, thus we refer the reader to the introductory material in [25] and [26].

Suppose that we are given the network topology as usual, in the form of a directed graph $G(V, E)$ and a vector of positive, finite link capacities $c = [c_e > 0 : e \in E]$ (see Table 1 for a summary on notations). Each user is associated with a unique source-destination pair $(s_k, d_k) : k \in \mathcal{K}$ and a set of static paths $\mathcal{P}_k$. Additionally, each user k independently presents its demand θ_k at the source node s_k. The task of the routing algorithm is to distribute the demands θ_k between the paths $\mathcal{P}_k$, $\forall k \in \mathcal{K}$ in a way as to avoid or minimize link oversubscription.

Table 1. Notations

$G(V, E)$	a directed graph, with the set of nodes V ($\lvert V\rvert = n$) and the set of directed edges E ($\lvert E\rvert = m$)
c	the column m-vector of edge capacities
ξ_e	the number of paths sharing the edge $e \in E$
(s_k, d_k)	the set of source-destination pairs (or users) for $k \in \mathcal{K} = \{1, \dots, K\}$
$\mathcal{P}_k$	the set of $s_k \rightarrow d_k$ paths assigned to some $k \in \mathcal{K}$
p_k	the number of paths for user k, $p_k = \lvert\mathcal{P}_k\rvert$
p	number of all paths, $p = \sum_{k \in \mathcal{K}} p_k$
P_k	an $m \times p_k$ matrix. The column corresponding to path $P \in \mathcal{P}_k$ holds the path-arc incidence vector of P
u_P	scalar, describing the traffic routed at path P
u_k	a column-vector, whose components are $u_P : P \in \mathcal{P}_k$ for some $k \in \mathcal{K}$ (whether we mean u_k or u_p will always be clear from the context)
u	a column vector representing a particular choice of u_Ps (a "routing")
θ_k	the demand/throughput of some user $k \in \mathcal{K}$
θ	a column K-vector representing a particular combination of throughputs (a "traffic matrix")
M	flow polytope, the set of path flows corresponding to $\mathcal{P}$ subject to non-negativity and capacity constraints
T	throughput polytope, i.e., the set of throughputs realizable over $\mathcal{P}$ subject to capacity constraints
$\mathcal{S}, \mathcal{S}_k$	a routing function, $\mathcal{S} : \mathbb{R}^K \mapsto \mathbb{R}^p$ and the routing function corresponding to $k \in \mathcal{K}$, $\mathcal{S}_k : \mathbb{R}^K \mapsto \mathbb{R}^{p_k}$, respectively
$\mathcal{S}_R, \mathcal{S}_b$	the optimal routing function over the region $R \in \mathbb{R}^K$, or for the throughput $b \in \mathbb{R}^K$

A routing is, consequently, represented by a vector of path flows: $u = [u_k : k \in \mathcal{K}] \in \mathbb{R}^{p_1} \times \mathbb{R}^{p_2} \times \ldots \times \mathbb{R}^{p_K} = \mathbb{R}^p$, where p_k is the number of paths for k and p is the number of all paths.

Definition 1. *The polytope $M = \{u : \sum_{k \in K} P_k u_k \leq c,\ u \geq 0\} \subset \mathbb{R}^p$ is called* flow polytope*. M contains all admissible routings, subject to link capacities and non-negativity constraints.*

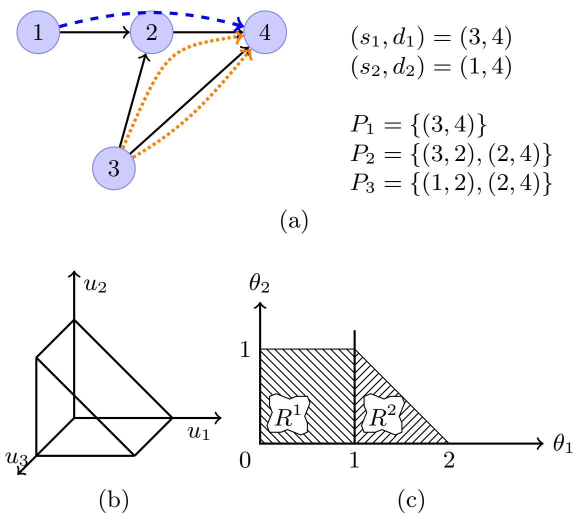

Fig. 1. The *(a)* sample network with unit edge capacities, source-destination pairs, the set of paths for each user and the corresponding *(b)* flow and *(c)* throughput polytopes

Consider the affine transformation (the so called *throughput mapping*) $\mathcal{T} : \mathbb{R}^p \to \mathbb{R}^K$ that from a routing u generates the corresponding traffic matrix θ by summing up the path flows for each particular path of a user:

$$\theta = \mathcal{T}(u) = [\theta_k = \sum_{P \in \mathcal{P}_k} u_P : k \in \mathcal{K}] \ .$$

Definition 2. *Mapping the flow polytope M through $\mathcal{T}$ gives the* throughput polytope *T [24]:*

$$\mathbb{R}^p \supset M \xrightarrow{\mathcal{T}} T \subset \mathbb{R}^K \ .$$

The throughput polytope contains all the traffic matrices realizable in the network by some properly chosen static routing without violating link capacities:

$$T = \{\theta : \exists u \in M \textit{ so that } \mathcal{T}(u) = \theta\} \ .$$

Consequently, we call $\theta \in T$ admissible.

A sample network and the corresponding polytopes are depicted in Fig. 1.

