End-to-end TCP-compatible Backpressure Routing

Christos Liaskos, Konstantinos Alexandris, Siyu Tang, Anindya Das
Applied Network Technology Lab, Huawei Munich
Emails: {christos.liaskos.ext, konstantinos.alexandris, siyutang, anindya.das1}@huawei.com

***Abstract*—Backpressure constitutes a well–known throughput-optimal packet scheduler. Backpressure scheduling decisions are taken based on commodity queue build-ups at the network nodes, and constant status updates among neighbors. Nonetheless, it is also well known for its incompatibility with TCP, which uses congestion control to avoid queue build-ups, while also being sensitive to packet timeouts and their delivery order. These factors limit the use of backpressure in the local packet forwarding plane, e.g., within switches. The present work presents a breakthrough in combining TCP and backpressure at the network layer, overcoming any compatibility issues. The key-idea is to apply the Lyapunov framework cumulatively, over a large set of router hardware clock ticks. This approach allows for the throughput-optimal backpressure decisions to be expressed via regular traffic engineering routing rules, instead of explicit per-packet forwarding directives. Moreover, these throughput-optimizing routing rules are derived using common metrics logged at commercial routers, and not via queue build-ups. Simulation results validate the analytical findings and demonstrate significant throughput gains over state-of-the-art related approaches.**

***Index Terms*—Backpressure, TCP, Lyapunov, Throughput, Datacenter.**

I. Introduction

Backpressure has been a highly influential routing approach [1] which, however, is inherently incompatible with TCP due to the explicit packet-level handling it dictates. According to it, each network node maintains a queue of packets per network destination. Prior to deciding which commodity packets to forward, all nodes exchange their local commodity queue lengths with their neighbors, and calculate the corresponding differential backlog. The commodity queues yielding the largest difference is selected for forwarding to the less burdened neighbor, ideally irrespective to any routing path concern. Moreover, a network-global orchestrator calculates and informs each node of the data transfer rate that must be employed per network commodity.

However, TCP flows perform rate adaptation to avoid queue build-ups, meaning that backpressure, even if the totality of its control signaling is performed as planned, will not yield persistent queue backlog differentiation. Additionally, the overhead of the backpressure control signaling per-packet, coupled with the path route-agnostic forwarding, has been well-documented to yield packet timeouts, out of order deliveries and, eventually, TCP congestion window resets and retransmissions [2]. Thus, related approaches have not targeted network-wide deployments of backpressure and TCP. Instead, the focus has been shifted on the network edge (i.e., the client devices), where backpressure is employed to schedule the data flow injection from the application layer to the transport layer [3].

The present study brings backpressure to the core of a wired network, such as the Internet or a datacenter, leading to a novel, network-wide (i.e., end-to-end), TCP-compatible version. The methodology consists of applying the Lyapunov drift analytical framework cumulatively, over large batches of router packet forwarding actions. This novel approach yields:

- A pure, network-layer version of backpressure *routing* (as opposed to the classic backpressure packet scheduling), which can be implemented solely via routing table entries in the context of regular traffic engineering.
- A novel metric replacing commodity differential backlogs, which does not require queue build-ups of packets waiting in routers.

Subsequently, the study evaluates the novel analysis in a datacenter setting, where throughput optimality has accentuated value. The evaluation outcomes show significant throughput gains (smaller flow completion times) over state-of-the art related solutions [4], as well as well-established max-min utilization approaches [5]. No increase in the frequency of TCP timeouts or in the average packet latency is observed compared to regular TE (i.e., changing the network's routing rules for any reason), given that the presented CIDAR operation *is expressed as a TE process itself*. A discussion on the synergy of the proposed approach with other traffic engineering goals at the network layer, and schedulers at the packet layer complements the study.

The novel analysis is given in Section II, leading to the algorithmic formulation of the TCP-compatible backpressure traffic engineering. Evaluation follows in Section III. The study is concluded in Section IV.

II. TCP-compatible Backpressure Routing

The course towards deriving a TCP-compatible backpressure routing process follows these steps: i) Model the queue dynamics occurring at a single router hardware clock tick, ii) derive the time-aggregated queue dynamics over a length of time coinciding with the next network traffic engineering update, and iii) apply the Lyapunov drift minimization over the aggregated dynamics.

Regarding the general network model, we initially consider a bidirectional n-node graph as the topology, while the leaf-spine is studied as a special case in the evaluation (Section III). We consider that the traffic is classified according to a set $\mathcal{C}$ of routing tuples upon entering a node. For instance, in destination-based routing, a $c \in \mathcal{C}$ can be a specific IP or an

IP subnet. In Software-Defined Networks (SDN), it can be a full TCP tuple comprising source and destination IPs and ports.

A. Clock-tick traffic dynamics

We consider the traffic at a given node n, destined towards c, during the time interval t to $t+\Delta t$. Δt is a small time interval, representing the duration of router's complete workflow (denoted as *clock tick*), i.e.: (1) Empty the internal buffer for any destination c, by forwarding packets to the router's exiting interfaces. This is done as a first step, to avoid buffer overflows. (2) Then, read the incoming data and store it to the internal buffer per $c \in \mathcal{C}$. (3) Finally, add the locally generated data to the internal buffer as well. Steps 2 and 3 can occur in any order.

