Toward Optimal Software-Defined Interdomain Routing

Qiao Xiang*‡, Jingxuan Zhang‡*, Kai Gao°,
Yeon-sup Lim◇, Franck Le◇, Geng Li‡*, Y. Richard Yang†‡,
*Tongji University, ‡Yale University, †Peng Cheng Laboratory,
°Sichuan University, ◇IBM T.J. Watson Research Center,

***Abstract*—End-to-end route control spanning a set of networks can provide opportunities to both end users to optimize interdomain control and network service providers to increase business offering. BGP, the de facto interdomain routing protocol, provides no programmable control. Recent proposals for interdomain control, such as MIRO, ARROW and SDX, provide more mechanisms and interfaces, but they are only either point or incremental solutions. In this paper, we provide the first, systematic formulation of the *software-defined internetworking (SDI)* model, in which a network exposes a programmable interface to allow clients to define the interdomain routes of the network, just as a traditional SDN switch exposes Openflow or another programmable interface to allow clients to define its next hops, extending SDN from intra-domain control to generic interdomain control. Different from intradomain SDN, which allows complete client control, SDI should also maximize network autonomy, such as by allowing a network to maintain the control of its interdomain export policies, to avoid fundamental violations such as valley routing. We define the *optimal end-to-end SDI routing problem* and conduct rigorous analysis to show that the problem is NP-hard. We develop a blackbox optimization algorithm, which leverages Bayesian optimization theory and important properties of interdomain routing algebra, to sample end-to-end routes sequentially and find a near-optimal policy-compliant end-to-end route with a small number of sample routes. We implement a prototype of our optimization algorithm and validate its effectiveness via extensive experiments using real interdomain network topology. Results show that in an interdomain network with over 60000 ASes and over 320000 AS-level links, in 80% experiment cases, the blackbox optimization algorithm can find a near-optimal policy-compliant end-to-end route by sampling less than 33 routes.**

I. Introduction

Although flexible, end-to-end route control may provide substantial benefits to both networks and end users (*e.g.*, conduct traffic engineering, or prevent DDoS attacks), it is extremely complex and difficult to achieve, if not impossible, in the current Internet. This is due to the design of the Border Gateway Protocol (BGP) [1], the *de facto* interdomain routing protocol. Specifically, with BGP, each autonomous systems (AS) can make and execute its own policy to select routes on a destination prefix basis, and export the selected routes, in terms of path vectors (*i.e.*, AS path), to its neighbor ASes. Although this design is simple and widely adopted, it provides very limited mechanisms for network operators and end users to achieve flexible, end-to-end route control. AS paths that span from a source AS to a destination AS, and that the path does not traverse a certain AS.

To appreciate the limitation of BGP, consider AS S shown in Fig. 1, who has a strong performance requirement to select a shortest AS path from it to a destination prefix p located at AS T. The ranking used by AS A and D to select routes is also shown in the figure. Assume that the export policies of each AS is to always announce the selected route to its neighbor ASes. When the network converges, AS A selects the AS-path $[A, B, D, F, H, T]$ and announces to AS S. As such, the only AS-path to T learnt by S is $[S, A, B, D, F, H, T]$. One can observe that there are indeed AS paths shorter than $[S, A, B, D, F, H, T]$ in the network, such as $[S, A, B, D, E, T]$. To use such a path, AS D must select the route $[D, E, T]$. Unfortunately, in BGP, AS S cannot control D to select $[D, E, T]$. As such, S can only use $[S, A, B, D, F, H, T]$, not satisfying its requirement. At the same time, AS D misses a potential business opportunity.

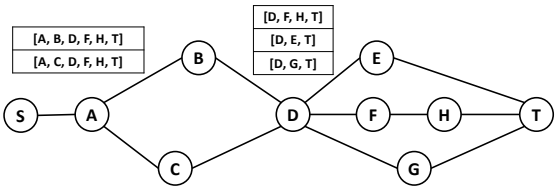


Fig. 1: A motivating example to illustrate the limitations of BGP and other interdomain routing systems for supporting flexible, end-to-end route control.