The other central object in the framework is the *routing function* $\mathcal{S}$, which determines the way a traffic matrix θ is mapped to the paths: $u = \mathcal{S}(\theta)$. The routing function $\mathcal{S}$ must always satisfy the *throughput invariance* rules, i.e., $\mathcal{S}(\theta)$ must realize precisely θ: $\forall \theta \in \mathbb{R}^K : \mathcal{T}(\mathcal{S}(\theta)) \equiv \theta$. Throughout this paper, we only consider affine routing functions of the form $\mathcal{S} : \mathbb{R}^K \to \mathbb{R}^p; \ \theta \mapsto F\theta + g$ (component-wise we have $\mathcal{S}_k : \theta \mapsto F_k\theta + g_k$), where F (F_k) is a $p \times K$ ($p_k \times K$) matrix and g (g_k) is a column vector of size p (p_k). Throughput invariance implies:

$$1^T F_{kl} = \delta_{kl} = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{otherwise} \end{cases}, \qquad 1^T g_k = 0 \ ,$$

where F_{kl} denotes the lth column of F_k. We call a routing function *distributed*, if

$$\frac{\partial \mathcal{S}_k}{\partial \theta_l} = 0 \text{ if } k \neq l$$

wherever the derivative is defined. The main advantage of distributed routing functions is that the amount of traffic sent to a path at a source node only depends on the actual demand at that node, and it is independent of the demands at other nodes.

Note that distributed routing functions can be treated as affine routing functions by restricting F to block-diagonal matrices. Another restriction can be made by fixing g at zero. In the latter case the routing function is a simple block diagonal linear transformation, i.e., it only specifies the splitting ratios based on which demands are distributed among the paths with the same ingress/egress nodes. Letting g be different from zero allows more freedom in assigning path-flows, though, this option might raise implementation issues as traffic splitting ratios need to vary in small steps in this case.

So far we have dealt with *simple routing functions*, in the sense that we tried to route all the admissible traffic matrices, that is, cover the whole throughput polytope, with one routing function. It is tempting to combine several routing functions into a single one. Such a *compound routing function* would be able to accommodate any admissible traffic matrix $\theta \in T$ with causing less link overload. To do this, we associate different routing functions with different regions of the throughput space, so $\mathcal{S}$ takes the form $\mathcal{S} = \{(R^i, \mathcal{S}^i) : i \in \mathcal{I}\}$ where R^is give a disjunct partition of the throughput space (e.g., see Fig. 1c, where two such a regions are depicted), and we use the mapping $\theta \mapsto \mathcal{S}^i(\theta)$ whenever $\theta \in R^i$.

Now, we are in a position to formulate the demand-oblivious routing problem in geometric terms.

Definition 3. *The minimal scalar α solving the optimization problem*

$$\min_{\mathcal{S}} \alpha : \mathcal{S}(T) \subseteq \alpha M \tag{1}$$

is called the oblivious ratio *and the corresponding routing function is called* demand-oblivious routing.

The interpretation of the optimization problem (1) is as follows. The set $\mathcal{S}(T)$ represents all the routings one can get when applying $\mathcal{S}$ to the set of admissible traffic matrices T. The objective is to up-scale the capacities of the network, and so the flow polytope of the network, to make all the routings in the set $\mathcal{S}(T)$ feasible. Hence, α signifies the maximal link over-utilization caused when routing *any* admissible traffic matrix over $\mathcal{S}$. This also implies that $\alpha \geq 1$.

So far, the oblivious ratio was defined with respect to the throughput set T. We need to find the oblivious ratio with respect to arbitrary regions and sometimes to arbitrary routing function. Thus, we need to introduce the following generalization.

Definition 4. *Given an arbitrary set of traffic matrices $\mathbb{R}^K \supset X \neq \emptyset$ and a routing function $\mathcal{S}$, the oblivious ratio $\alpha(X, S)$ with respect to X and $\mathcal{S}$ is the optimal solution of the optimization problem:*

$$\alpha(X, S) = \underset{\alpha}{\operatorname{argmin}}\{\mathcal{S}(X) \subseteq \alpha M\} . \tag{2}$$

Definition 5. *Given an arbitrary set of traffic matrices $\mathbb{R}^K \supset X \neq \emptyset$, the oblivious ratio $\alpha(X)$ with respect to X is the optimal solution of the optimization problem:*

$$\alpha(X) = \min_{\mathcal{S}} \alpha : \alpha(X, \mathcal{S}) \equiv \min_{\mathcal{S}} \alpha : \mathcal{S}(X) \subseteq \alpha M . \tag{3}$$

Remark 1. Note that $\alpha(X)$ is equivalent to the conventional notion of oblivious ratio when $X = T$. In other cases, it depends on X and it may even be smaller than 1 when $X \subset T$.

Remark 2. The routing function S minimizing the optimization problem 3 is called *optimal routing function* over the set of traffic matrices X.

Definition 6. *Given a throughput polytope T and a compound routing function $\mathcal{S} = \{(R^i, \mathcal{S}^i) : i \in \mathcal{I}\}$ the cumulative oblivious ratio of the system is defined as follows.*

$$s_\alpha(\{R^i\}_{i\in\mathcal{I}}) = \max_{i\in\mathcal{I}}\{\alpha(R^i)\} . \tag{4}$$

Note that the previous definition of cumulative oblivious ratio is the natural generalization of the oblivious ratio for compound routing functions, recalling that the oblivious ratio can be interpeted as the maximal link over-utilization.

3 Cumulative Oblivious Ratio over Hyper-cubic Regions and a Partitioning Algorithm

In this section, we analyze the cumulative oblivious ratio, when dividing the throughput polytope T into finite hyper-cubic partitions. First, we prove some theoretical properties of these partitions. Then, we introduce a novel algorithm, which, in contrast to the algorithm given in [23], provides provable guarantees on the cumulative oblivious ratio.

3.1 The Theoretical Upper Bound

The proof of our main theorem is based on some basic properties of hyper-cubic throughput regions. Thus, we first introduce an auxiliary theorem summarizing these observations.