The queue dynamics for this operation are expressed by the well-known relation, $\forall n, c$ [6, p. 54]:

$$U_{n,c}^{t+\Delta t}=\max\left\{0,U_{n,c}^{t}-O_{n,c}^{t\rightarrow t+\Delta t}\right\}+I_{n,c}^{t\rightarrow t+\Delta t}+G_{n,c}^{t\rightarrow t+\Delta t}, \quad (1)$$

where $U_{n,c}^{t}$ is the amount of buffered data for c, in node n, at time t, and: $O_{n,c}^{t\rightarrow t+\Delta t}$ is the outgoing traffic leaving node n routed via rule c; $I_{n,c}^{t\rightarrow t+\Delta t}$ denote incoming traffic to the node from neighboring nodes; $G_{n,c}^{t\rightarrow t+\Delta t}$ will denote the *local traffic variation component* as follows.

The $I_{n,c}^{t\rightarrow t+\Delta t}$ quantity will be considered as non-varying, while any variation, e.g., owed to the TCP traffic dynamics or the local data generation, is captured within the $G_{n,c}^{t\rightarrow t+\Delta t}$ component.

B. Time-aggregated traffic dynamics

Let T be a traffic engineering time interval, i.e., an interval after which the routing rules at the network nodes need to be updated. (Naturally, the update does not need to affect every single node, nor to occur strictly periodically). Additionally, let RTT denote a maximal (boundary) round trip time of the active TCP flows in the network. We will assume that T fulfills the following:

$$T=K\cdot\Delta t,\ K\in\mathbb{N}^{*},\text{where } T\gg RTT,\ T\gg\Delta t\Rightarrow K\gg 1. \quad (2)$$

The router traffic dynamics every T is essentially a time aggregated application of the Δt dynamics expressed by equation (1). In order to study the throughput optimality at the T-scale, we will extract the Lyapunov drift upper bound between the transition from $U_{n,c}^{t}$ to $U_{n,c}^{t+T}$ as follows. (The drift is defined as $\Delta L(t)=L(t+T)-L(t)$, where $L(t)=\sum\limits_{\forall n,c}\left\{U_{n,c}^{t}\right\}^{2}$. An upper bounded drift means that the system is stable [6]).

Let's consider some values of $U_{n,c}^{(k)}$, with $k\in\mathcal{K}=\{1,\ldots,K\}$, using a shorthand writing for brevity as:

$$U_{n,c}^{(k)}=\max\left\{0,U_{n,c}^{(k-1)}-O_{n,c}^{(k)}\right\}+I_{n,c}^{(k)}, \quad (3)$$

where $U_{n,c}^{(k)}\leftrightarrow U_{n,c}^{t+k\cdot\Delta t}$, $U_{n,c}^{(k-1)}\leftrightarrow U_{n,c}^{t+(k-1)\cdot\Delta t}$, $O_{n,c}^{(k)}\leftrightarrow O_{n,c}^{t+(k-1)\cdot\Delta t\rightarrow t+k\cdot\Delta t}$, and $I_{n,c}^{(k)}\leftrightarrow I_{n,c}^{t+(k-1)\cdot\Delta t\rightarrow t+k\cdot\Delta t}+G_{n,c}^{t+(k-1)\cdot\Delta t\rightarrow t+k\cdot\Delta t}$. Subsequently, we consider the identity [6, p. 53]:

$$V\leq\max\{0,U-\mu\}+A\Rightarrow\{V\}^{2}\leq\{U\}^{2}+\{\mu\}^{2}+\{A\}^{2}-2U\cdot\{\mu-A\}, \quad (4)$$

and we apply it to Eq. (3), $\forall\ k\in\mathcal{K}$, obtaining:

$$\left\{U_{n,c}^{(1)}\right\}^{2}\leq\left\{U_{n,c}^{(0)}\right\}^{2}+\left\{O_{n,c}^{(1)}\right\}^{2}+\left\{I_{n,c}^{(1)}\right\}^{2}-2\cdot U_{n,c}^{(0)}\cdot\left\{O_{n,c}^{(1)}-I_{n,c}^{(1)}\right\}, \quad (5)$$

...

$$\left\{U_{n,c}^{(k)}\right\}^{2}\leq\left\{U_{n,c}^{(k-1)}\right\}^{2}+\left\{O_{n,c}^{(k)}\right\}^{2}+\left\{I_{n,c}^{(k)}\right\}^{2}-2\cdot U_{n,c}^{(k-1)}\cdot\left\{O_{n,c}^{(k)}-I_{n,c}^{(k)}\right\}, \quad (6)$$

...

$$\left\{U_{n,c}^{(K)}\right\}^{2}\leq\left\{U_{n,c}^{(K-1)}\right\}^{2}+\left\{O_{n,c}^{(K)}\right\}^{2}+\left\{I_{n,c}^{(K)}\right\}^{2}-2\cdot U_{n,c}^{(K-1)}\left\{O_{n,c}^{(K)}-I_{n,c}^{(K)}\right\}. \quad (7)$$

We will rewrite Eq. (6) as follows:

$$\left\{U_{n,c}^{(k)}\right\}^{2}\leq\left\{U_{n,c}^{(k-1)}\right\}^{2}+\left\{O_{n,c}^{(k)}-U_{n,c}^{(k-1)}\right\}^{2}+\left\{I_{n,c}^{(k)}+U_{n,c}^{(k-1)}\right\}^{2}-2\cdot\left\{U_{n,c}^{(k-1)}\right\}^{2}. \quad (8)$$

