REBATE: a REpulsive-BAsed Traffic Engineering Protocol for Dynamic Scale-Free Networks

Dmitrii Chemodanov[*], Flavio Esposito[‡], Prasad Calyam[*] and Andrei Sukhov[†]
[*]University of Missouri-Columbia, USA; [‡] Saint Louis University, USA; [†]Samara National Research University, Russia;
[†]National Research University Higher School of Economics, Russia
Email: *duman190@gmail.com, calyamp@missouri.edu, flavio.esposito@slu.edu, amskh@yandex.ru*

Abstract

Nowadays the abundance of IoT devices has the potential of changing our lives dramatically, but brings new routing and traffic orchestration challenges for the next-generation Internet providers: core routers are already overwhelmed, see *e.g.*, the routing table size growth problem. Although some researchers still argue whether or not the next-generation networks should feature scale-free properties, recent results have shown benefits of embedding such scale-free networks in a hyperbolic space of negative curvature. Specifically, this allows geometrically route packets by using only a local topology knowledge (*i.e.*, with average $O^*(1)$ space-time complexity) at no extra communication overhead (*i.e.*, without routing protocols). To our knowledge, however, there is no Traffic Engineering (TE) protocol with the aforementioned properties that can be used in dynamic scale-free networks. In this paper, *we propose the first to our knowledge REpulsive-BAsed Traffic Engineering (REBATE) protocol for dynamic scale-free networks*. REBATE is built upon dual principles of the demand-aware TE and fundamentals properties of hyperbolic spaces. Using trace-driven numerical simulations, we then show how REBATE can reduce the maximum link utilization up to 25% when compared to a common geometric routing-based traffic steering. Although REBATE can perform worse than common demands-aware and oblivious TE approaches, we think that our work should pave the way for more efficient TE in the next-generation dynamic scale-free networks.

Keywords: Dynamic Scale-Free Networks, Geometric Routing, Traffic Engineering, Next-Generation Internet, IoT.

1. Introduction

The advent of Internet of Things (IoT) changes our world dramatically [1]. Nowadays, a large pletora of Internet-connected sensors is available on the market, attempting to make anything 'smart', including entire cities [2]. Such abundance of IoT devices creates new routing and traffic orchestration challenges for the next-generation Internet providers. Particularly, Internet today already struggles from large routing tables in its core routers [3, 4]. This is due to the fact that current Internet routing best practices linearly depend on the number of devices N, and thus have $O(N)$ space complexity [5]. In the case when Internet-like networks (*e.g.*, the next-generation edge networks [6]) feature a *scale-free* nature [7, 8], they can benefit from a tree-based routing schemes that operates on the data of $O(logN)$ space complexity [5].

To avoid potential confusion, by *scale-free* networks we mean networks whose average node degree k follows a power law, *e.g.*, $P(k)\sim k^{-\gamma}$ and have strong clustering properties [7]. Thus, if some parts of network have the average node degree k that does not depend on N (*i.e.*, have amortized $O^*(1)$ space complexity to store information about neighbors), we can say that these parts of the network feature scale-free properties. Although some researchers argue that current Internet doesn't have a scale-free nature [9], or that such network properties are rare in the real-world [10], other recent studies keep showing benefits of utilizing scale-free properties for Internet [7, 11, 12].

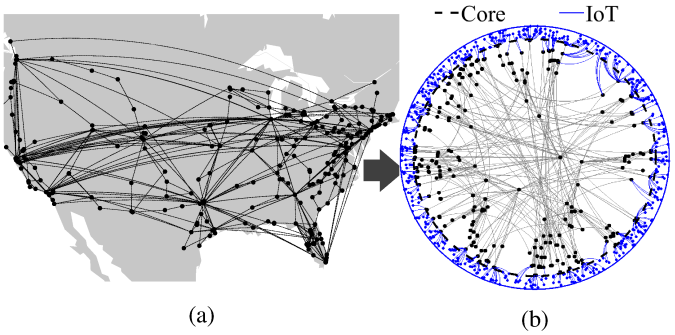

Figure 1: (a) A core network of the current US Internet segment that spans 7 Tier-1 US providers and comprises of 286 Point of Presence (PoP) nodes and 534 links. (b) Its augmented with IoT version embedded in the hyperbolic space of negative curvature using Poincare Disk model.

As it is still questionable whether or not next-generation networks should feature scale-free properties, *in this paper we just hypothesize that this is possible (at least for some parts of these networks) and further investigate scale-free benefits.*

Compact routing solutions such as [5] mainly operate on static networks. On the contrary, the next-generation (core) scale-free network augmented with IoT may be highly dynamic [13]. It has been shown that for such dynamic scale-free networks we need at least $O(N)$ number of control messages to converge to a new (routing) network state using compact routing [5]. Be-

ing an intractable problem for dynamic scale-free networks, recent solutions proposed greedy approaches for forwarding (also called geometric routing) to cope with this problem [7, 14]. Particularly, the idea is to embed dynamic scale-free networks into the hyperbolic space of the negative curvature, *i.e.*, assign virtual coordinates to each node. Once coordinates are assigned, a geometric routing uses a gradient descent of the distance function to deliver packets, *i.e.*, each node forwards packets to the next closest to the destination node [11, 14]. The resulting routing strategy requires only $O^*(1)$ amortized space-time complexity (*i.e.*, only local information about k neighbors) stored on nodes and introduces no communication overhead [7].[1] Fig. 1 illustrates our case study: a map of US Tier1 core network is embedded in a hyperbolic space with negative curvature, that in turn is used by geometric routing to deliver packets without a global topology knowledge.

In addition to routing, providers deploy different Traffic Engineering (TE) algorithms to optimally orchestrate their network traffic under a set of physical and application constraints. For instance, they may wish to minimize the maximum link utilization to balance their network load and minimize its overall congestion probability [15], or they can attempt to share fairly their network capacity among known applications to increase the overall utilization [16, 17]. For applications that pose strong latency requirements, providers may also need to take into account time-sensitive networking constraints [18, 19, 20]. We classify existing TE approaches that exist today into two main types: oblivious and demands-aware. The former uses only a global topology knowledge for the traffic optimization [15], whereas the latter uses both global network and application demands knowledge [16, 17]; all TE approaches need at least $O^*(N)$ space-time complexity on average to store information used for TE optimizations. To the best of our knowledge, there is no suitable TE solution that can operate with data of $O^*(1)$ space-time complexity on average (*i.e.*, that can use only a local information) to optimize traffic in dynamic scale-free networks.

Note that in this paper, by scale-free TE we do not intend to propose or replace what is currently used in every Wide-Area-Networks of the Internet, since large segments are subject to local ISP or IXP policies [5]. However, *scale-free TE can be crucial for the next-generation Internet segments of hardly controllable, large dynamic subnetworks, namely, those made by edge [6, 13] and IoT devices [1].*

Our contributions. *In this paper, we propose REpulsive-BAsed Traffic Engineering (REBATE) — the first to our knowledge dynamic scale-free TE protocol.* To optimize traffic with REBATE, we utilize fundamental properties of hyperbolic metric spaces[2] as well as dual principles of the optimal demands-aware TE. Particularly, we use the theory of potentials [21] to build a metric whose gradient descent allows to (*i*) deliver packets and (*ii*) preserve the dual feasibility of the optimal TE in a best-effort manner. Intuitively, we use potential fields to 'repeal' traffic away from routes that may unnecessarily make TE performance worse, *e.g.*, unnecessarily increase the network maximum link utilization.

Furthermore, using an actual US Tier-1 core network case study in our trace-driven numerical simulations (see Fig. 1), we found how our REBATE is superior to a traffic steering technique that utilizes commonly adopted Gravity Pressure Greedy Forwarding (GPGF) algorithm [7, 14].[3] Particularly, REBATE can reduce the maximum link utilization in the network up to 25% when using the hyperbolic mapping coordinate assignment technique [11], and gain up to 20% of such reduction when greedy embedding assignment technique is used instead [14]. Despite our REBATE performs worse compared to both demands-aware and (common) oblivious TE, *we think that our results are promising and should pave the way for more efficient TE in future networks with a dynamic scale-free nature.*

Paper organization. The rest of the paper is organized as follows: In Section 2, we discuss related work. Section 3 outlines TE problem and motivation. Section 4 describes our REBATE model, and in Section 5, we build on this model to propose a practical REBATE protocol. Section 6 presents our performance evaluation results that show pros and cons of our proposed approach. Section 7 concludes the paper.

2. Related Work

The overall TE optimization problem is to assign flows with specified demands to a set of paths within a given capacitated network while optimizing some function, *e.g.*, minimizing the maximum link utilization. This problem is known in optimization literature as the multi-commodity flow problem and can be solved in polynomial time using linear programming (LP) if flows are splittable (*i.e.*, can be assigned to multiple paths) [22]. In this paper, we roughly divide all approaches to *demands-aware* TE (if flow demands are known) and *oblivious* TE (if such demands are unknown).

Demands-aware TE: Due to practical hardware granularity limitations, flows have finite quantization constraints, and thus the NP-hard integer programming has to be used to solve the multi-commodity flow problem [16, 17]. To avoid potential complexity intractabilities, the authors in [16] have proposed a greedy algorithm that selects paths taking into account practical flow quantization granularities. Their algorithm is 25x faster than corresponding LP solution with a good optimality performance. SWAN [17] instead reduces a feasible space of the general multi-commodity flow problem by decomposing it to a set of k-shortest path candidates for each specified flow. In our evaluation, we use (best possible) LP-based demands-aware TE algorithm for the general multi-commodity flow problem assuming no flow quantization constraints to achieve the maximum link utilization lower-bound.