Theorem 1. *Let $a_i, b_i \in \mathbb{R}_+, \forall i \in \{1, \cdots, K\}, a_i \leq b_i$ and $H = \bigtimes_{i=1}^{K} [a_i, b_i] \subset \mathbb{R}^K$ a hyper rectangle in the K dimensional throughput space. The optimal routing function $\mathcal{S}_H$ for the throughput region H is block diagonal and $\mathcal{S}_H = \mathcal{S}_b$, where $b = (b_1, \cdots, b_K)$ and $\mathcal{S}_b$ is the optimal routing function for the (singleton) throughput set $\{b\}$.*

Proof. Obviously, the routing function $\mathcal{S}_b$ can be written in block diagonal matrix form, viz., the routing function for a single throughput simply defines the distribution of demand between the routes being used.

Moreover, $\forall\theta \in H : \theta \le b$ results that $\mathcal{S}_b$ is routing function for θ and $\mathcal{S}_b(b) \ge \mathcal{S}_b(\theta)$. The lowest possible oblivious ratio for b is achieved using the $\mathcal{S}_b$ routing function (because of the optimality of $\mathcal{S}_b$), thus $\mathcal{S}_H = \mathcal{S}_b$ is the optimal routing function over the whole region H.

Remark 3. $\overset{K}{\underset{i=1}{\times}}[a_i, b_i] \subset T \Rightarrow \alpha(\overset{K}{\underset{i=1}{\times}}[a_i, b_i]) \le 1$.

Remark 4. The routing function $\mathcal{S}_b$ is distributed linear, i.e.,

$$0 \le \{\mathcal{S}_b\}_{ij} \le \begin{cases} 1 & \text{if } ij \text{ is a block diagonal element} \\ 0 & \text{otherwise} \end{cases} .$$

Let us take a closer look on the network configuration depicted in Fig. 1. We derive the routing function of the hyper-cubic region R^1. The maximum point of this region is the point $(1,1) \in \mathbb{R}^2$, which can be routed using the routing function

$$\mathcal{S}_{(1,1)} : \theta \mapsto \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \theta; \quad \theta \in R^1 .$$

Observe that $\mathcal{S}_{(1,1)}$ is a distributed linear routing function. Additionally, $\mathcal{S}_{(1,1)}$ routes any demand in R^1 without causing link over-utilization, as the first demand is routed using path P_1, while the second demand uses only P_3. Moreover, all other demands in the region R^1 can be routed using this routing function, too.

Now, we are ready to state the main theorem that gives an easy-to-compute upper bound for the cumulative oblivious ratio over finite hyper-cubic partitions.

Theorem 2. *Let $\{H_i^\epsilon\}_{i\in\mathcal{I}}$, $H_i^\epsilon \subset \mathbb{R}_+^K$ be finite partition of any finite hypercube containing the throughput polytope T. Suppose, that H_i^ϵ are mutually exclusive and collectively exhaustive hypercubes with side length ϵ. Moreover, let $a_i = \min\{x : x \in H_i^\epsilon\}$ and let $\mathcal{S}_{a_i}$ be the optimal routing function for the point a_i. For any such partition of T, the cumulative oblivious ratio is given by:*

$$s_\alpha^\epsilon \le 1 + \max_{e\in E}\left\{\tfrac{\xi_e}{c_e}\right\}\epsilon ,$$

where c_e and ξ_e denotes the capacity and number of paths sharing the edge $e \in E$, respectively.

Proof. Introducing the notation $\overline{T} = \mathbb{R}_+^K \setminus T$ we have:

$$\begin{aligned} s_\alpha^\epsilon &= s_\alpha(\{H_i^\epsilon \cap T\}_{i\in\mathcal{I}}) = \max_{i\in\mathcal{I}}\{\alpha(H_i^\epsilon \cap T)\} = \\ &= \max_{\substack{i\in\mathcal{I} \\ H_i^\epsilon \cap T \neq 0}}\{\alpha(H_i^\epsilon \cap T)\} = \max_{\substack{i\in\mathcal{I} \\ H_i^\epsilon \cap T \neq 0 \\ H_i^\epsilon \cap \overline{T} \neq 0}}\{\alpha(H_i^\epsilon \cap T)\} , \end{aligned}$$

where the last equality is valid, because all the boundary points of T have at least oblivious ratio equal to 1 (note that each boundary point of T fills the capacities along at least one edge of the network). Simply put, when calculating the oblivious ratio only hypercubes (hypercube splits) containing at least one boundary point from T count.

Let P_k^e denote the row of the arc-path incidence matrix of k corresponding to link e. Now, we have

$$\begin{aligned}
s_\alpha^\epsilon &\overset{\boxed{1}}{\leq} \max_{e\in E} \max_{\substack{i\in\mathcal{I}\\ H_i^\epsilon\cap T\neq 0\\ H_i^\epsilon\cap\overline{T}\neq 0}} \frac{\sum_{k=1}^{K} P_k^e \mathcal{S}_{a_i}(H_i^\epsilon \cap T)}{c_e} \leq \max_{e\in E} \max_{\substack{i\mathcal{I}\\ H_i^\epsilon\cap T\neq 0\\ H_i^\epsilon\cap\overline{T}\neq 0}} \frac{\sum_{k=1}^{K} P_k^e \mathcal{S}_{a_i}(H_i^\epsilon)}{c_e} \\
&\overset{\boxed{2}}{\leq} \max_{e\in E} \max_{\substack{i\in\mathcal{I}\\ H_i^\epsilon\cap T\neq 0\\ H_i^\epsilon\cap\overline{T}\neq 0}} \frac{\sum_{k=1}^{K} P_k^e \mathcal{S}_{a_i}(a_i + \underline{1}\epsilon)}{c_e} \\
&\overset{\boxed{3}}{=} \max_{e\in E} \max_{\substack{i\in\mathcal{I}\\ H_i^\epsilon\cap T\neq 0\\ H_i^\epsilon\cap\overline{T}\neq 0}} \left\{ \frac{\sum_{k=1}^{K} P_k^e \mathcal{S}_{a_i}(a_i)}{c_e} + \frac{\sum_{k=1}^{K} P_k^e \mathcal{S}_{a_i}(\underline{1}\epsilon)}{c_e} \right\} \\
&\overset{\boxed{4}}{\leq} 1 + \max_{e\in E} \max_{\substack{i\in\mathcal{I}\\ H_i^\epsilon\cap T\neq 0\\ H_i^\epsilon\cap\overline{T}\neq 0}} \frac{\sum_{k=1}^{K} P_k^e \mathcal{S}_{a_i}(\underline{1}\epsilon)}{c_e} \overset{\boxed{5}}{\leq} 1 + \max_{e\in E} \left\{ \frac{\xi_e}{c_e} \right\} \epsilon,
\end{aligned}$$