Applying summation by parts to all inequalities for $k\in\mathcal{K}$, rewritten as in Eq. (8), we obtain:

$$\left\{U_{n,c}^{(K)}\right\}^{2}-\left\{U_{n,c}^{(0)}\right\}^{2}\leq\sum_{k=1}^{K}\left\{O_{n,c}^{(k)}-U_{n,c}^{(k-1)}\right\}^{2}+\sum_{k=1}^{K}\left\{I_{n,c}^{(k)}+U_{n,c}^{(k-1)}\right\}^{2}-2\cdot\sum_{k=1}^{K}\left\{U_{n,c}^{(k-1)}\right\}^{2}. \quad (9)$$

We proceed to sum both parts for all n and for all c:

$$\sum_{\forall n,c}\left\{U_{n,c}^{(K)}\right\}^{2}-\sum_{\forall n,c}\left\{U_{n,c}^{(0)}\right\}^{2}\leq\sum_{\forall n,c}\sum_{k=1}^{K}\left\{O_{n,c}^{(k)}-U_{n,c}^{(k-1)}\right\}^{2}+\sum_{\forall n,c}\sum_{k=1}^{K}\left\{I_{n,c}^{(k)}+U_{n,c}^{(k-1)}\right\}^{2}-2\cdot\sum_{\forall n,c}\sum_{k=1}^{K}\left\{U_{n,c}^{(k-1)}\right\}^{2}. \quad (10)$$

Considering that the Lyapunov drift is defined as $\Delta L(t)=L(t+T)-L(t)$, where $L(t)=\sum\limits_{\forall n,c}\left\{U_{n,c}^{t}\right\}^{2}$, we rewrite relation (10) as:

$$\Delta L(t)\leq\sum_{\forall n,c}\sum_{k=1}^{K}\left\{O_{n,c}^{(k)}-U_{n,c}^{(k-1)}\right\}^{2}+\sum_{\forall n,c}\sum_{k=1}^{K}\left\{I_{n,c}^{(k)}+U_{n,c}^{(k-1)}\right\}^{2}-2\cdot\sum_{\forall n,c}\sum_{k=1}^{K}\left\{U_{n,c}^{(k-1)}\right\}^{2}. \quad (11)$$

Remark 1. Our goal is to minimize the right-hand side of relation (11), denoted as $RHS_{(11)}$. In other words, at time t, we want to take the optimal routing decisions that minimize the $RHS_{(11)}$, i.e., for the system operation within $[t,t+T]$.

Reminding that

$$I_{n,c}^{(k)}\leftrightarrow I_{n,c}^{t+(k-1)\cdot\Delta t\rightarrow t+k\cdot\Delta t}+G_{n,c}^{t+(k-1)\cdot\Delta t\rightarrow t+k\cdot\Delta t}. \quad (12)$$

We proceed to write:

$$I_{n,c}^{(k)}=I_{n,c}^{t+(k-1)\cdot\Delta t\rightarrow t+k\cdot\Delta t}\leq\sum_{\forall d(n)}\int_{t+(k-1)\cdot\Delta t}^{t+k\cdot\Delta t}\mu_{d(n)\rightarrow n}^{c}dt=\Delta t\cdot\sum_{\forall d(n)}\mu_{d(n)\rightarrow n}^{c}. \quad (13)$$

where $d(n)$ stands for the neighboring node of n, and $\mu^c_{d(n)\to n}$ is the maximum data rate destined to c and incoming to node n from link $d(n)\to n$. In the same manner, $O^{(k)}_{n,c}$ becomes:

$$O^{(k)}_{n,c}=O^{t+(k-1)\cdot\Delta t\to t+k\cdot\Delta t}_{n,c}\leq\sum_{\forall b(n)}\int_{t+(k-1)\cdot\Delta t}^{t+k\cdot\Delta t}\mu^c_{n\to b(n)}dt$$
$$=\Delta t\cdot\sum_{\forall b(n)}\mu^c_{n\to b(n)}, \quad (14)$$

where $b(n)$ represents the neighboring node of n, at the end of each outgoing link. Moreover, for ease of presentation we set:

$$\mathring{\mu}^c_n=\sum_{\forall b(n)}\mu^c_{n\to b(n)}. \quad (15)$$

Using relations (13), (14) and (15), relation (11) becomes:

$$\Delta L(t)\leq\sum_{\forall n,c}\sum_{k=1}^{K}\left\{\Delta t\cdot\mathring{\mu}^c_n-U^{(k-1)}_{n,c}\right\}^2+$$
$$\sum_{\forall n,c}\sum_{k=1}^{K}\left\{\Delta t\cdot\sum_{\forall d(n)}\mu^c_{d(n)\to n}+G^{(k)}_{n,c}+U^{(k-1)}_{n,c}\right\}^2-$$
$$2\cdot\sum_{\forall n,c}\sum_{k=1}^{K}\left\{U^{(k-1)}_{n,c}\right\}^2. \quad (16)$$