[1] Note that similarly to conventional routing approaches the size of coordinates (or size of node addresses) is considered to be fixed.

[2] Note that maintaining hyperbolic coordinates in presence of network dynamics has been studied in prior works [12]. Thus, finding the best such technique to use with REBATE is out of scope for this paper.

[3] Note that only geometric routing-based traffic steering solutions can be currently applied in dynamic scale-free networks. Thus, in our simulations (available at `https://github.com/duman190/rebate`), we omit comparison with other common TE approaches [15, 16, 17] and only use (lower-bound) demands-aware TE solution to access optimality performance of REBATE.

Table 1: REBATE comparison with relevant TE approaches.

TE			
Scheme:	*Demands-aware*	*Need global network knowledge*	*Space-time complexity*
B4 [16]	✓	✓	$O^*(N)$
SWAN [17]	✓	✓	$O^*(N)$
COYOTE [15]	✗	✓	$O^*(N)$
SMORE [23]	✗	✓	$O^*(N)$
PATHBOOK [24]	✗	✓	$O^*(N)$
REBATE	✗	✗	$O^*(1)$

Table 2: REBATE comparison with relevant geometric routing approaches.

Geometric routing			
Protocol:	*Relies on network state*	*Space-time complexity*	*Costs of communication*
GDSTR [27, 28]	✓	$O^*(logN)$	$O(N)$
MDT [29]	✓	$O^*(1)$	$O(N)$
WEAVE [30]	✓	$O^*(1)$	**?**
GPGF [14]	✗	$O^*(1)$	N/A
REBATE	✗	$O^*(1)$	N/A

?: difficult to bound without considering topology dynamics (failures/mobility).

Oblivious TE: In contrast to demands-aware TE, oblivious TE has no *a priori* knowledge on flow demands which is common for most Internet providers today [23]. The first remarkable result in the oblivious routing has been shown by Räcke [25] — there exists an oblivious routing scheme with the worst congestion ratio of $\Omega(logN)$ factor from the optimum (demands-aware) routing. To further improve traffic optimization, recent oblivious TE scheme COYOTE [15] takes directed acyclic graphs of the network (that requires at least $O(N)$ space-time complexity) constructed at each node as an input to compute path weights with the use of geometric programming. To further overcome potential expressiveness limitations of existing protocols such as Open Shortest Path First (commonly used for Equal-Cost Multi-Path forwarding), COYOTE inserts 'lies' to adjust their path weights. To find paths for their subsequent weights optimization, the recent semi-oblivious TE scheme SMORE [23] uses both a special 'routing tree' structure and the network graph. Thus, SMORE similarly to COYOTE needs at least $O(N)$ space-time complexity to optimize traffic. Another recent oblivious TE scheme in [24] aims to generate a 'pathbook' – a small set of paths which minimize routing costs. Specifically, the authors use a multi-commodity flow problem to find pathbooks for all commodities, and hence, their solution also requires at least $O(N)$ space-time complexity.

We found no TE approach with amortized $O^*(1)$ space-time complexity that can be used in dynamic scale-free networks. In this paper, we propose the first to our knowledge TE protocol called REBATE that operates only with a local network information and can be used in the next-generation Internet segments of dynamic scale-free networks.

Geometric routing in hyperbolic metric spaces. Compass routing is the first known geometric routing approach where each node forwards packets to the closest to their destination neighbor. As it is subject to the *local minimum problem* — packets can stuck at nodes that have no closer to the destination neighbors than themselves [26] — several *stateless* and *stateful* recovery approaches have been proposed.

Stateful routing: In stateful routing, some network knowledge is used to recover packets from local minima. Thus, the authors in [27, 28] propose Greedy Distributed Spanning Tree Routing which requires a distributed spanning tree construction (of $O^*(logN)$ space-time complexity on average) to guarantee delivery and recover from a local minimum. More recent works [29, 30, 31] show that spanning trees are sensitive to network dynamic topologies and mobility, *e.g.*, within IoT networks. Thus, they can be subject to large communication costs of $O(N)$ [31]. More recent geometric routing approaches such as MDT and WEAVE [29, 30] can cope with topology dynamics to some extent. For example MDT constructs Delaunay triangulation (DT) graphs to make nodes aware of their Delaunay neighbors (with amortized $O^*(1)$ space-time complexity) for a local minimum recovery. In contrast to MDT, WEAVE protocol relies on partial paths (of amortized $O^*(1)$ space-time complexity) stored in packet headers whereas learns them when greedy forwarding packets. The down-side of these techniques is that the maintenance of either Delaunay neighbors or partial routing tables can be subject to large communication costs [29]. For instance, MDT's per node communication cost can be bounded as $O(N)$ - the maximum length of the loop-free path [29]. At the same time, the authors of WEAVE [30] do not expose such costs which are hard to bound without considering different topology dynamics such as failures and mobility. Finally, neither MDT nor WEAVE has been previously applied for hyperbolic metric spaces.

Stateless routing: In contrast to stateful routing, stateless routing does not rely on any network knowledge (*i.e.*, has $O^*(1)$ space-time complexity on average) for packets recovery. One of the first stateless geometric routing solution which guarantees delivery was Greedy Perimeter Stateless Routing [32]. To recover from a local minimum, this protocol uses face routing which requires strong assumptions such as unit disk and planar graphs. This however cannot be guaranteed in hyperbolic metric spaces [33]. To avoid packets recovery, Kleinberg et al. [33] use spanning d-regular trees for greedy embedding, *i.e.*, for an assignment of hyperbolic space coordinates to greedy forward a packet without facing a local minimum. The authors in [14] extend the work of Kleinberg et al. [33] and propose the greedy embedding scheme for general irregular trees. Moreover, they show that due to inaccurate greedy embedding caused by topology dynamics, packets can still get stuck in local minima. To this aim, they propose GPGF protocol which is shown to have guaranteed packet delivery [14]. To recover from local minima, GPGF counts the number of node visits (storing that information in packet headers) to press packets from local minima until greedy forwarding can resume. The key idea beyond pressure recovery is a greedy forwarding gradient descent property — once a packet reaches a location closer to the destination, there is no way how the packet can be forwarded back to the previous location of a local minimum.

Although the authors in [31] indicate that such a recovery 'imposes a large overhead to the packets header especially in large-scale topologies', we found no proof to support this

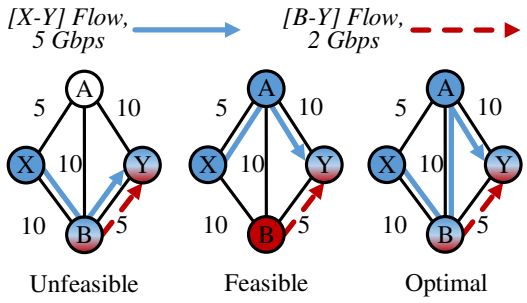

Figure 2: Illustrative example of a TE flow assignment to paths (numbers denote link capacities in Gbps) while minimizing the maximum link utilization: (left) an unfeasible flow assignment due to a link $B-Y$ capacity violation; (center) a feasible but suboptimal flow assignment due to the link $X-A$ maximum link utilization of 1; (right) the feasible and optimal flow assignment with the minimum possible maximum link utilization of 0.5.

argument in the literature. Thus, similarly to prior work [7, 14] we use GPGF to steer traffic in scale-free networks embedded in hyperbolic metric spaces and compare its optimization performance with REBATE. Finally, we also use GPGF pressure mode within our REBATE protocol to recover packets from local minima and propose a hash function that identifies the uniqueness of visited nodes to reduce the protocol header size. By using trace-driven simulations, we assess this header size under practical Time-To-Leave (TTL) constraints and show how it does not introduce a significant overhead.

3. Traffic Engineering Problem and Motivation

In this section, we give an overview of our Traffic Engineering (TE) problem. We then define what we mean by optimal TE and how we will try to achieve it with our REBATE approach. Finally, we discuss a few challenges in TE for scale-free networks, and how duality theory can help overcoming some of them.

3.1. Traffic Optimization Objective and Example

Today, TE is especially beneficial for providers who controls both network infrastructure and applications that use this network. For instance, large companies such as Google and Microsoft use TE techniques to fairly share their inter-datacenter network among their services to drive their link utilization to near 100% [16, 17]. In this paper, we focus however on a more traditional goal of *minimizing the maximum link utilization* to better balance the network load and minimize a congestion probability common for oblivious TE [15, 23].

Fig. 2 illustrates an example of a traffic optimization process that assigns 2 flows with specified demands to a set of path in the capacitated network of 4 nodes. The goal is to minimize the maximum link utilization, and in the first case (see Fig. 2, left) we can see an unfeasible flow assignment due to capacity violations of $B-Y$ link. In the second case (see Fig. 2, center), the flow assignment is now feasible but is still suboptimal with the maximum link $X-A$ utilization of 1. Finally, the third case (see Fig. 2, right) illustrates the optimal flow assignment where the maximum link utilization of 0.5 is achieved.