where $\boxed{1}$ is valid, because instead of the optimal routing function we introduced a special (maybe not optimal) one in the formulae; $\boxed{2}$ is valid because of Theorem 1; $\boxed{3}$ is valid because of the linearity of $\mathcal{S}_{a_i}$ (see Remark 4); $\boxed{4}$ is valid because $a_i \in T$; $\boxed{5}$ is valid because of Remark 4.

Remark 5. During the proof we used the routing function $\mathcal{S}_{a_i}$, which belongs to the strictest class of possible routing functions covered in this paper ($\mathcal{S}_{a_i}$ is a distributed linear routing function; recall Remark 4). Thus, the derived formula is valid for more general affine functions, too (that is, when we let $g \neq 0$).

The results of Theorem 2 empowers us to analyze the behaviour of the *cumulative oblivious ratio over infinitesimally small hyper-cubic regions.*

Corollary 1. $\lim_{\epsilon\to 0} s_\alpha^\epsilon = \lim_{\epsilon\to 0} \left(1 + \max_{e\in E} \left\{\frac{\xi_e}{c_e}\right\} \epsilon\right) = 1$

Simply put, Corollary 1 states that using smaller and smaller hyper-cubic regions as partitions the cumulative oblivious ratio converges to its minimum. In other words, link over-utilization can be eliminated be using sufficiently small regions.

3.2 The Partitioning Algorithm

After the theoretical considerations, we introduce a novel algorithm to compute a hybrid demand-oblivious routing function that guarantees that link over-utilization does exceed a given parameter. The input to the algorithm is, consequently, the desirable oblivious ratio δ. This parameter then, according to Theorem 2, determines an upper bound ϵ on the size of hypercubes we need to cover the throughput polytope as follows:

$$\epsilon \leq \frac{\delta - 1}{\max_{e \in E}\{\xi_e / c_e\}} \ . \tag{5}$$

Calculating the optimal routing functions over these hypercubes (i.e., solving Equation (3) for each hypercube R^i, $i \in \mathcal{I}$, separately), a compound routing function $\mathcal{S} = \{(R^i, \mathcal{S}^i) : i \in \mathcal{I}\}$ can be constructed with guaranteed oblivious ratio δ (see Algorithm 1).

Algorithm 1. Partitioning algorithm

```
function PARTITIONING_OBLIVIOUS_ROUTING($\delta$)
    $\epsilon \leftarrow \frac{\delta-1}{\max_{e\in E}\{\xi_e/c_e\}}$
    $\{H_i^\epsilon\}_{i\in\mathcal{I}} \leftarrow \cup_{i\in\mathcal{I}} H_i^\epsilon$ so that $T \subseteq \{H_i^\epsilon\}_{i\in\mathcal{I}}$
    for $i \in \mathcal{I}$
        $R^i = H_i^\epsilon \cap T$
        $(\alpha^i, \mathcal{S}^i) \leftarrow \min_{\mathcal{S}} \alpha : \mathcal{S}(R^i) \subset \alpha M$
        store $(\alpha^i, \mathcal{S}^i, R^i)$
    end for
end function
```

Hitherto, we have not specified the type of the routing functions used in the partitioning algorithm. According to Theorem 2, it is possible to use any kind of routing functions, however, it is a good decision to restrict the routing functions to affine distributed functions. The main advantage of the distributed routing functions is that the actual splitting ratio for a given demand depends only upon that demand, and it is not dependent upon other demands.

Hence, our scheme will require minimal central control, only to lookup the right routing region based on the actual demands. For this, the central controller periodically determines the actual traffic matrix and checks whether the current demand θ still resides in the current routing region R^i . If yes, no action is taken as the current routing function is correct. Otherwise, the controller searches for a new region. Organizing the regions into a decision tree improves the online complexity to $\mathcal{O}(\log |\mathcal{I}|)$, where $|\mathcal{I}|$ denotes the number of routing regions.

Recall the sample network depicted in Fig. 1. For this configuration, due to the edge between the nodes labelled by 2 and 4, and due to the unit capacity of the edges, $\max_{e\in E} \xi_e = 2$. Suppose that we need to fulfill a given constraint, say, the cumulative oblivious ratio should be less than $\delta = 3$. According to expression (5), we get $\epsilon = 1$ as the size of the hypercubes needed to cover the throughput polytope to attain this δ as the oblivious ratio. A possible partitioning is depicted in Fig. 1c. Then, running Algorithm 1 over this partitioning (that is, solving (3) for each hyper-cubic region in Fig. 1c) over the class of distributed affine functions, we obtain the following routing functions:

$$\mathcal{S}^1 : \theta \mapsto \begin{pmatrix} 1\,0 \\ 0\,0 \\ 0\,1 \end{pmatrix} \theta; \quad \theta \in R^1$$

$$\mathcal{S}^2 : \theta \mapsto \begin{pmatrix} 0\,0 \\ 1\,0 \\ 0\,1 \end{pmatrix} \theta + \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}; \quad \theta \in R^2$$

Note that for this compound routing function the cumulative oblivious ratio is not only less than the desired one, but it is equal to 1. This suggests that our theoretical bound on the oblivious ratio is somewhat pessimistic.