To provide more flexible, end-to-end route control, several interdomain routing systems have been designed and deployed [2]–[10]. For example, in MIRO [9] and ARROW [10], an AS allows a client (*i.e.*, an operator of another AS or an end user) to control its interdomain routing, by mapping the client's traffic into tunnels to achieve more flexible interdomain traffic control. In SDX [5], an Internet exchange point, which can also be considered as an AS, allows a direct upstream AS of the exchange to control how traffic is distributed among its downstream, and a direct downstream AS to control how traffic gets into the downstream, using flexible matching conditions (*e.g.*, match on TCP/IP 5-tuple). Although these systems provide clients more mechanisms than BGP to control interdomain routing, they either are point solutions (*e.g.*, SDX) or may have datapath overhead such as tunneling processing on each data packet.

In this paper, we investigate a novel, systematic, low-overhead interdomain route control model which we call the software-defined internetworking (SDI) model. Motivated by the success of intradomain SDN models such as Openflow or P4 but extending to interdomain, SDI defines an interdomain

programmable interface so that a network exposes to a client its available interdomain routes (*i.e.*, its interdomain routing information base) to a destination, and the client can then choose one of them, just as an intradomain SDN client can select a port as the next hop among a set of available output ports of an SDN switch; Different from intradomain SDN, however, SDI maximizes network autonomy, by allowing a network to maintain the control of its interdomain export policies, to avoid fundamental violations such as valley routing.

Allowing client control and at the same time maximizing network autonomy by allowing a network to fully control its export policies exposes challenging problems not investigated before, although the problems also exist in aforementioned early work. Specifically, consider the same interdomain network in Fig. 1 and assume that the ASes have the following autonomous, *private* export policies: (1) AS D will not announce any AS path containing AS E to AS B; (2) AS C will not announce any AS path containing AS E to AS A; (3) AS B will not announce any AS path containing AS G to AS A. Still consider AS S, whose goal is to send traffic toward p along a shortest AS path. Conducting SDI control, AS S may ask AS D and AS A to select route $[D, E, T]$ and route $[A, B, D, E, T]$, respectively, hoping that it can then use the route $[S, A, B, D, E, T]$. However, because D will not announce route $[D, E, T]$ to B due to its export policy, the route $[S, A, B, D, E, T]$ cannot be used by S. In the worst case, only after enumerating many paths can S find that $[S, A, C, D, T]$ is the only *export-policy-compliant* shortest AS path. Such an interdomain SDN with autonomy problem has not been studied before.

We formulate the optimal interdomain SDI route control with policy compliance problem, which we refer to as the optimal end-to-end SDI routing problem. We conduct rigorous analysis on the computational complexity of this problem and prove its strong NP-hardness. The fundamental reason behind the computational intractability of this problem is that each AS can autonomously make and execute its own export policy. Previous studies [11], [12] show that when ASes all follow the Gao-Rexford route selection and export policies based on the customer/peer/provider business relationship [13], the shortest policy-compliant route can be computed in polynomial time. However, a recent survey [14] on many network operators shows that a high percentage of ASes are actually breaking the Gao-Rexford condition in their export policies.

We develop a blackbox optimization algorithm to sample end-to-end routes sequentially and find a near-optimal policy-compliant end-to-end route with a small number of samples. Our algorithm, built on the Bayesian optimization framework, is highly efficient for two reasons. First, it iteratively leverages the prior belief about the problem to help direct the sampling, and to trade exploration and exploitation of the search space [15], [16]. Second, it leverages important properties from interdomain routing algebra [17], [18] to derive an accurate estimation on the expected improvement of an end-to-end route, which avoids the efficiency loss brought by rounding continuous sample points into discrete sample routes [19].