3.2. Optimal demands-aware TE Problem

The problem of assigning flows with specified demands on top of the capacitated network is known as the multi-commodity flow problem. It can be formulated as a linear program assuming that flows are splittable and can be assigned to multiple paths. [22]. Formally, to minimize the maximum link utilization we solve the below linear program which we denote as primal (**P**).

Problem 1 (**P**). *Given a network graph $G = (N, L)$, where N is set of nodes and L is the set of links, and a set of flows F, let f_{ij}^{st} be a positive variable denoting an amount of st flow with demand D^{st} on the link ij that has capacity C_{ij}, and let α be a positive variable denoting the maximum link utilization, the primal (**P**) TE optimization problem can be formulated as follows:*

$$\textit{minimize } \alpha \tag{1}$$

subject to
Link Utilization Constraints:

$$\alpha - \sum_{st \in F} f_{ij}^{st} / C_{ij} \geq 0, \forall ij \in L \tag{2}$$

Flow Conservation Constraints:

$$\sum_{k \in N} f_{ik}^{st} - \sum_{l \in N} f_{li}^{st} = \begin{cases} D^{st}, & i = s \\ 0, & i \notin \{s, t\}, \forall i \in N, st \in F \\ -D^{st}, & i = t \end{cases} \tag{3}$$

where symbols and notations of sets, parameters and variables are summarized in Table 3.

Note that the optimal solution of **P** with $\alpha \geq 1$ means that no feasible solution exists in practical settings.

3.3. Scale-Free TE Challenges and Dual Problem Relevance

While observing TE primal problem, one can notice that its solution requires global knowledge of the network graph ($G = G(N, L)$) as well as of the flow demands (D^{st}) which requires application awareness [16, 17]. Thus, solving **P** for dynamic scale-free TE is intractable due to the fact that the latter operates on the knowledge of $O(1)$ size. For example, each node can use only local information of one-hop neighbors to make routing decisions due to scale-free networking limitations. As a result, even without an application demands awareness, we cannot store the entire network graph on a single node, *i.e.*, we cannot use oblivious TE [15] too. That is what makes the dynamic scale-free TE problem unique.

To solve the dynamic scale-free TE problem, we aim to benefit from knowledge mined from hyperbolic metric spaces to route packets. Our goal is to use this knowledge to behave similarly to optimizers that solve the demands-aware TE problem while routing packets. However, our dynamic scale-free TE protocol cannot behave as optimizers (*e.g.*, network simplex) that solve **P**. The optimizer starts improving solutions from a feasible point. Thus, when solving **P** we can try to mimic its behavior only when all flow demands are met by redirecting them through better alternative paths. This procedure can be hard due to the scale-free networking nature — we may never get to a state in which all flow demands are satisfied. Fortunately, we can solve **P** by solving its dual problem (**D**) as follows:

Table 3: Symbols and Notations

Sets:		
N	$\triangleq$	Set of network graph nodes
L	$\triangleq$	Set of network graph links
F	$\triangleq$	Set of network flows
Variables:		
α	$\triangleq$	Positive variable that denotes the maximum link utilization
f_{ij}^{st}	$\triangleq$	Positive variable that denotes an amount of st flow on the link ij
y_{ij}	$\triangleq$	Dual positive variable that corresponds to the primal link utilization constraints (see Eq, 2)
w_i^{st}	$\triangleq$	Dual unrestricted variable that corresponds to the primal flow conservation constraints (see Eq, 3)
Parameters:		
C_{ij}	$\triangleq$	Link ij capacity
D^{st}	$\triangleq$	Demand of st flow

Problem 2 (D). *Given a network graph $G = (N, L)$, where N is set of nodes and L is the set of links and a set of flows F, let y_{ij} be a positive dual variable corresponding to the primal link utilization constraint (see Eq. 2), and let w_i^{st} be an unrestricted dual variable corresponding to the primal flow conservation constraint (see Eq. 3), the dual (**D**) TE optimization problem can be formulated as follows:*

$$maximize_{y,w} \sum_{st \in F} D^{st}(w_s^{st} - w_t^{st}) \tag{4}$$

subject to
Dual Utilization Constraint:

$$\sum_{ij \in L} y_{ij} \leq 1 \tag{5}$$

Dual Node Constraints:

$$w_i^{st} - w_j^{st} \leq y_{ij}/C_{ij}, \quad \forall st \in F, ij \in L \tag{6}$$

where symbols and notations of sets, parameters and variables are summarized in Table 3.

To solve **D** we can start from a feasible solution when all dual variables are initialized to zero, *i.e.*, when none of the corresponding flow packets are sent. Our goal is to route packets while trying to satisfy flow demands and preserve feasibility of problem **D**. If we can do so, than, once all flow demands are satisfied, we can reach **P** optimality due to the strong duality theorem — at optimality **P**=**D** [34].

In the next section, we build a metric whose gradient descent allows to forward packets in a way that (*i*) delivers them to their destinations and (*ii*) preserves feasibility of the dual problem **D** in a best-effort manner.

4. REBATE: Model

In this section, we build our REBATE approach for scale-free TE based on the duality principles presented in the previous section. To this end, we first consider two common hyperbolic coordinate assignment techniques for dynamic scale-free networks [7, 14, 11]. Our goal is to understand to what extent greedy forwarding in hyperbolic metric spaces matches with a TE optimization process. Based on duality principles and hyperbolic space properties we then derive a metric to deliver packets and preserve **D** feasibility.

4.1. Hyperbolic Properties and Coordinates Assignment

It has been shown that to route a packet without global topology knowledge, it is sufficient for each node to know merely its own and all its neighbors coordinates [7, 14] (greedy forwarding or geometric routing). Note that this information has $O(1)$ space complexity. Once a packet with the destination coordinates arrives, it is forwarded to the neighbor closest to the destination. If no such neighbor exists, we say that the node faces the local minimum and apply corresponding actions, *e.g.*, drop the packet. To assign coordinates to each node in the network so that no local minima exist, a *greedy embedding* can be used [7, 14, 33]. The *greedy embedding* can be defined as follows [33]:

Definition 1 (greedy embedding)**.** *Given a network graph $G(N)$ where N is a set of nodes, and given a metric space (M, d), the **greedy embedding** is a mapping function $\mathcal{E} : N \rightarrow M$ where for any distinct pair of nodes (u, v), node u has a neighbor l such that $d(\mathcal{E}(l), \mathcal{E}(v)) < d(\mathcal{E}(u), \mathcal{E}(v))$.*

It has been shown that 2-D coordinates (*i.e.*, of $\mathbb{R}^2$ plane) are sufficient for *greedy embedding* if the hyperbolic space of a negative curvature is used. As in prior work [14, 33], we use the Poincaré Disk model to construct the $\mathbb{R}^2$ hyperbolic plane. In this model, the coordinates are represented by complex numbers from a set $\{z_i \in \mathbb{C} : |z_i| \leq 1\}$. Points located at the Poincaré Disk border $|z_i| = 1$ are said to be located at ∞. The distance between any two arbitrary points z_i and z_j can be found as following:

$$d(i, j) = cosh^{-1}\left(1 + \frac{2|z_i - z_j|^2}{(1 + |z_i|^2)(1 + |z_j|^2)}\right) \tag{7}$$

Greedy Embedding Assignment Technique. To greedy embed an arbitrary network in the Poincaré Disk hyperbolic plane, existing solutions (*e.g.*, in [14, 33]) require a (distributed) construction of a spanning tree. In this paper, we use a greedy embedding algorithm proposed in [14] to embed networks, *i.e.*, assign virtual hyperbolic coordinates z_i to each node i. Fig. 3 shows an example of greedy embedding of a simple tree-like network (that comprises 15 nodes) to the Poincaré Disk hyperbolic plane. Note that if network dynamics (e.g., failures/mobility) affect an underlying spanning tree, the greedy embedding property (see Def 1) is not guaranteed [14].

Hyperbolic Mapping by Replaying Network Growth. Although the above coordinates assignment strategy ensures that *greedy embedding* property (see Def. 1) is satisfied, in practice, if tree links of the underlying spanning tree fail, the property defined in Def. 1 is violated [14]. Alternatively, hyperbolic coordinates z_i can be assigned to each node i by replaying the network growth [11]. This technique is called 'hyperbolic mapping' and has the goal of trying to place nodes with higher node degree closer to the origin of the hyperbolic coordinate system. In our evaluation, we use this hyperbolic mapping technique on real traces of the US Tier1 network (see Fig. 1b).

Space complexity of hyperbolic coordinates assignment techniques. The core network requires at least $O(N)$ space complexity for its *initial* either greedy embedding or hyperbolic

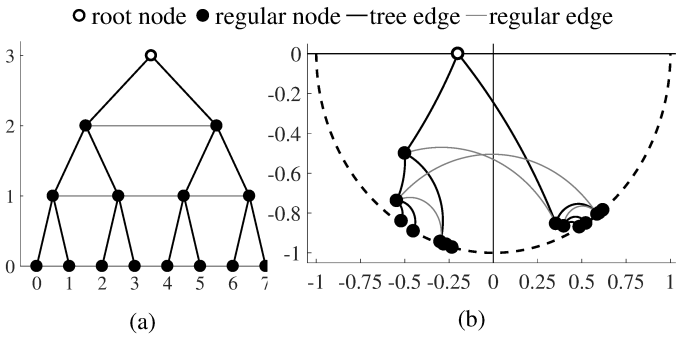

Figure 3: Illustrative example of a simple tree-like network (a) *greedy embedded* in the Poincaré Disk hyperbolic plane after the minimum spanning tree construction (b): as shown in [14, 33], greedy forwarding always delivers packets in the Poincaré Disk plane when using distance function in Eq. 7.