4 Evaluation

Algorithm. During the simulations, we used a bit different version of the previously introduced partitioning algorithm. Instead of the desired oblivious ratio, the parameter of the modified algorithm is rather the iteration depth, that is, the number of cutting planes in each dimension of the throughput space (cutting planes are equally distanced in each dimension). Note that there is a direct connection between the cube-size value ϵ and the iteration depth j, namely, $\epsilon^j = \max_{k\in\mathcal{K}} \frac{m_k}{j}$, where m_k is the (single-commodity) maximum flow for the k-th source-destination pair. We chose this modification of the algorithm because it fits better the purposes of our simulation studies: we can increase the iteration depth and observe the change in the cumulative oblivious ratio, instead of having to guess the latter value and calculating the size of hypercubes used by partitioning. Moreover, in this way it can be guaranteed that for different iteration depth the number of control regions is different, too. However, we cannot warrant anymore that the oblivious ratio decreases when increasing the iteration depth.

Performance Metrics. We used the following three performance metrics to characterize the efficiency of our algorithm: *(i)* the oblivious ratio as defined in the optimization problem (1); *(ii)* the cumulative oblivious ratio as defined in Equation (4); and *(iii)* the number of control regions denoted by $|\mathcal{I}|$. Note that the number of control regions is directly related to the complexity of the routing function: the more regions, the more routing functions must be calculated and stored, and hence the more lookups are needed during the operation of the network.

Simulation Instances. We ran our evaluations on ISP data maps from the Rocketfuel dataset [27]. We used the same method as in [14] to obtain approximate POP-level topologies: we collapsed the topologies so that nodes correspond to cities, we eliminated leaf-nodes and we set link capacities inversely proportional to the link weights. The number of users was increased from 2 to 8, source-destination pairs were chosen according to the bimodal distribution and paths were provisioned maximally node-disjoint. The number of cutting planes (the iteration depth) was increased from 1 to 4. Fifteen evaluation runs, using distributed affine (denoted by 1 in the superscript) and distributed linear (denoted by 2 in the superscript) routing functions, respectively, were conducted on different randomly chosen network samples and the results were averaged. The results are depicted in Table 2.

Apart from the realistic network topologies supplied by the Rocketfuel data set, we also conducted simulations on artificial networks in order to assess the worst case performance of our algorithm. The networks marked by OK-x (for $x = 3, 4, 5$ and 6, resp.) were originally constructed in [16] to derive the worst-case value of the oblivious ratio in specially crafted networks. For a given value x, the OK-x network is constructed as follows. It has $N = \binom{x}{2} + x + 1$ vertices denoted by $a_{i,j}$ for all $1 \leq i \leq j \leq x$, b_i for all $1 \leq i \leq x$ and a vertex denoted by t. The edges of the network are all of unit capacity and are as follows: $(a_{i,j}, b_i)$ and $(a_{i,j}, b_j)$ for all $1 \leq i \leq j \leq x$, and (b_i, t) for $1 \leq i \leq x$. The source-destination pairs are $(a_{j,y}, t)$ for all $1 \leq j \leq y - 1$, and $(a_{y,j}, t)$ for all $y + 1 \leq j \leq x$. The simulation results for these networks are depicted in Fig. 2.

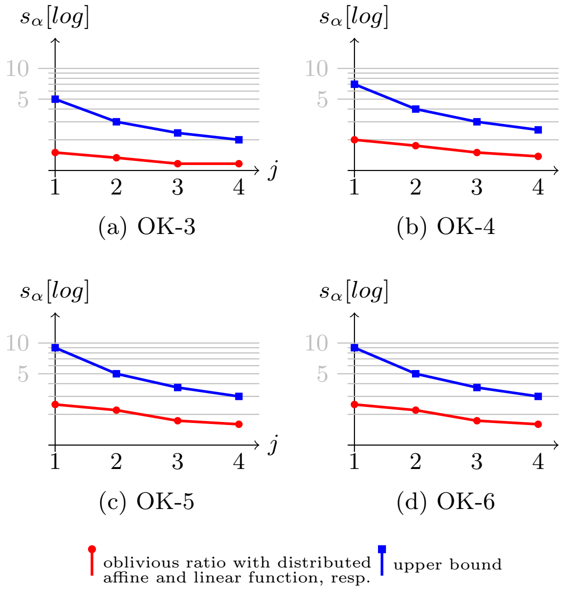

Fig. 2. Measured oblivious ratio for the selected artificial networks and iteration depths

Table 2. Measured cumulative oblivious ratios for the selected rocketfuel topologies. α^2 and α^1 denotes the cumulative oblivious ratio using distributed linear and distributed affine routing functions, resprectively. α_j^2, α_j^1 and β_j denote the average normed cumulative oblivious ratio (i.e, α_j^2 is the average of the normed values $s_{\alpha_j^2}/\alpha^2$) and the calculated average normed upper bound for the j-th iteration, respectively.