We proceed to optimize the $RHS_{(16)}$ subject to the routing decisions $\mu^c_{n\to b(n)}$. The sufficient conditions for the presence of a minimum are:

$$\begin{cases}\frac{\partial RHS_{(16)}}{\partial\mu^c_{n\to b(n)}}=0, & (i)\\ 0<\mathcal{H}\left(\frac{\partial^2 RHS_{(16)}}{\partial\mu^c_{n\to b(n)}\cdot\partial\mu^c_{m\to b(m)}}\right)<\infty, & (ii)\\ RHS_{(16)}\text{ is convex w.r.t. }\mu^c_{n\to b(n)},\mu^c_{m\to b(m)}, & (iii)\end{cases} \quad (17)$$

where m denotes a node, $\mathcal{H}$ is the Hessian matrix [7] and the requirement $0<\mathcal{H}<\infty$ refers to each of its elements. From condition (17-i) we obtain:

$$2\cdot\sum_{k=1}^{K}\left[\Delta t\cdot\sum_{\forall d(b(n))}\mu^c_{d(b(n))\to b(n)}+U^{(k-1)}_{b(n),c}+G^{(k)}_{b(n),c}\right]$$
$$+2\cdot\sum_{k=1}^{K}\left[\Delta t\cdot\mathring{\mu}^c_n-U^{(k-1)}_{n,c}\right]=0\Longleftrightarrow \quad (18)$$

$$\Delta t\cdot\sum_{k=1}^{K}\left[\sum_{\forall d(b(n))}\mu^c_{d(b(n))\to b(n)}+\mathring{\mu}^c_n\right]-$$
$$\left[\sum_{k=1}^{K}U^{(k-1)}_{n,c}-\sum_{k=1}^{K}U^{(k-1)}_{b(n),c}-\sum_{k=1}^{K}G^{(k)}_{b(n),c}\right]=0,\forall n,c. \quad (19)$$

It is not difficult to show that conditions (17-ii, iii) are satisfied, since it holds:

$$\frac{\partial^2 RHS_{(16)}}{\partial\mu^c_{n\to b(n)}\cdot\partial\mu^c_{m\to b(m)}}\propto\Delta t\geq 0,\forall n,m. \quad (20)$$

Firstly, we notice that the term $\Delta t\cdot\sum_{k=1}^{K}\left[\sum_{\forall d(b(n))}\mu^c_{d(b(n))\to b(n)}+\mathring{\mu}^c_n\right]$ includes *maximum* data rates μ. Thus, this term constitutes an upper bound for the incoming traffic towards $b(n)$. Subsequently, we focus on the following case:

$$\Delta t\cdot\sum_{k=1}^{K}\left[\sum_{\forall d(b(n))}\mu^c_{d(b(n))\to b(n)}+\mathring{\mu}^c_n\right]>$$
$$\left[\sum_{k=1}^{K}U^{(k-1)}_{n,c}-\sum_{k=1}^{K}U^{(k-1)}_{b(n),c}-\sum_{k=1}^{K}G^{(k)}_{b(n),c}\right]. \quad (21)$$

Here, equality (e.g., as in Eq. (19)) is not applicable. Nonetheless, as expressed by Eq. (20), the $RHS_{(16)}$ is convex. Thus, the $RHS_{(16)}$ is minimal when the first derivative is nearest to zero, i.e., when:

$$\Delta_{n,c}(t)=\left[\sum_{k=1}^{K}U^{(k-1)}_{n,c}-\sum_{k=1}^{K}U^{(k-1)}_{b(n),c}-\sum_{k=1}^{K}G^{(k)}_{b(n),c}\right]. \quad (22)$$

Finally, the routing decision at node n affecting the traffic towards node c is:

$$b^*(n)=\arg\max_{b(n)}\{\Delta_{n,c}(t)\} \quad (23)$$

where $b^*(n)$ is the best neighbor of n for offloading c-data to.

C. CIDAR: A queue-less backpressure metric

Notably, the re-routing decision expressed via relations (22) and (23), derived from the time-aggregated traffic analysis, is based on the quantities $\sum U$ and $\sum G$.

We will denote the first quantity as the CIDAR metric (Continuously Integrated Data Rate):

$$\text{CIDAR}^n_c=\sum_{k=1}^{K}U^{(k-1)}_{n,c} \quad (24)$$

The CIDAR metric does not consider a queue build-up at any node n. On the contrary, *CIDAR simply aggregates all pass-through data bytes, i.e., traversing n while being dispatched via routing rule c over a time duration of $T=K\cdot\Delta t$*. As such, CIDAR can be readily calculated in existing routers via standard traffic accounting tools (such as the well-known `iptables` command in Linux and CISCO systems [8]).

As a side-note, CIDAR is calculated per routing rule $c\in\mathcal{C}$ at each node. However, routing rules may be expressed in aggregated format in one node, and more specific form in a neighbor (and vice-versa). In such cases, the more aggregated routing rule is virtually decomposed to specific routing rules at the same granularity rule as the neighbor.

Regarding the quantity $\sum G$, recall from relation (1) that it expresses the generic, *local traffic variation component*. We proceed to specialize it to the TCP dynamics as follows.