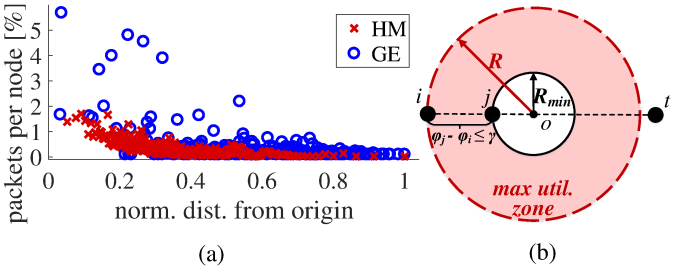

Figure 4: (a) Percent of packets forwarded by nodes in the US Tier1 network w.r.t. nodes' distances from the origin: when packets forwarded in either hyperbolically mapped (HM) or greedy embedded (GE) network, its negative curvature forces nodes located closer to the origin to forward more packets. (b) Illustration of bounded potential difference ($\varphi_j^{st} - \varphi_i^{st} \leq \gamma$) of nodes i and j, located on the opposite side from the destination t.

mapping [11, 14]. However, once embedded or mapped, both techniques have efficient distributed algorithms that use only knowledge limited to a local topology (*i.e.*, of $O(1)$ complexity) for dynamic node coordinate updates [7, 14]. Alternatively, one can use a recent geohyperbolic scheme that uses only a local topology knowledge for the entire (scale-free) network coordinates assignment [12].

TE in Hyperbolic Space. The negative curvature makes straight lines in Euclidean space curved towards the origin of a hyperbolic plane. For instance, links that are straight line segments in Euclidean space are circumference segments in the Poincaré Disk as shown in Figs. 1b and 3b. Thus, by greedy forwarding packets while minimizing their distance to the destination, nodes located closer to the origin are expected to *statistically* carry more traffic than nodes located further away from it (see Fig. 4a). We use this hyperbolic space property for our dynamic scale-free TE problem: *we approximate the maximum link utilization α being statistically proportional to some function of a distance from the hyperbolic space origin*, *i.e.*, that $\alpha \propto f(R)$, where $R \in [R_{min}, R_{max}]$ is defined as the *maximum link utilization zone* radius, and R_{min} and R_{max} are minimum and maximum node distances from the origin, respectively.

4.2. Model based on Dual Principles

Considering a network embedded in the continuous hyperbolic $\mathbb{R}^2$ plane (*e.g.*, Poincaré Disk) and assuming its $\alpha \propto f(R)$ property, we remind the reader that our goal is to create a metric whose gradient descent allows packets forwarding that (*i*) delivers them to destinations and (*ii*) preserves (in a best-effort manner) **D** feasibility. To preserve dual feasibility, we have to satisfy Eq. 5 and 6. Their satisfaction requires both global network and flow demands knowledge to compute dual variables y_{ij} and w_i^{st}. As having this knowledge is practically intractable for scale-free TE, instead of computing these dual variables directly, we can try to mimic their behavior by utilizing knowledge mined from hyperbolic metric spaces.

Let us start by closely considering dual variables y_{ij} and w_i^{st} and their properties. We start by observing unrestricted dual variables $w_i^{st} \in [-\infty, \infty]$ that corresponds to node i. To mimic this variable, we aim at finding node i metric $\varphi_i^{st} \propto -w_i^{st}$. Our first goal is to use φ_i^{st} as the metric for gradient descent during packet forwarding. Thus, we need φ_i^{st} to take its minimal value (*i.e.*, $-\infty$) at the destination t. To preserve **D** feasibility instead, let us consider positive dual variable y_{ij} that corresponds to link ij. Due to **P** and **D** *complementary slackness*, $y_{ij}(\alpha - \sum_{st \in F} f_{ij}^{st}/C_{ij}) = 0$ [34]. Thus, if the utilization of a network link ij is not equal to α, $y_{ij} = 0$, and y_{ij} is bounded by Eq. 5 ($y_{ij} \leq 1$), otherwise.

Attraction Case ($\sum_{st \in F} f_{ij}^{st}/C_{ij} < \alpha$): By substituting φ_i^{st} in Eq. 6, we need to ensure that $\varphi_t^{st} = -\infty$ and $\varphi_j^{st} \leq \varphi_i^{st}$ for the packet being forwarded from node i to node j. To this aim, we can apply the 'potential theory' and define φ_i^{st} with the following potential function [21]:

$$\varphi_i^{st} = -\frac{1}{d(i,t)} \tag{8}$$

where $d(i,t)$ is a distance function defined in Eq 7. Note how φ_i^{st} minimization gains $-\infty$ at the destination that better aligns with the unrestricted variable w_i^{st} nature, and it is in contrast to common $d(i,t)$ minimization, where $d(t,t) = 0$. The potential function in Eq. 8 can be also used in situations when we need just to deliver packets by 'attracting' them to their destinations without α minimization.

Repulsion Case ($\sum_{st \in F} f_{ij}^{st}/C_{ij} \leq \alpha$): In the general case, forcing $\varphi_j^{st} \leq \varphi_i^{st}$ can lead to suboptimalitlies, *i.e.*, $\varphi_j^{st} - \varphi_i^{st} \leq 0 < y_{ij}/C_{ij}$, when $y_{ij} > 0$. Thus, we need to add a positive counter-effect to the node potential φ_i^{st}:

$$\varphi_i^{st} = -\frac{1}{d(i,t)} + \chi_i^{st} \tag{9}$$

where χ_i^{st} is a positive potential that can 'repeal' packets from some places in the hyperbolic plane. Due to the property of hyperbolic spaces ($\alpha \propto f(R)$), we need to repeal packets from the origin $o \in \mathbb{C}$ to minimize their contribution to the maximum link utilization α. Thus, we envision χ_i^{st} to be of the following form:

$$\chi_i^{st} = \frac{Q}{d^n(i,o)} \tag{10}$$

where Q is an intensity of the positive potential field (to be derived later), and $n \in (0, \infty)$ is an attenuation order of the decaying potential χ_i^{st} — the larger n the faster diminishes the

positive counterpart contribution to φ_i^{st}. We estimate the best approximate of n empirically in Section 6.

We only need to derive Q before we are ready to engineer our REBATE protocol. To this end, we need to satisfy the following condition $\varphi_j^{st} - \varphi_i^{st} \leq y_{ij}/C_{ij}$ (see Eq. 6). To this aim, we bound with $\gamma = \max_{ij \in E}\{y_{ij}/C_{ij}\}$ the maximum possible difference of a potential function. Fig. 4b illustrates this situation when node j is located at the minimum possible distance from the origin o (*i.e.*, R_{min}), and node i is located on the radius R of the *maximum link utilization zone* defined in the previous subsection. Moreover, both nodes are located on the opposite side of the diameter w.r.t. the destination t. By substitution of Eq. 9 to the dual node constraint of **D** in Eq. 6, we have $\frac{Q}{R_{min}^n} - \frac{Q}{R^n} = \gamma + \frac{1}{R_{min}+d(o,t)} - \frac{1}{R+d(o,t)}$. Q can then be computed as following:

$$Q = \frac{R_{min}^n R^n(\gamma(R_{min} + d(o,t))(R + d(o,t)) + R - R_{min})}{(R^n - R_{min}^n)(R_{min} + d(o,t))(R + d(o,t))} \quad (11)$$

where the *maximum link utilization zone* radius can be computed as $R = \alpha(R_{max} - R_{min}) + R_{min}$ due to $\alpha \propto f(R)$. Finally, as at optimality (when $\mathbf{P} = \mathbf{D}$) we have $\alpha = \sum_{st \in F} D^{st}((w_s^{st} - w_i^{st}) + (w_i^{st} - ... - w_j^{st}) + (w_j^{st} - w_t^{st}))$, we can approximate $\gamma \approx \alpha / \min_{st \in F}\{D_{st}\}$, where $\min_{st \in F}\{D_{st}\}$ can be assumed a *unitary* flow due to the lack of the flow demand-awareness. As a result, when $\alpha \to 0, \gamma \to 0$ and $R \to 0$, and hence $Q \to 0$.

4.3. Examples of 'Attraction' and 'Repulsion' Cases

To demonstrate the difference between *attraction* and *repulsion* cases, let us consider our greedy embedding example of a simple tree-like network in Fig. 3. When we greedy forward a packet using a potential in the *attraction* case, we try only to deliver it to the destination, *i.e.*, preserve **P** feasibility in a best-effort manner. As a result, the packet is routed along the shortest path in terms of hyperbolic distance (see Eq. 7) and traverses nodes located in the maximum link utilization zone (closer to the root node) as shown in Fig 5a. When we greedy forward this packet using a potential in the *repulsion* case instead, we now try to also preserve **D** feasibility in a best-effort manner by repealing the packet away from the origin as defined by packet's positive potential counterpart (see Eq. 10). The resulting packet's route avoids nodes in the (current) maximum link utilization zone by traversing nodes located further away from the root as shown in Fig 5b.