	oblivious ratio		2				3				4			
K	α^2	α^1	α_2^2	α_2^1	β_2	$\lvert\mathcal{I}_2\rvert$	α_3^2	α_3^1	β_3	$\lvert\mathcal{I}_3\rvert$	α_4^2	α_4^1	β_4	$\lvert\mathcal{I}_4\rvert$
2	1.012	1.012	0.997	0.990	2.687	4.0	0.998	0.989	2.121	9.0	0.996	0.989	1.838	15.9
3	1.095	1.095	0.989	0.937	2.734	8.0	0.963	0.940	2.129	26.8	0.960	0.932	1.826	62.7
4	1.139	1.139	0.978	0.929	3.228	15.9	0.947	0.919	2.446	76.2	0.945	0.908	2.055	228.0
5	1.119	1.119	0.983	0.940	2.721	31.7	0.960	0.936	2.114	231.6	0.955	0.926	1.810	930.1
6	1.229	1.229	0.981	0.911	3.440	60.8	0.929	0.895	2.566	618.9	0.919	0.878	2.128	3090.4
7	1.231	1.231	0.976	0.924	4.213	114.7	0.933	0.897	3.081	1652.4	0.919	0.878	2.515	10674.7
8	1.269	1.269	0.968	0.914	3.452	244.3	0.923	0.888	2.565	5217.6	0.905	0.867	2.121	45566.7

(a) AS 1239

	oblivious ratio		2				3				4			
K	α^2	α^1	α_2^2	α_2^1	β_2	$\lvert\mathcal{I}_2\rvert$	α_3^2	α_3^1	β_3	$\lvert\mathcal{I}_3\rvert$	α_4^2	α_4^1	β_4	$\lvert\mathcal{I}_4\rvert$
2	1.025	1.025	0.998	0.987	3.414	4.0	0.994	0.982	2.602	8.9	0.991	0.982	2.196	15.5
3	1.062	1.062	0.987	0.972	4.138	8.0	0.980	0.960	3.075	26.4	0.971	0.961	2.543	61.4
4	1.163	1.163	0.986	0.938	5.156	15.6	0.956	0.912	3.725	75.4	0.939	0.904	3.010	220.7
5	1.188	1.188	0.983	0.939	3.836	31.5	0.956	0.905	2.839	211.9	0.931	0.898	2.340	829.3
6	1.208	1.208	0.978	0.942	5.261	62.4	0.959	0.914	3.783	617.3	0.933	0.898	3.045	3231.2
7	1.235	1.235	0.969	0.929	5.997	120.0	0.946	0.906	4.268	1616.7	0.918	0.885	3.404	10077.9
8	1.273	1.273	0.964	0.932	6.893	228.3	0.933	0.900	4.858	4154.8	0.909	0.876	3.840	33333.1

(b) AS 1755

	oblivious ratio		2				3				4			
K	α^2	α^1	α_2^2	α_2^1	β_2	$\lvert\mathcal{I}_2\rvert$	α_3^2	α_3^1	β_3	$\lvert\mathcal{I}_3\rvert$	α_4^2	α_4^1	β_4	$\lvert\mathcal{I}_4\rvert$
2	2.000	1.000	1.000	1.000	3.764	4.0	1.000	1.000	2.843	9.0	1.000	1.000	2.382	16.0
3	2.061	1.096	0.996	0.939	3.822	8.0	0.989	0.930	2.872	27.0	0.987	0.930	2.397	64.0
4	2.196	1.143	0.985	0.942	5.994	16.0	0.965	0.912	4.301	79.2	0.953	0.907	3.454	246.4
5	2.248	1.210	0.989	0.924	3.973	32.0	0.964	0.890	2.947	237.6	0.949	0.875	2.434	985.6
6	2.289	1.175	0.989	0.929	4.626	64.0	0.961	0.902	3.377	712.8	0.946	0.895	2.752	3942.4
7	2.357	1.228	0.991	0.931	5.657	128.0	0.959	0.898	4.054	2138.4	0.933	0.882	3.253	15769.6
8	2.406	1.281	0.983	0.902	6.175	256.0	0.940	0.871	4.394	6269.4	0.923	0.850	3.504	60620.8

(c) AS 3257

	oblivious ratio		2				3				4			
K	α^2	α^1	α_2^2	α_2^1	β_2	$\lvert\mathcal{I}_2\rvert$	α_3^2	α_3^1	β_3	$\lvert\mathcal{I}_3\rvert$	α_4^2	α_4^1	β_4	$\lvert\mathcal{I}_4\rvert$
2	1.044	1.044	0.993	0.971	3.626	4.0	0.983	0.969	2.738	8.9	0.979	0.967	2.294	15.7
3	1.108	1.108	0.980	0.944	4.185	8.0	0.962	0.930	3.092	26.6	0.953	0.924	2.546	62.1
4	1.176	1.176	0.986	0.940	6.610	16.0	0.954	0.907	4.692	78.0	0.933	0.898	3.733	236.8
5	1.190	1.190	0.983	0.932	6.645	31.5	0.956	0.911	4.713	222.8	0.938	0.895	3.746	888.0
6	1.232	1.232	0.977	0.926	5.286	62.7	0.939	0.905	3.797	654.6	0.921	0.884	3.053	3304.5
7	1.314	1.314	0.951	0.911	6.042	124.5	0.906	0.869	4.284	1893.9	0.881	0.847	3.405	12333.0
8	1.334	1.334	0.953	0.912	7.420	235.5	0.910	0.874	5.198	5139.5	0.884	0.850	4.087	43033.1

(d) AS 3967

Evaluation results. The main observations are as follows. First, as the theoretical results suggest the measured oblivious ratio is always less than the predicted one. However, the difference between the two values can become significant. This is the cost we must pay for using rough approximations in order to be able to derive a simple theoretical upper bound. Second, we can clearly see that using the more general distributed affine functions (recall, the ones marked with 1 in superscript) better cumulative oblivious ratio can be obtained. Thus, it might be tempting to use affine routing functions for real life networks. However, the simulation results also show that the benefit is highly network dependent. For example, in case of the network topology AS3257 this difference is quite straightforward, but for the other ones the benefits of the general routing function is not that significant. Third, the benefit caused by increasing the iteration depth is also dependent on the type of the routing function. Using the more general distributed affine function, naturally, yields better results than distributed linear functions.