Proposition 2. *Let a node n and its neighbors $b(n)$. The node n must decide where to offload traffic using the routing rule c. Let node $b^*(n)$ as selected to forward the traffic of c to a link with physical capacity $\mathcal{B}$. Let $\mathcal{N}_{b^*(n)}$ be the number of the TCP flows that currently compete over this link. Moreover let $\mu^c_{b^*(n)}$ be the current, steady-state TCP rate over this link. We can employ the quantity G to represent the new TCP steady-state that would result from offloading traffic from n to $b^*(n)$ as:*

$$G^{(k)}_{b^*(n),c}=\left(\frac{\mathcal{B}}{\mathcal{N}_{b^*(n)}+\mathbf{1}\left(c\notin\mathcal{N}_{b^*(n)}\right)}-\mu^c_{b^*(n)}\right)\cdot\Delta t,\forall i, \quad (25)$$

and

$$\sum_{k=1}^{K}G^{(k)}_{b^*(n),c}=\left(\frac{\mathcal{B}}{\mathcal{N}_{b^*(n)}+\mathbf{1}\left(c\notin\mathcal{N}_{b^*(n)}\right)}-\mu^c_{b^*(n)}\right)\cdot\Delta t\cdot T.$$

In other words, G can express the new TCP equilibrium, assuming that T is much larger than the TCP convergence interval due to condition (2).

Therefore, it becomes evident that i) CIDAR-based, and ii) $\sum G$-based selection of neighbors to offload traffic via a new routing rule are not necessarily aligned: on one hand, CIDAR-based selection does not pose a limit on how many flows can

be offloaded to neighbors. On the other hand, $\sum G$ calls for a careful selection of such flows, in order to account for the competition among TCP flows over common links.

D. Flow selection for throughput optimization: Problem formulation

In general, the problem of flow selection for throughput optimization can be formulated as a generic optimization problem under an objective function, with exclusivity constraints on the decision variables:

$$\max_{\mathbf{x}\in\{0,1\}^{|\mathcal{V}|}} \quad F(\mathbf{x}) \tag{26}$$

$$\text{s.t.} \quad \textstyle\sum_{l\in\mathcal{L}^n} x^n_{l,c}\leq 1, \forall c\in\mathcal{C}^n, \forall n\in\mathcal{N}, \tag{27}$$

where $\mathcal{V}=\mathcal{L}^1\times\mathcal{C}^1\times\cdots\times\mathcal{L}^{|\mathcal{N}|}\times\mathcal{C}^{|\mathcal{N}|}$ and $\mathbf{x}\triangleq\left[x^1_{1,1}\ldots x^n_{l,c}\ldots x^{|\mathcal{N}|}_{|\mathcal{L}^{|\mathcal{N}|}|,|\mathcal{C}^{|\mathcal{N}|}|}\right]^\top$, and following the notation below:

- $\mathcal{L}^n$: set of links at router n.
- $\mathcal{C}^n$: set of commodities expressing active TCP flow rules at router n.
- $\tilde{\mathcal{C}}^n$: complementary commodities expressing constant bitrate flows (CBR, e.g., UDP) at router n.
- $\mathcal{N}$: set of routers in the topology.

$|\cdot|$: cardinality of a set. Last, we define as $x^n_{l,c}\in\{0,1\}$, the decision (binary) variable that corresponds to the selection of the commodity c to be associated with link l at router n and the subvector of $\mathbf{x}$:

$$\mathbf{x}^n_l=\left[x^n_{l,1},\cdot,x^n_{l,c},\cdots x^n_{l,|\mathcal{C}^n|}\right]^\top,\ l\in\mathcal{L}^n, n\in\mathcal{N}. \tag{28}$$

The objective function $F(\mathbf{x})$ is expressed as:

$$F(\mathbf{x})=\textstyle\sum_{n\in\mathcal{N}}\sum_{l\in\mathcal{L}^n}\left(\frac{\mathcal{B}_l-\sum_{\tilde{c}\in\tilde{\mathcal{C}}^n}\text{CBR}_{l,\tilde{c}}}{\sum_{c\in\mathcal{C}^n}x^n_{l,c}}\cdot\sum_{c\in\mathcal{C}^n}\left(\mathbf{x}^n_l\right)\Delta R^n_c\right)=$$

$$F(\mathbf{x})=\textstyle\sum_{n\in\mathcal{N}}\sum_{l\in\mathcal{L}^n}\left(\mu(\mathbf{x}^n_l;\text{CBR}^n_l)\cdot\sum_{c\in\mathcal{C}^n}\left(\mathbf{x}^n_l\right)\Delta R^n_c\right) \tag{29}$$

where

$$\mu(\mathbf{x}^n_l;\text{CBR}^n_l)=\frac{\mathcal{B}_l-\sum_{\tilde{c}\in\tilde{\mathcal{C}}^n}\text{CBR}_{l,\tilde{c}}}{\sum_{c\in\mathcal{C}^n}x^n_{l,c}} \tag{30}$$

and ΔR^n_c is the difference in the CIDAR metric of eq. (24) between: i) router n, and ii) the next hop router of commodity c, which is subject to the link selection $x^n_{l,c}$.

In more detail, in relation (29), we target to allocate flows to the network links in order to maximize the output rate $\mu\left(\mathbf{x}^n_l;\text{CBR}^n_l\right)$ per link biased by a CIDAR-based weight, $\sum\Delta R^n_c$. Notice that $\mu\left(\mathbf{x}^n_l;\text{CBR}^n_l\right)$ expresses the TCP fair-share over a link, i.e., all TCP flows compete equally for the link bandwidth minus the portion taken up by CBR flows.