In the next section, we use our REBATE model to propose a practical protocol that uses both potentials from Eq. 8 and 9, referred to as 'attractive' and 'repulsive' based on their performance, respectively. We remark that the former aims only at delivering packets to preserve **P** feasibility (i), whereas the latter aims at delivering packets (i) while also preserving **D** feasibility (ii) in a best-effort manner.

5. REBATE: Protocol

In this section, we first outline our REBATE protocol algorithm and discuss its practical implications. We then describe its packet header.

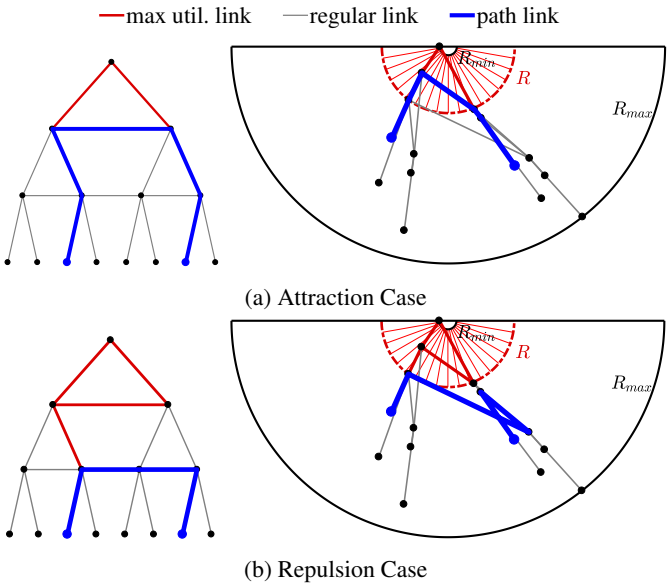

Figure 5: Illustration of a packet forwarding within a greedy embedded network example (see Fig. 3) in both attraction (a) and repulsion (b) cases: by greedy forwarding packets in the *attraction* case, we aim only to deliver them to their destinations, and route packets among the shortest (hyperbolic distance) paths; by greedy forwarding packets in the *repulsion* case, we now try to also preserve **D** feasibility (in a best-effort manner) by repealing packets away from the origin to avoid nodes in the maximum link utilization zone.

5.1. Algorithm

Note how gradient descent of both 'attractive' and 'repulsive' potentials (see Eq. 8 and 9) is subject to the *local minimum problem* similar to the gradient descent of $d(i,t)$ distance function [14]. As a result, packets may not be delivered to the destination. To maximize a chance of packets delivery in dynamic scale-free networks, we propose a solution that alternates packets forwarding using both 'repulsive' and 'attractive' potentials. We remark that greedy forwarding using the 'repulsive' potential tries to deliver packets (i) while preserving **D** feasibility (ii), whereas greedy forwarding based on the 'attractive' potential aims to achieve only (i). Finally, we also add a greedy forwarding in 'pressure' mode, as proposed in [14], to recover packets that are simultaneously stuck in local minima of both 'repulsive' and 'attractive' modes.

Algorithm 1 outlines the REBATE greedy forwarding logic: upon receiving a packet P^{st} going from source s to destination t to be forwarded at node i, i first retrieves the current maximum link utilization α (line 2). We later discuss a simple technique that can avoid the need of using a global α knowledge. Once α is retrieved, node i checks if P^{st} should proceed further in 'repulsive', 'attractive' or 'pressure' mode. To do so, node i first checks that the best known potential in 'repulsive' mode attached to packet $P^{st}\_\varphi_{rep}$ is greater than φ_i^{st}, potential of node i (line 4). This step is important to ensure that packets progress in 'repulsive' mode towards the destination without the possibility of returning to previously encountered local minima. Thus, if $P^{st}\_\varphi_{rep} > \varphi_i^{st}$, i attaches its 'repulsive' potential to the packet header (line 5) and computes neighbors $j \in Nbrs$ potential φ_j^{st} (line 6) via Eq. 9.

If no neighbor j has a potential $\varphi_j^{st} < \varphi_i^{st}$, i switches to the 'attractive' mode, where it computes its own (φ_i^{st}) and all neighbors $j \in Nbrs$'s potential (φ_j^{st}) using Eq. 8 (line 12). If node i in 'attractive' mode is unavailable to find next hop j, i also saves the last best known potential of the 'attractive' mode to the φ^{st}_attr of the packet header (line 13). This step is also needed to ensure progress in 'attractive' mode. When both 'repulsive' and 'attractive' forwarding modes fail to find the next hop j, REBATE switches to 'pressure' mode (line 20). The key idea behind recovery in 'pressure' mode is to forward packet to j neighbor with minimal potential computed via Eq. 9 among the least visited neighbors (line 23).

Algorithm 1: REBATE

```
   /* Upon receiving a packet P^st at node i                    */
 1 if i ≠ dst then
 2     retrieve α
 3     next ← NIL
 4     if P^st_φ_rep > φ_i^st (using Eq. 9) then
           /* repulsive mode                                     */
 5         P_φ_rep ← φ_i^st (using Eq. 9)
 6         j ← argmin_{j∈Nbrs(i)} φ_j^st (using Eq. 9)
 7         if φ_j^st < φ_i^st then
 8             next ← j
 9             forward(P^st, next)
10         end
11     end
12     if next == NIL and P_φ_attr > φ_i^st (using Eq. 8) then
           /* attractive mode                                    */
13         P_φ_attr ← φ_i^st (using Eq. 8)
14         j ← argmin_{j∈Nbrs(i)} φ_j^st (using Eq. 8)
15         if φ_j^st < φ_i^st then
16             next ← j
17             forward(P^st, next)
18         end
19     end
20     if next == NIL then
           /* pressure mode                                      */
21         visits_min ← min_{j∈Nbrs(i)} P^st_visits(j)
22         Candidates ← {j ∈ Nbrs(i) and P^st_visits(j) == visits_min}
23         j ← argmin_{j∈Candidates} φ_j^st (using Eq. 9)
24         P^st_visits(j) ← P^st_visits(j) + 1
25         next ← j
26         forward(P, next)
27     end
28 else
29     terminate
30 end
```

5.2. Global vs Local α Knowledge

On one hand, we need the maximum link utilization α (see Alg. 1, line 2) to calculate the intensity of the repulsion field (see Eq. 10 and 11). On the other hand, we need to repeal only those flows that contributes to the maximum link utilization based on **D** node constraints (see Eq. 6), *i.e.*, flows that traverse edges with $y_{ij} > 0$. Thus, *we hypothesis that repealing all flows based on the 'global' maximum link utilization α can lead to a worse satisfaction of **D** feasibility.* Instead, we aim to calculate the intensity of the repulsion field based on the *local* α *awareness* of the flow. Intuitively, if a flow contributes to the actual (*i.e.*, global) α, it will be aware of it making its repulsion field stronger, or weaker otherwise.

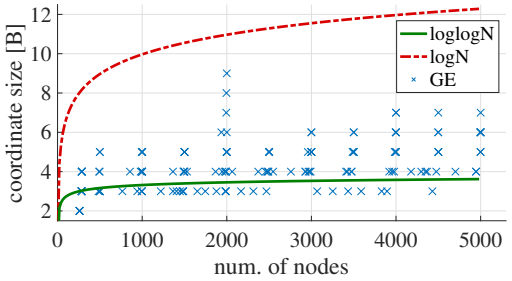

Figure 6: Minimal coordinate sizes for GE of both real US Tier-1 and generated based on Waxman model networks.

We propose a simple scheme on how to use a local knowledge of α stored on packet P^{st} which is forwarded from s to t. In particular, we store two additional values α_cur and α_next. During P^{st} forwarding each node i uses $\alpha = \alpha\_cur$, and updates α_next if one of its adjacent links has higher utilization than α_next. The idea is to benefit from common two-way communications (*e.g.*, transport protocols like TCP). At the beginning, node s decides on α_cur based on the last received information from t (*e.g.*, upon receiving an ACK message). To this end, s computes α_cur as follows:

$$\alpha\_cur = \begin{cases} \alpha\_next, & \text{if } \alpha\_next \geq \alpha\_cur \\ \lambda\alpha\_next + (1-\lambda)\alpha\_cur, & \text{otherwise} \end{cases} \tag{12}$$

where $\lambda \in [0, 1]$ represents a network "cold-down" property, *i.e.*, the larger is λ, the faster the maximum link utilization of the network diminishes.

In the evaluation section, we empirically validate our intuition on a local α knowledge, *i.e.*, we show how such knowledge (see Eq. 12) is sufficient for our REBATE.

5.3. Complexity Analysis

In the worst case scenario, Algorithm 1 proceeds exactly once in all three modes: 'attractive', 'repulsive' and 'pressure' modes. The asymptotic computational complexity of each mode is $O(k)$, where k is an average node degree. This is because each node estimates its own potential φ_i^{st} and φ_j^{st} potential of all of its neighbors to make a forwarding decision. Thus, Algorithm 1 has the following complexity:

$$O(3 \cdot k) = O(k). \tag{13}$$

Note that for networks that features a *scale-free* nature, k on average is of constant size $O^*(1)$. Hence, REBATE has amortized $O^*(1)$ space-time complexity based on Equation 13 similar to common hyperbolic geometric routing solutions [7]. At the same time, for the arbitrary graphs k (and hence the REBATE space-time complexity) is $O(N)$ inthe worst case.