Lastly, we consider the complexity of the routing (functions) expressed in terms of the depends on the number of routing regions. As it is clearly observable, the number of routing regions increases rapidly with the iteration depth increasing the complexity of the routing function, making the proposed algorithm somewhat inefficient. Contrasting these results to the ones in [23], we see that the algorithm in [23] is more efficient and less complex. The algorithm in this paper, though, is still extremely important, because it provides a firm theoretical worst-case bound on the oblivious ratio, and hence, link over-utilization experienced by the network in case of *any* admissible traffic matrix. This was impossible with previous algorithms.

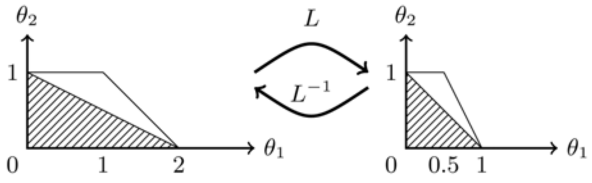

Fig. 3. Due to the down-monotonicity of the throughput polytope, after the linear transformation L the K-simplex is going to be the part of the transformed throughput polytope. Thus, the number of hypercubes covering the standard K-simplex is less than the number of the routing regions.

It is tempting to investigate, whether a hyper-cubic partitioning algorithm exists that is both efficient and simple (i.e., contains at most a polynomial number of hyper-cubic regions) at the same time. Unfortunately, this does not seem to be the case, as the following lower bound on the number of routing regions suggests:

$$|\mathcal{I}| \geq \lceil \frac{v(K)}{(1/j)^K} \rceil .$$

In this expression, $v(K)$ denotes the volume of the standard K-simplex and j, as before, is the iteration depth. The expression compares the volume of the standard K-simplex and the hypercubes with side length $\frac{1}{j}$. We used the observation that the standard K-simplex can be naturally transformed, using a linear transformation, into the interior of the throughput polytope, and vice versa (for more details, consult Fig. 3). What the expression states is that halving the permitted link over-utilization (i.e., halving the value of ϵ) needs about $\mathcal{O}(2^K)$ times more regions.

5 Related Work

Demand-oblivious routing has rich literature [14, 16–19, 21]. There are various results regarding the worst-case performance: Räcke gives a method with poly-logarithmic oblivious ratio in undirected graphs [15], while Azar *et al.* proves that no such bound exists for directed graphs: they give a directed graph of $\binom{k}{2} + k + 1$ nodes, $\binom{k}{2}$ source-destination pairs where the oblivious ratio is at least $\binom{k}{2}$ [16].

Recently, in [28] it was shown that for every network the link over-utilization can be eliminated introducing compound affine routing functions. Here, the routing functions are calculated using multi-parametric linear programs. However, in contrast to the algorithm presented in this paper, the optimal algorithm uses extensive central control, both for setting the traffic splitting ratios and for the selection of the actual routing region.

The idea of compound routing functions inspired the development of a hybrid algorithm [23]. This algorithm simplifies the type of the routing regions, as it generates only hyper-cubic regions. The algorithm uses heuristics when creating the routing regions: in each step the algorithm tries to minimize the cumulative oblivious ratio by slicing each (previously created) routing region into two new hyper-cubic regions. The algorithm is a conquer-and-divide fashion algorithm, i.e., for each region the cutting plane is selected, which minimizes the oblivious ratio. According to [23], by only a few cuts the oblivious ratio can be drastically decreased, however, no convergence results exist.

Our recent algorithm can be viewed as the successor of the previous algorithm. It keeps all of its advantages, but tries to solve the problem from the bottom-up perspective, instead of the top-down one. As a result, it can give a theoretical upper bound for the cumulative oblivious ratio, and it can be used to design routings satisfying any given maximal link over-utilization. Unfortunately, the comparison of the simulation results of the two methods show that our recent method needs more routing regions to achieve the prescribed oblivious ratio than the one in [23].

There are several methods combining basic oblivious routing with some real life measurements to predict future state. These prediction based algorithms work on traffic matrix samples collected during some time interval. Instead of optimizing oblivious ratio over the full throughput polytope, as it is done in demand-oblivious routing, here the optimization is performed – and the routing function is calculated – only over the convex hull of the collected traffic matrices. Thus, these algorithms are effective when the future demands fall into the computed convex hull. Unfortunately, fulfillment of this condition cannot be guaranteed. One method solving this problem is called COPE [7]. In COPE, a penalty envelope is introduced, thus, not only the oblivious ratio is optimized over the convex hull of the collected traffic matrices, but also some penalty function of the routing function over the whole throughput polytope is bounded. Simulation results in [7] show that COPE can achieve efficient resource utilization under a variety of real topologies and scenarios.

There are also several on-line TE methods, which, in contrast to our algorithm, use feedback from the network. For example, REPLEX [13] and DATE [10] are both such methods. They both solve the routing problem in a distributed manner, i.e., there is a given convergence time to calculate the appropriate routing function. In contrast, in our algorithm a central controller is needed to periodically determine the actual traffic matrix and select the right routing region and the (previously, offline calculated) routing function.

6 Conclusion

In this paper, we analyzed the properties of demand-oblivious routing over hyper-cubic regions. We determined an easy to computable worst case bound for the cumulative oblivious ratio, which empowered us to design a hybrid centralized-distributed partitioning algorithm for calculating a compound routing function with upper bounded oblivious ratio (and hence link over-utilization). To the best of our knowledge, this is the first time that a demand-oblivious routing algorithm with provable worst-case performance in directed graphs is presented.