The exclusivity constraint in the formulated max-weight problem can be defined and interpreted as:

$$\textstyle\sum_{l\in\mathcal{L}^n}x^n_{l,c}\leq 1, \forall c\in\mathcal{C}^n, \forall n\in\mathcal{N}. \tag{31}$$

The latter expresses the requirement that each flow is assigned at up to one link. Here, notice that a solution search satisfying (31) can yield high complexity due to the combinatorial nature of the problem. Therefore, we proceed to present an algorithmic approach that bypasses constraint (31) by using a default routing tree, and focusing on optimizing relation (29).

E. Flow selection for throughput optimization: Algorithmic approach

We proceed to propose an algorithm to serve the $F(\cdot)$ objective function and the exclusivity constraint. As a groundwork, we seek to ensure that any routing rules derived via the optimization methodology yield: i) always assured connectivity for any source-destination in the network bypassing constraint (31), and ii) loop freedom. To these ends, we will consider that:

- Each router n is equipped with a set of *underlying, default routing rules* for each destination (e.g., via OSPF or similar approach). These default routing rules offer assured connectivity and can be derived, e.g., via OSPF. These default routing rules are executed with the lowest priority, i.e., they are activated only if there is no other overriding routing rule, that is specifically marked as *prioritized* (e.g., in the SDN manner). All routing rules derived via the CIDAR-driven optimization are dynamic, hold until the next actuation time (e.g., $t+T$ in the periodic case) and are marked as *prioritized*.
- The ΔR^n_c notation in Eq. (29) implied the CIDAR difference between the CIDAR of flow rule c at router n and the CIDAR of flow c at *neighboring router* $b(n)$ connected to n via link l. In order to avoid the formation of loops, we will consider only neighboring routers that are equidistant or closer to the end-destination that node n for rule c, over the default routing rules.

Moreover, we introduce the notion of CIDAR preference tables, denoted as PT. According to it, each node n in the studied topology contacts its neighbors, $b_1(n)$, $b_2(n)$, etc., and asks for their CIDAR value pertaining to flow rule $c\in\mathcal{C}$. Subsequently, it calculates the CIDAR differences, ΔR, filling in its CIDAR PT with $\langle b(n),\Delta R\rangle$ potential flow rules. A row of this PT for a flow rule referring to commodity c_1 could then look as follows:

Flow rule for comm.: c_1	CIDAR preferences: $\langle b_1(n),\Delta R_1\rangle$,$\langle b_2(n),\Delta R_2\rangle$,...

Essentially, the outcome of the optimization process is to select at most one $\langle b(n),\Delta R\rangle$ potential flow rules at each row of each CIDAR preference table(n), in order to maximize the objective function of relation (29). Given the combinatorial nature of this problem we present a greedy algorithm denoted as CIDAR-TE (CIDAR-driven traffic engineering), formulated as Algorithm 1.

CIDAR-TE receives as inputs the CIDAR preference tables, $PT(n)$, for each network router, n, and outputs the optimal routing rule (i.e., which $<b(n),\Delta R^n_c>$ pair, or no pair) to keep at each row of every $PT(n)$. The algorithm follows two stages (also indicated by the comment lines in Algorithm 1):

- Flatten and join all PT entries, and sort the resulting array by descending ΔR metric value.
- Iterate over the resulting entry and follow the process for every entry: (a) Tentatively activate the routing rule and log the resulting optimization objective fitness via equation (29). (b) If the rule activation led to better fitness objective, keep it

Algorithm 1 The centralized, offline CIDAR-TE algorithm.

INPUTS: The CIDAR preference tables $PT(n)$, $\forall n$.
OUTPUT: The $global\_solution_{n,c} : \langle b(n), \Delta R_c^n \rangle$, $\forall n, c$.
//1.Flatten, join and sort all PT entries.
$Sorting_{n,c,i} \leftarrow \emptyset$;
for each router n,
  for each commodity c in router n,
    $global\_solution_{n,c} \leftarrow \emptyset$;
    for each $\langle b(n), \Delta R_c^n \rangle$ in $PT(n)$ row c, (iterated via i)
      $Sorting_{n,c,i}.push(\langle n, c, i, \Delta R_c^n \rangle)$;
$Sort(Sorting_{n,c,i}, by\ \Delta R_c^n, DESC)$;
  //Create an empty global solution.
for each router n,
  for each commodity c in router n,
    $global\_solution_{n,c} \leftarrow \emptyset$;
//2.Greedily subset $Sorting_{nci}$ as a global solution.
$best\_solution \leftarrow \emptyset$;
for each $\langle n, c, i, \Delta R_c^n \rangle$ in $Sorting_{n,c,i}$,
  $backup \leftarrow Clone(global\_solution)$;
  if $global\_solution_{n,c} = \emptyset$ **then**
    $global\_solution_{n,c} = PT(n) \rightarrow row_c \rightarrow FlowRule_i$;
  $f \leftarrow F(global\_solution)$; //Eq.(29)
  if $best\_solution = \emptyset$ or $f > F(best\_solution)$ **then**
    $best\_solution \leftarrow Copy\_of(global\_solution)$;
  else
    $global\_solution \leftarrow backup$;
return $global\_solution$;

in the final solution. Else, revoke it and proceed to the next element of the flattened/sorted array.