Aside from space complexity due to number of entries stored on routers, there is another potential problem related to the size of coordinates. It is obvious that with the increase of the network size the precision of coordinates $\{z_i \in \mathbb{C} : |z_i| \leq 1\}$

should increase too to make sure all coordinates are unique. Figure 6 illustrates such coordinate size dependency on the network size when both real US Tier-1 and generated based on Waxman model [35] networks are greedy embedded. GE can be considered as less efficient hyperbolic coordinates assignment technique in terms of required coordinates precision. This is due to the fact that chosen GE method [14] uses only the half Poincare plane and rapidly pushes node coordinates closer to $|z_i| = 1$ boundary (*e.g.*, see Figure 3b). However, we can see how even in this case node coordinates are in between $O(logN)$ and $O(loglogN)$ size. As designing the best hyperbolic coordinate assignment technique is out of scope for this paper, *we just fixed 6 bytes for each real and imaginary parts of* z_i to allow unique hyperbolic coordinate assignment for networks with $\approx 10^6$ to 10^{20} nodes depending on the network topology. Note that when network node address/coordinates are fixed (as in conventional routing or TE), REBATE space-time complexity doesn't depend on it.

5.4. Protocol

Having both REBATE model and its algorithm discussed, we now describe REBATE protocol and its header. Our protocol is supposed to operate on Layer 2.5 w.r.t. OSI model — common for non-IP routing solutions [36, 37]. Thus, routers check if REBATE can be used first, and if not, they forward packets using standard IP techniques. Each node i has to know its own coordinate (z_i) as well as all its neighbors' coordinates. In addition to that, our REBATE algorithm needs knowledge of both the minimum (R_{min}) and the maximum (R_{max}) hyperbolic radiuses of the embedded *core network*. The above information is static and requires no communication costs to update. It can be stored on each node during its embedding process, *e.g.*, when node retrieves its coordinate from the default router [7]. All other required information needs to be stored in the REBATE packet header itself.

Firstly, our REBATE header needs to store source and destination coordinates. We remark that each node i has a hyperbolic coordinate represented by a complex number $z_i \in \mathbb{C} : |z_i| \leq 1$. To store this information we can use 6 bytes to represent the real ($real(z_i)$) and imagery ($imag(z_i)$) parts of the complex number z_i as discussed in Section 5.3. Hence, we need 12 bytes to store the source (z_{src}) and 12 bytes to store the destination (z_{dst}) coordinates. Note that such information is common for all geometric routing protocols.

Secondly, we need to store REBATE-specific information: we need 4 bytes (regular float precision) to store both (known) potentials in *attractive* (φ_{attr}) and *repulsive* (φ_{rep}) modes computed via Eq. 8 and 9, respectively; we also need 2 bytes (half precision) to store the current maximum link utilization (α_{cur}) and the next proposed maximum link utilization (α_{next}) used to retrieve α (see Eq. 12).

Finally, we need to store GPGF-specific information to track number of the unique node visits during a packet recovery from local minima in the *pressure* mode [14]. Each node uniqueness can be identified by its coordinate z_i (16 byte information). Alternatively, one can use IPv6 (128 bit) or MAC (48 bit) addresses to this aim. However, all such information can be expensive to store within a dynamic part of the header [31]. To avoid potential intractabilities caused by a large size of the dynamic header part, we propose a simple hash function for the node i coordinate z_i:

$$hash\,(i) = 2^{\frac{N}{2}} \cdot round\left((2^{\frac{N}{2}} - 1) \cdot \frac{d(o,i) - R_{min}}{R_{max} - R_{min}}\right) + round\left((2^{\frac{N}{2}} - 1) \cdot \frac{\phi_i}{2\pi}\right) \tag{14}$$

where N is a number of bits of the hash key, and ϕ_i is an angular coordinate of z_i complex number computed as:

$$\phi_i = \begin{cases} tan^{-1}(\frac{imag(z_i)}{real(z_i)}), & real(z_i) \geq 0 \\ tan^{-1}(\frac{imag(z_i)}{real(z_i)}) + \pi, & otherwise \end{cases} \tag{15}$$

For example, if we hash node i with 8-bit keys, Eq. 14 produces integer numbers from 0 to 255. Similarly, Eq. 14 produces integer numbers from 0 to 65535 for 16-bit keys.

The resulting REBATE packet header is shown in Table 4, where n is a number of unique nodes visited in the *pressure* mode, and N is a number of bits of the hash key.

Table 4: REBATE Packet Header.

src z **coord.**	**dst** z **coord.**	**repulsive potential**	**attractive potential**	α **current**	α **next**	**hash keys of nodes**	**# of visits**
$2 \cdot 6$ B	$2 \cdot 6$ B	4 B	4 B	2 B	2 B	$n \cdot N/2^3$ B	n B

We conclude that, if REBATE packet does not enter the *pressure* mode (does not face a local minimum), its static packet header size is equal to $(12+4+2) \cdot 2 = 36$ bytes. In the next, we show how the dynamic REBATE header part does not introduce intractabilities under practical TTL and when hash functions are used. We also show how our hash function performance is sufficient for 16-bit keys.

6. Performance Evaluation

In this section, we establish the practicality of our REBATE approach by evaluating its performance using traces from the US Tier-1 core network [38, 39]. [4]

Simulation Settings. Our Java-based simulations are run on an Ubuntu 16.04 OS GNU/Linux *x*86_64 machine with an *Intel Core i5* 2.4 GHz CPU and 8GB RAM. We use both Internet Topology Zoo [38] and Atlas [39] topology databases to recreate the US Tier-1 providers' network as shown in Fig. 1a. We assume that this network has 10 Gbps capacity links. We also assume that each US Tier-1 (*i.e.*, core) network node has attached from 10 to 1000 IoT devices that generate from 100 to 1000 concurrent flows. Each flow has a demand ranging from 10 Mbps to 1 Gbps and following a Pareto distribution — 20% of the flows have demand closer to 1 Gbps whereas the rest 80% have demand near 10 Mbps. This flow demand distribution

[4] The source code of our trace-driven simulations is publicly available at `https://github.com/duman190/rebate`.

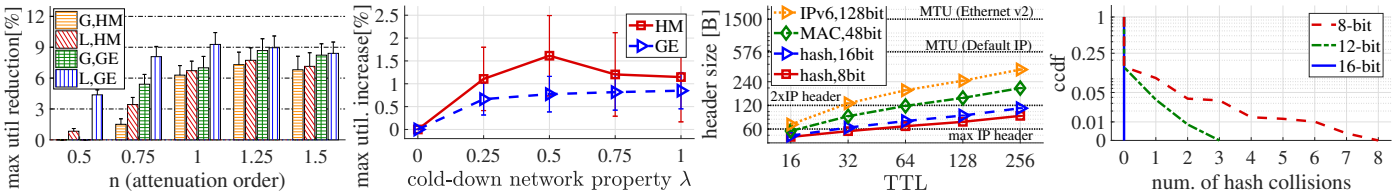

Figure 7: (a) REBATE percentage reduction of the maximum link utilization α for 100 concurrent flows and different attenuation orders n w.r.t. its performance under $n = 0.5$ and the global (G) α knowledge: REBATE performance improves under a local (L) α knowledge, as less unnecessary packets are repealed from the origin. (b) Local REBATE performance for 100 concurrent flows: REBATE is not sensitive to either greedy embedded (GE) or hyperbolically mapped (HM) network cold-down property λ, but properly adjusted λ in Eq. 12 can slightly improve its performance.(c) Dynamic header size of REBATE for packets routed in hyperbolically mapped network (it is static size of 36 bytes when the pressure mode is not used, e.g., within greedy embedded network): using 16-bit hash function in Eq. 14 allows REBATE packet header to not exceed doubled maximum IP header size in 95% cases. (d) The hash function (see Eq. 14) complementary CDF (*i.e.*, P(X≥x) = y) of collisions w.r.t. its hash key size in bits under TTL=256.

reflects a common traffic aggregation from end-devices into a single flow also known as a flow group or a tunnel [16, 17]. Under the maximum (practical) packet's TTL of 256 hops, we attempt to deliver IoT traffic of these flows among their associated $src - dst$ pairs.

Our main goal is to evaluate performance of our REBATE when balancing the ***core network*** *load by minimizing the maximum link utilization under stress. We remark that the main challenge for the dynamic scale-free TE is to do such optimization without both network topology knowledge and flow demand awareness.* All our results show 95% confidence interval over 100 trials, and our randomness lies in both the source-destination flow pairs and their demands.

Comparison Methods and Metrics. To empirically evaluate which potential field best approximates dual node prices, we tested the performance of our REBATE under different potential field attenuation orders n. We then leverage our finding ($n = 1$ when the network is Greedy Embedded (GE) and $n = 1.25$ when it is Hyperbolically Mapped (HM) in our other experiments. We then compare our REBATE scale-free TE with the optimal demand-aware TE (by solving **P** with CPLEX [40]) and the traffic steering that utilizes common scale-free routing protocol — GPGF [14]. Note that while comparing with GPGF-based traffic steering, we use both GE and HM hyperbolic coordinates assignment techniques. Our goal is to show that REBATE statistically outperforms GPGF-based traffic steering in the traffic optimization. Furthermore, we assess REBATE optimality gaps, and we estimate the impact of GE and HM on our REBATE performance. The related solutions are compared across the following metrics: the maximum link utilization (the lower the better), the maximum link utilization increase (the lower the better) and its reduction (the higher the better) measured in %. We also compare complimentary CDFs (CCDFs) of algorithms path hop count, their satisfied flows ratio as well as the link utilization CDF. Finally, we compare CCDFs of our proposed hash function (see Eq. 14) using different hash key sizes.