Simulation studies using several real-life network topologies showed that our algorithm indeed admits the theoretical worst-case behavior. In addition, the algorithm successfully decreases the cumulative oblivious ratio in only a few iterations. Though, a closer investigation of the compound routing functions generated unearthed complexity problems: halving the link over-utilization needs about $\mathcal{O}(2^K)$ times as many regions, where K is the dimension of the throughput space.

We were able to prove that link over-utilization can be fully eliminated when decreasing the size of the hyper-cubic regions to infinitesimally small, (i.e., using infinity number of routing regions) regardless of the type of the routing function. The question arises, whether there are networks for which the link over-utilization can be eliminated with using only a finite number of hyper-cubic routing regions. Our future work will focus on finding the class of networks having this property.

References

1. Awduche, D., Chiu, A., Elwalid, A., Widjaja, I., Xiao, X.: Overview and principles of Internet traffic engineering. RFC 3272 (May 2002)
2. Cantor, D.G., Gerla, M.: Optimal routing in a packet-switched computer network. IEEE Transactions on Computer 23(10), 1062–1069 (1974)
3. Fortz, B., Rexford, J., Thorup, M.: Traffic engineering with traditional IP routing protocols. IEEE Communications Magazine 40(10), 118–124 (2002)
4. Roughan, M., Thorup, M., Zhang, Y.: Traffic engineering with estimated traffic matrices. In: IMC 2003: Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement, pp. 248–258 (2003)
5. Zhang, C., Liu, Y., Gong, W., Moll, J., Towsley, R.D.: On optimal routing with multiple traffic matrices. In: INFOCOM 2005, vol. 1, pp. 607–618 (2005)
6. Medhi, D.: Multi-hour, multi-traffic class network design for virtual path-based dynamically reconfigurable wide-area ATM networks. IEEE/ACM Transactions on Networking 3(6), 809–818 (1995)
7. Wang, H., Xie, H., Qiu, L., Yang, Y.R., Zhang, Y., Greenberg, A.: COPE: traffic engineering in dynamic networks. SIGCOMM Comput. Commun. Rev. 36(4), 99–110 (2006)
8. Bertsekas, D.P.: Dynamic behavior of shortest path routing algorithms for communication networks. IEEE Trans. on Automatic Control 27, 60–74 (1982)
9. Chiang, M., Low, S.H., Calderbank, A.R., Doyle, J.C.: Layering as optimization decomposition: A mathematical theory of network architectures. Proceedings of the IEEE 95(1), 255–312 (2007)
10. He, J., Bresler, M., Chiang, M., Rexford, J.: Towards robust multi-layer traffic engineering: Optimization of congestion control and routing. IEEE Journal on Selected Areas in Communications 25(5), 868–880 (2007)
11. Lagoa, C.M., Che, H., Movsichoff, B.A.: Adaptive control algorithms for decentralized optimal traffic engineering in the internet. IEEE/ACM Trans. Netw. 12(3), 415–428 (2004)
12. Kandula, S., Katabi, D., Davie, B., Charny, A.: Walking the Tightrope: Responsive Yet Stable Traffic Engineering. In: ACM SIGCOMM 2005 (August 2005)
13. Fischer, S., Kammenhuber, N., Feldmann, A.: REPLEX: dynamic traffic engineering based on wardrop routing policies. In: Proceedings of CoNEXT 2006, pp. 1–12 (2006)
14. Applegate, D., Cohen, E.: Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs. In: Proceedings of SIGCOMM 2003, pp. 313–324 (2003)
15. Räcke, H.: Minimizing congestion in general networks. In: FOCS 2002, pp. 43–52 (2002)
16. Azar, Y., Cohen, E., Fiat, A., Kaplan, H., Räcke, H.: Optimal oblivious routing in polynomial time. J. Comput. Syst. Sci. 69(3), 383–394 (2004)
17. Wellons, J., Xue, Y.: Oblivious routing for wireless mesh networks. In: ICC 2008, pp. 2969–2973 (May 2008)
18. Li, Y., Bai, B., Harms, J.J., Holte, R.C.: Stable and Robust Multipath Oblivious Routing for Traffic Engineering. In: Mason, L.G., Drwiega, T., Yan, J. (eds.) ITC 2007. LNCS, vol. 4516, pp. 129–140. Springer, Heidelberg (2007)
19. Applegate, D., Breslau, L., Cohen, E.: Coping with network failures: routing strategies for optimal demand oblivious restoration. SIGMETRICS Perform. Eval. Rev. 32(1), 270–281 (2004)

20. Hajiaghayi, M., Kim, J., Leighton, T., Räcke, H.: Oblivious routing in directed graphs with random demands. In: STOC 2005, pp. 193–201 (2005)
21. Bansal, N., Blum, A., Chawla, S., Meyerson, A.: Online oblivious routing. In: SPAA 2003, pp. 44–49 (2003)
22. Towles, B., Dally, W.: Worst-case traffic for oblivious routing functions. In: SPAA 2002, pp. 1–8 (2002)
23. Rétvári, G., Németh, G.: Demand-oblivious routing: distributed vs. centralized approaches. In: INFOCOM 2010 (March 2010)
24. Rétvári, G., Bíró, J.J., Cinkler, T.: Fairness in capacitated networks: A polyhedral approach. In: INFOCOM 2007, vol. 1, pp. 1604–1612 (May 2007)
25. Ziegler, G.M.: Lectures on Polytopes. Graduate Texts in Mathematics, vol. 152. Springer, Heidelberg (1998)
26. Grünbaum, B.: Convex Polytopes. John Wiley & Sons (1967)
27. Mahajan, R., Spring, N., Wetherall, D., Anderson, T.: Inferring link weights using end-to-end measurements. In: IMW 2002: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment, pp. 231–236 (2002)
28. Rétvári, G., Németh, G.: On optimal rate-adaptive routing. In: ISCC 2010, pp. 605–610 (2010)