Remarks. CIDAR-TE: i) Initiates by dropping all prioritized, dynamic routing rules produced at previous executions, keeping only the default ones. ii) Is an offline algorithm. Only the final solution is actually deployed to the routers, which is produced after the process concludes. iii) Requires $O(M \cdot log(M) + M)$ steps to conclude, M being the total number of PT entries (i.e., for all network routers). This is due to the sorting at stage 1 (requiring $O(M \cdot log(M))$ steps) and the serial iteration at stage 2 (requiring $O(M)$ steps). iv) CIDAR-TE can be implemented as either *centralized* or *distributed* algorithm. The only input of the algorithm is the set of $PT(n)$ tables. The difference for a centralized/distributed implementation is the way these tables are gathered.

III. Evaluation

In this Section we proceed to evaluate CIDAR-TE and the related analytical conclusion of relation (23). We employ an open-source network TE simulator implemented in JAVA [9]. The evaluation parameters are:

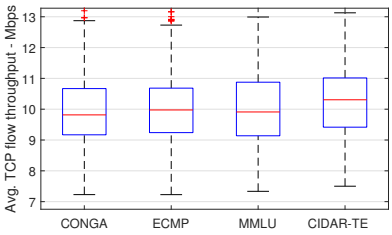
a) Topology: All runs consider a leaf-spine topology with 2 spines and 2 leaves, given its wide-spread use in modern datacenters [4]. A number of $N_c = 5$ clients are added at each leaf. The link capacities are C_{LS} =1 Gbps (leaf-spine) and $C_{CL} = 100$ Mbps (client-leaf). This choice represents an over-provisioned topology with a $1:10$ C_{CL} to C_{LS} ratio. It constitutes a scaled-down version of link capacities used in actual datacenters (which can be, e.g., 100 Gbps and 10 Gbps, respectively [4]). The scaling-down is performed to keep the simulation runtimes tractable. Apart from the over-provisioned topology, we also study the under-provisioned case in the runs below. The link latency is set to 0.1 μsec globally and ethernet interfaces are considered everywhere. The TE controller is collocated with a spine and communicates with all other nodes in an out-of-band fashion.

b) Traffic generation: We assume that each client hosts: i) a web server listening at port 1000, and ii) a set of web-clients that each connects to one server. The web-client is an application that works as follows. First, a server is picked at random, but not from the same leaf as the client. The web server popularity for this pick follows the Zipf(θ) distribution (θ is stated per set of simulation runs) [10]. The web-client proceeds to open a TCP connection (New Reno, advertised window: 65535 B) and send an HTTP request with size 350 B. The server replies by sending a data file whose total size is picked at random from a reported distribution [11]. The MSS for this transfer is picked at random from another reported distribution [12]. Once the transfer is completed, the TCP connection is closed and the web-client enters a ThinkTime interval whose duration is exponentially distributed with 1 ms average. Then, the web-client proceeds to re-initiate the same connection repeat this cycle, until the end of the simulation. The total number of web-clients per client host is picked at the beginning of the simulation from the corresponding distribution [11]. Additionally, we introduce background traffic in the form of one UDP flow per client, whose rate (CBR) is picked randomly in the range $[0, {}^{C_{CL}}/_{N_c}]$ prior to every TE action. This background traffic forces the TCP flows to re-compete for a new fair-share, thereby testing the efficiency of the compared schemes in challenging traffic conditions.

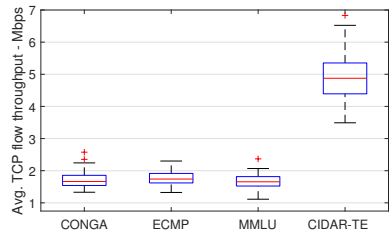
c) Metrics of Performance: For each run, we log: i) the average TCP flow throughput, ii) the average TCP flow completion time, and iii) the average end-to-end packet delay [13].

d) Compared Schemes : i) CONGA, first presented in [4] constitutes a benchmark solution for Clos and leaf-spine topologies in datacenters. Its principle of operation relies on the timely aggregation of traffic congestion levels from all routers and the subsequent allocation of TCP flows to the least congested network paths. For the sake of the comparison, we employ the same 2×2 topology and operational parameters as in [4]. ii) The Max-Min Link Utilization (MMLU) approach, integrated in many TE techniques such as [14], is a general approach for traffic load balancing. MMLU is given K-alternative paths for each source-destination in a network. Subsequently, it monitors the current data rate of each traffic flow and re-routes it to one of the K-paths in a way that maximizes the minimum link utilization across the network. The problem is formulated as a mixed-integer linear program that is solved heuristically. iii) The well-known ECMP scheme [15].

While none of the compared schemes requires a periodic operation (i.e., the TE being repeated upon a fixed time duration, T), we employ it for all of them for the purpose of fair comparison since our focus is to deduce the throughout efficiency of the network-layer TE actions per scheme. In order to pick a value of T, we operate via a simple approximation as follows. First, we study the employed flow size PDF [11], and observe that ~85% of the flows have a size ≤ 0.85 MB. Second, in the employed 2×2 topology, we can deduce



(a) Overprovisioned leaf-spine links (1 Gbps capacity).



(b) Underprovisioned leaf-spine links (100 Mbps cap.).