Results. Our evaluation results fall under three salient findings: (*i.a*) *The local REBATE knowledge of the maximum link utilization is "enough"*. (*i.b*) *Dynamic REBATE packet header size does not introduce intractabilities under practical TTL and when hash functions are used*. (*ii*) *REBATE can decrease the maximum link utilization by 25% when compared with a traffic steering based on a scale-free routing*. (*iii*) *We should hyperbolically map to better balance the network load under high workloads/network dynamics and greedy embed to minimize the maximum utilization otherwise*. (*iv*) *REBATE is suboptimal to both demands-aware and oblivious TE*.

(i.a) Local REBATE knowledge of the maximum link utilization is enough. Note how Figure 7a confirms our hypothesis that our REBATE approach is not affected by a lack of global knowledge on the current maximum link utilization in the network (see Section 5.2). As expected, we can see how in all cases REBATE with a local α knowledge slightly outperforms its variant with a global α awareness. Moreover, REBATE achieves the minimum of the maximum link utilization when attenuation order $n = 1$ while operating with its either global or local knowledge in the greedy embedded network. This attenuation order matches the one generated by a potential field created by the destination. At the same time, for the hyperbolically mapped network, REBATE shows the minimum of the maximum link utilization when attenuation order is $n = 1.25$. This result is due to the fact that HM cannot guarantee packets delivery by the greedy forwarding and needs a recovery scheme which increases path stretches in the network. Thus, by slightly increasing attenuation order n, we can mitigate an additional path stretch due to REBATE packets repulsion. We use $n = 1$ and $n = 1.25$ for GE and HM for the rest of simulations, respectively.

An additional evaluation has been conducted to estimate the local REBATE sensitivity to the network cold-down property λ accuracy used to retrieve α (see Eq. 12). Fig. 7b shows how during network 'peak hours' (*i.e.*, $\lambda = 0$), the maximum link utilization achieved by our (local) REBATE slightly increases (*e.g.*, up to 1.5%) if λ is incorrectly specified. We conclude that properly adjusted λ can slightly improve our REBATE performance and has an insignificant impact. Note that infrastructure providers can tune REBATE parameters to improve its performance based on the infrastructure topology, utilized hardware, and specific traffic demands, e.g., orchestrating traffic during the day hours can be based on different parameters w.r.t. the traffic orchestration at night.

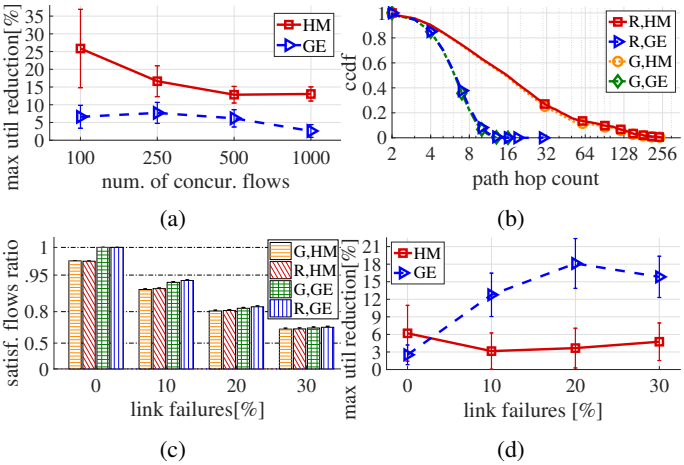

Figure 8: REBATE (R) reduction percentage of the maximum link utilization (a,d) compared to GPGF-based traffic steering (G); corresponding Complement CDFs of 1000 concurrent flows associated path hop counts (b) satisfied flow ratios under different link failure rates (c) using both hyperbolic mapping (HM) and greedy embedding (GE): REBATE can better optimize traffic and has the same quality of packets routing as GPGF.

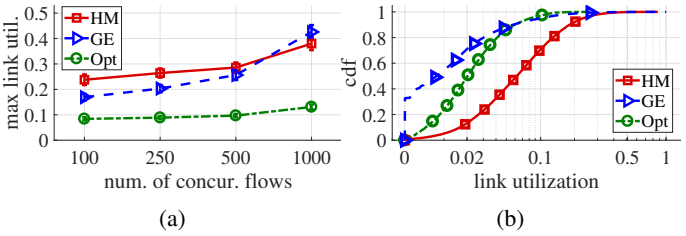

Figure 9: Maximum link utilization (a) and link utilization CDFs to host 1000 flows (b) results of REBATE (using both hyperbolic mapping (HM) and greedy embedding (GE)) compared to the optimal demand-aware TE (Opt): REBATE is subject to large optimality gaps in practice; it is recommended to use HM for high network workloads to better balance it, and GE otherwise.

(i.b) Dynamic REBATE packet header size does not introduce intractabilities under practical TTL and when hash functions are used. Due to use of GPGF *pressure* mode that tracks node visits to recover packets from local minima, our REBATE protocol header has a dynamic size. From Fig. 7c we can see how our REBATE header size does not introduce large overhead for packets routed in a hyperbolically mapped network under practical TTL. For example, under TTL= 256 and when 16-bit hash function is used (see Eq. 14), our rebate packet header size is no more than twice of the maximum IP header size (*i.e.*, $<$ 120 bytes) in 95%. This is by an order of magnitude less than the common MTU size. We remark that we use hash functions to identify node uniqueness while tracking their visits. We can further reduce our REBATE packet header size at expense of a small number of hash collisions as shown in Fig 7d. At the same time, identifying node uniqueness via use of MAC or IPv6 addresses can lead to large packet header sizes of the REBATE protocol. Thus, we use 16-bit hash function in Eq. 14 within our REBATE protocol for the rest of simulations.

(ii) REBATE can decrease the maximum link utilization by 25% in comparison with GPGF traffic steering. Fig. 8a shows how our REBATE statistically outperforms a traffic steering with the common GPGF geometric routing (designed for dynamic scale-free networks) when the core network is embedded in the hyperbolic space of the negative curvature [14, 11]. Particularly, REBATE can decrease the maximum link utilization by 10% or 25% when GE or HM coordinate assignment techniques are used, respectively. This is due to the fact that REBATE repeals excess traffic away from the maximum link utilization zone. Moreover, REBATE can further reduce the maximum link utilization up to 20% w.r.t. GPGF-based traffic steering under greedy embedded network dynamics. For example, observe a case of 20% link failures for 1000 concurrent flows in Figure 8d; this is due to the fact that tree link failures revoke the packet delivery guarantees of GE, repealing traffic away from the origin reduces packets routing over such links. As a result, we do not observe similar improvements for hyperbolically mapped networks, as HM does not use spanning trees for coordinate assignments.

We can also see that the reduction in the maximum link utilization of REBATE is not caused by its worse flow demands satisfaction (see Fig. 8c) as well as by the path stretch increase that affects packet latency (see Fig. 8b) compared to the GPGF. Moreover, flow demands satisfaction depends more on a coordinate assignment technique — the larger are paths stretches in the network due to poor greedy forwarding performance the lower is the scale-free routing packet delivery ratio under practical TTL.

(iii) Hyperbolically map to better balance the network load under high workloads/dynamics, greedy embed to minimize the maximum utilization otherwise. When observing Fig. 8c and 9a, we found that by greedy embedding a network in hyperbolic spaces and under low network dynamics ($\lesssim$ 10% of link failures), REBATE better copes with flow demand satisfaction and better minimizes the maximum link utilization. This is due to a lower path stretch of the greedy forwarding (see Figure 8b). However, the higher the workload or network dynamics the higher is the link utilization of the underlying spanning tree used by GE. As a result, at high workload/dynamics levels (*e.g.*, at 1000 concurrent flows or for $\geq$ 10% of failures), REBATE balances the network load better when this network is hyperbolically mapped (Fig. 9b).

(iv) REBATE is suboptimal to both demands-aware and oblivious TE. The lack of a flow demand awareness in oblivious TE introduces a large *theoretical* optimality gap [15], but does not prevent it from achieving a low optimality gap in practice. For instance, the recent semi-oblivious TE approach reaches optimality in 75-80% practical cases [23]. In contrast with oblivious TE, our REBATE is subject to large *practical* optimality gaps compared to demands-aware TE, even though it significantly improves TE performance in comparison with GPGF traffic steering. For example, in the case of 500 concurrent flows, our REBATE for hyperbolically mapped network achieves a maximum link utilization of $\approx$ 0.3 whereas the optimal is $\approx$ 0.1 (see Figs. 9a and 9b).

This result is due to the fact that *our REBATE does not use neither global network knowledge nor flow demand awareness*,

and motivates the need for more research in both TE and scale-free networking areas to further improve dynamic scale-free TE performance which can be essential for the next-generation Internet augmented with IoT.

7. Conclusion

In this paper, we have proposed the first to our knowledge Traffic Engineering protocol that can be used within dynamic scale-free networks. We hypothesize that our protocol can be particularly crucial for next-generation Internet segments augmented with edge [6, 13] and IoT devices [1] to optimize traffic in their hardly controllable, dynamic and large core subnetworks. We called our solution REpulsive-BAsed Traffic Engineering (REBATE); REBATE operates only with a local knowledge of the topology, *i.e.*, it needs only $O(1)$ space complexity to optimize traffic. To do so, we leveraged dual principles of the optimal demand-aware TE as well as fundamental properties of hyperbolic metric spaces. Using real US Tier-1 network traces in our simulator, we have found that REBATE can reduce the maximum link utilization of the network by 25% in comparison with a traffic steering technique that uses commonly adopted GPGF geometric routing [7, 14]. However, we found that our REBATE can be still subject to large optimality gaps in practice when compared to both demand-aware and oblivious TE. Thus, we conclude that REBATE is the first step towards an efficient TE protocol that can serve needs of dynamic scale-free networks.

As an open question we leave the use of oblivious TE principles while operating with a local (topology) knowledge of $O^*(1)$ amortized space-time complexity. Another promising direction can be designing a hyperbolic coordinates assignment technique that is better congruent with the traffic optimization process itself — similarly to how existing greedy embedding schemes are congruent with greedy packet forwarding [7, 33]. We remark that the current version of the protocol has other major limitations aside from the lack of optimality guarantees. For instance, REBATE currently supports only MLU minimization and orchestrates traffic on a per-packet basis which itself can be subject to high CPU utilization of switches and packets re-ordering. Thus, other future interesting avenues to explore are the following: supporting other objectives with better optimization performance; using different address schemes (e.g., one that yield simple bit-wise operations as opposite to math calculations over complex numbers); and developing complementary mechanisms to cope with packets re-ordering (e.g., in addition to native TCP mechanisms).

Acknowledgments

This material is based upon work supported by the National Science Foundation (NSF) under Award Number CNS1647182, Coulter Foundation Translational Partnership Program and Russian Foundation for Basic Research 13-07-00381a. Any opinions, findings or conclusions expressed in this publication are those of the author(s) and do not necessarily reflect the views of the funding agencies.

References

[1] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[2] Taewoo Nam and Theresa A Pardo. Conceptualizing smart city with dimensions of technology, people, and institutions. In *Proceedings of conference on digital government innovation in challenging times*, pages 282–291. ACM, 2011.

[3] Qing Li, Dan Wang, Mingwei Xu, and Jiahai Yang. On the scalability of router forwarding tables: Nexthop-selectable fib aggregation. In *Proceedings of INFOCOM*, pages 321–325. IEEE, 2011.

[4] Garegin Grigoryan, Yaoqing Liu, Michael Leczinsky, and Jun Li. Veritable: Fast equivalence verification of multiple large forwarding tables. In *Proceedings of IEEE INFOCOM*, pages 621–629. IEEE, 2018.

[5] Dmitri Krioukov, Kevin Fall, Arthur Brady, et al. On compact routing for the internet. *ACM SIGCOMM Computer Communication Review*, 37(3):41–52, 2007.

[6] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of workshop on mobile cloud computing*, pages 13–16. ACM, 2012.

[7] Fragkiskos Papadopoulos, Dmitri Krioukov, Marián Boguñá, and Amin Vahdat. Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces. In *2010 Proceedings IEEE INFOCOM*, pages 1–9. IEEE, 2010.

[8] Albert-László Barabási. Scale-free networks: a decade and beyond. *science*, 325(5939):412–413, 2009.

[9] Walter Willinger, David Alderson, and John C Doyle. Mathematics and the internet: A source of enormous confusion and great potential. *Notices of the American Mathematical Society*, 56(5):586–599, 2009.

[10] Anna D Broido and Aaron Clauset. Scale-free networks are rare. *Nature communications*, 10(1):1017, 2019.

[11] Fragkiskos Papadopoulos, Constantinos Psomas, and Dmitri Krioukov. Network mapping by replaying hyperbolic growth. *IEEE/ACM Transactions on Networking*, 23(1):198–211, 2015.

[12] Ivan Voitalov, Rodrigo Aldecoa, Lan Wang, and Dmitri Krioukov. Geo-hyperbolic routing and addressing schemes. *ACM SIGCOMM Computer Communication Review*, 47(3):11–18, 2017.

[13] Niroshinie Fernando, Seng W Loke, and Wenny Rahayu. Mobile cloud computing: A survey. *Future generation computer systems*, 29(1):84–106, 2013.

[14] Andrej Cvetkovski and Mark Crovella. Hyperbolic embedding and routing for dynamic graphs. In *Proceedings of INFOCOM*, pages 1647–1655. IEEE, 2009.

[15] Marco Chiesa, Gábor Rétvári, and Michael Schapira. Lying your way to better traffic engineering. In *Proceedings of conference on emerging Networking EXperiments and Technologies*, pages 391–398. ACM, 2016.

[16] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, et al. B4: Experience with a globally-deployed software defined wan. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 3–14. ACM, 2013.

[17] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. Achieving high utilization with software-driven wan. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 15–26. ACM, 2013.

[18] Frederick Prinz, Michael Schoeffler, Armin Lechler, and Alexander Verl. Dynamic real-time orchestration of i4. 0 components based on time-sensitive networking. *Procedia CIRP*, 72:910–915, 2018.

[19] Polona Štefanič, Matej Cigale, Andrew C Jones, Louise Knight, Ian Taylor, Cristiana Istrate, George Suciu, Alexandre Ulisses, Vlado Stankovski, Salman Taherizadeh, et al. Switch workbench: A novel approach for the development and deployment of time-critical microservice-based cloud-native applications. *Future Generation Computer Systems*, 99:197–212, 2019.

[20] Hossam Farag, Mikael Gidlund, and Patrik Österberg. A delay-bounded mac protocol for mission-and time-critical applications in industrial wireless sensor networks. *IEEE Sensors Journal*, 18(6):2607–2616, 2018.

[21] Oliver Dimon Kellogg. *Foundations of potential theory*, volume 31. Springer Science & Business Media, 2012.

[22] Eva Tardos. A strongly polynomial algorithm to solve combinatorial linear programs. *Operations Research*, 34(2):250–256, 1986.

[23] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Chiun Lin Lim, and Robert Soulé. Semi-oblivious traffic engineering: The road not taken. In *USENIX NSDI*, 2018.

[24] Mathieu Leconte, Apostolos Destounis, and Georgios Paschos. Traffic

engineering with precomputed pathbooks. In *Proceedings of IEEE INFOCOM*, pages 234–242. IEEE, 2018.

[25] Harald Racke. Minimizing congestion in general networks. In *Proceedings of Symposium on Foundations of Computer Science*, pages 43–52. IEEE, 2002.

[26] Harvinder Singh. *Compass routing on geometric graphs*. University of Ottawa (Canada), 1999.

[27] Ben Leong, Barbara Liskov, and Robert Morris. Geographic routing without planarization. In *USENIX NSDI*, volume 6, page 25, 2006.

[28] Jiangwei Zhou, Yu Chen, Ben Leong, and Pratibha Sundar Sundaramoorthy. Practical 3d geographic routing for wireless sensor networks. In *Proceedings of Conference on Embedded Networked Sensor Systems*, pages 337–350. ACM, 2010.

[29] Simon S Lam and Chen Qian. Geographic routing in d-dimensional spaces with guaranteed delivery and low stretch. *IEEE/ACM Transactions on Networking*, 21(2):663–677, 2013.

[30] Michal Król, Eryk Schiller, Franck Rousseau, and Andrzej Duda. Weave: Efficient geographical routing in large-scale networks. In *EWSN*, pages 89–100, 2016.

[31] Sahel Sahhaf, Wouter Tavernier, Didier Colle, Mario Pickavet, and Piet Demeester. Experimental validation of resilient tree-based greedy geometric routing. *Computer Networks*, 82:156–171, 2015.

[32] Brad Karp and Hsiang-Tsung Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *Proceedings of MobiCom*, pages 243–254. ACM, 2000.

[33] Robert Kleinberg. Geographic routing using hyperbolic space. In *Proceedings of INFOCOM*, pages 1902–1909. IEEE, 2007.

[34] Mokhtar S Bazaraa, John J Jarvis, and Hanif D Sherali. *Linear programming and network flows*. John Wiley & Sons, 2011.

[35] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. Brite: An approach to universal topology generation. In *MASCOTS 2001, Proceedings Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 346–353. IEEE, 2001.

[36] Braulio Dumba, Guobao Sun, Hesham Mekky, and Zhi-Li Zhang. Experience in implementing & deploying a non-ip routing protocol viro in geni. In *Proceedings of ICNP*, pages 533–539. IEEE, 2014.

[37] Deep Medhi and Karthik Ramasamy. *Network routing: algorithms, protocols, and architectures*. Morgan Kaufmann, 2017.

[38] Simon Knight, Hung X Nguyen, Nick Falkner, Rhys Bowden, and Matthew Roughan. The internet topology zoo. *Journal on Selected Areas in Communications*, 29(9):1765–1775, 2011.

[39] Ramakrishnan Durairajan, Subhadip Ghosh, Xin Tang, Paul Barford, and Brian Eriksson. Internet atlas: a geographic database of the internet. In *Proceedings of workshop on HotPlanet*, pages 15–20. ACM, 2013.

[40] Ibm cplex solver. `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html`. Accessed: July, 2018.