Fig. 1: Evaluation outcomes, performing a parametric variation study of the compared schemes. The objective is to compare CIDAR-TE to related schemes under realistic traffic conditions, for over-provisioned and under-provisioned leaf-spine topologies. Each boxplot captures 100 runs.

that the upstream/downstream throughput fair-share per client is $t_{fs} = 2 \times 1\,\text{Gbps}/N_c\,\text{clients}$. Finally, if we consider n parallel TCP flows per client as as a rule of a thumb, then $T = (0.85\text{MB} \cdot 8/\,(t_{fs}/n)\ \text{Mbps}) = 3.4 \cdot n \cdot N_c$ msec suffice to ideally service the ~85% of the active flows within one full TE interval, i.e., with a maximum of one route modification during their lifetime.

Finally, all runs were repeated 100 times to yield dependable values, and each run lasts for ~30 sec simulated system time. Moreover, we vary θ randomly in the range $[0 : 0.25 : 1.0]$ per each of the 100 runs, in order to increase the traffic diversity of the ensuing benchmarking results.

Results. The results are shown in Fig. 1 ($N_c = 5$). In the over-provisioned case (Fig. 1a), the equal distribution of the flows over the leafs (i.e., essentially ECMP) is the optimal strategy, given that the leaf-spine links cannot be bottlenecks. Thus, all compared solutions behave similarly, with a marginal foothold for CIDAR-TE. In the under-provisioned case (Fig. 1b), the leaf-spine links become potential bottlenecks. CIDAR-TE, taking into consideration the future TCP fair share optimization, yields a clear x3 improvement over the related schemes. Moreover:

For the *overprovisioned case*, all schemes yielded a flow completion time of $\approx 0.155\,sec$, and an end-to-end packet delay of $\approx 2.2\,msec$. The variations around these average values are aligned to those presented in Fig. 1a and the corresponding illustration is skipped for brevity. This similarity in performance is expected given that the compared schemes yield almost the same average TCP flow throughput.

For the *underprovisioned case*, CONGA, ECMP and MMLU yielded flow completion times of $\approx 0.92\,sec$, and end-to-end packet delays of $\approx 14\,msec$. The variations around these average values are analogous to those presented in Fig. 1b. CIDAR-TE yielded a flow completion time of $0.317\,sec$ (25^{th} percentile: $0.29\,sec$, 75^{th} percentile: $0.35\,sec$), and an end-to-end packet delay of $4.5\,msec$ (25^{th} percentile: $4.2\,msec$, 75^{th} percentile: $5.2\,msec$). (Once again, the corresponding illustrations were skipped for brevity).

IV. Conclusion

Network throughput maximization constitutes an impactful goal, especially in modern datacenters. While the well-known backpressure packet scheduling framework offers analytically-optimal throughput maximization, its workflow is inherently incompatible with the TCP protocol. The present work bridged this gap yielding a novel form of backpressure routing that is fully TCP compliant. The key-idea was to apply the Lyapunov network stability framework cumulatively over large sets of individual packet scheduling decisions. This allowed the resulting throughput optimization actions to be expressed as routing rules at the network layer, as opposed to explicit packet scheduling actions of the classic backpressure. Simulations demonstrated the significant gains in throughput over well-known state of the art traffic engineering approaches.

References

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, 1992.

[2] Z. Jiao *et al.*, "Backpressure-based routing and scheduling protocols for wireless multihop networks," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 102–110, 2016.

[3] H. Seferoglu and E. Modiano, "TCP-aware backpressure routing and scheduling," in *ITA'14*, pp. 1–9.

[4] M. Alizadeh *et al.*, "Conga: Distributed congestion-aware load balancing for datacenters," in *Proceedings of ACM SIGCOMM*, 2014, pp. 503–514.

[5] Y. Zhang, D. Leonard, and D. Loguinov, "Jetmax: scalable max–min congestion control for high-speed heterogeneous networks," *Computer Networks*, vol. 52, no. 6, pp. 1193–1219, 2008.

[6] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource Allocation and Cross-Layer Control in Wireless Networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–144, 2005.

[7] J.-B. Hiriart-Urruty, J.-J. Strodiot, and V. H. Nguyen, "Generalized Hessian matrix and second-order optimality conditions for problems withC 1,1 data," *Applied Mathematics & Optimization*, vol. 11, no. 1, pp. 43–56, 1984.

[8] "Traffic accounting with linux iptables." [Online]. Available: https://catonmat.net/traffic-accounting-with-iptables

[9] C. Liaskos *et al.*, "A novel framework for modeling and mitigating distributed link flooding attacks," in *IEEE INFOCOM'16*, pp. 1–9.

[10] L. Durbeck, J. G. Tront, and N. J. Macias, "Energy efficiency of zipf traffic distributions within facebook's data center fabric architecture," in *IEEE PATMOS'15*, pp. 152–160.

[11] A. Greenberg *et al.*, "Vl2: A scalable and flexible data center network," in *ACM SIGCOMM'09*, pp. 51–62.

[12] "Packet size distribution function for equinix-nyc.dira.20181018-130000.utc," Oct 2018. [Online]. Available: https://www.caida.org/catalog/datasets/trace_stats/nyc-a/2018/equinix-nyc.dira.20181018-130000.utc.df/

[13] K.-C. Leung, V. O. Li, and D. Yang, "An overview of packet reordering in transmission control protocol (tcp): problems, solutions, and challenges," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 4, pp. 522–535, 2007.

[14] X. Li and K. L. Yeung, "Traffic engineering in segment routing networks using milp," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1941–1953, 2020.

[15] H. Zhang *et al.*, "SDN-based ECMP algorithm for data center networks," in *IEEE ITA'14*, pp. 13–18.