Symbol	Meaning
v	An AS
(u, v)	An edge from AS u to AS v in the AS Graph
$\mathbf{r}$	A route (AS Path)
p	A destination IP prefix
$e_v(p, \mathbf{r})$	The export policy of AS v for prefix p and route $\mathbf{r}$
$c_v(\mathbf{r})$	The pricing policy of AS v for route $\mathbf{r}$
f	The objective function

TABLE I: Symbols

The **main** contributions of this paper are as follows:

- We provide the first, systematic formulation of the software-defined internetworking(SDI) model, extending intradomain SDN to generic interdomain SDN;
- We systematically study the optimal end-to-end SDI routing problem, and show that the problem is strongly NP-hard;
- We develop a blackbox optimization algorithm, which integrates the Bayesian optimization theory and important properties in interdomain routing algebra, to efficient find near-optimal end-to-end routes with a small number of route sampling;
- We implement a prototype of our algorithm and evaluate its performance via extensive experiment using real-world topology. Results show that in an interdomain network with over 60000 ASes and over 320000 AS-level links, in 80% experiment cases, the blackbox optimization algorithm can find a near-optimal policy-compliant end-to-end route by sampling less than 33 routes.

II. SDI Model and Problem Formulation

In this section, we first give the systematic formulation of the SDI model. Then we define the problem of finding a route that is compliant to local policies and achieves global optimal objectives, namely the optimal end-to-end SDI routing problem.

A. Basic Definitions

Global network. We use a graph $G = (V, E)$ to denote an interdomain AS network, where ASes are interconnected by BGP[1]. A vertex $v \in V$ represents an AS in the network, and a bidirectional edge $(u, v) \in E$ represents a BGP session between AS u and AS v.

Interdomain route. An interdomain route $\mathbf{r}$ is represented as a sequences of ASes $[v_1, v_2, \ldots, v_n]$ (also called an AS path), where $v_i \neq v_j$ for any two integers $i \neq j \in [1, n]$. Given a route $\mathbf{r}$, we call v_1 the source AS and v_n the destination AS. Given two routes $\mathbf{r}_1 = [v_1, \ldots, v_n]$ and $\mathbf{r}_2 = [u_1, \ldots, u_m]$, if $(v_n, u_1) \in E$ and $\forall i \in [1, n], \forall j \in [1, m], v_i \neq u_j$, the concatenation of $\mathbf{r}_1$ and $\mathbf{r}_2$, represented as $\mathbf{r}_1 \oplus \mathbf{r}_2 = [v_1, \ldots, v_n, u_1, \ldots, u_m]$, is also a route.

Network configuration. Each AS node v is configured with two key attributes: an export policy, and a pricing policy. In

[1]For simplicity of presentation, we stick to the one-big-switch abstraction widely used in BGP studies (*e.g.*, [1], [13], [17], [20]).

this paper, we assume the ASes are propagating the routing information from destination to source. Thus, the export policy also only applies to the routes from the current AS to the destination. The export policy is modeled as a function $e_v(p, \mathbf{r}, u)$. It takes a destination IP prefix p and a route $\mathbf{r}$, and returns either 1 if AS v will announce route $[v] \oplus \mathbf{r}$ to its neighbor u if $\mathbf{r}$ is available, or 0 otherwise. A pricing model is important when an AS allows a client to override the network's default interdomain route selection policy. In this paper, we denote the pricing model as a function $c_v(\mathbf{r})$. It takes a route $\mathbf{r}$ and returns a real number, which stands for the price to set up $\mathbf{r}$ for a match in AS v.

***Definition** 1 (Policy-Compliant Route):* A route $\mathbf{r} = [v_1, v_2, \ldots, v_n]$ toward a destination IP prefix p is export-policy-compliant or simply policy-compliant if and only if the following condition holds:

$$e(p, \mathbf{r}) \triangleq \bigwedge_{i=2}^{n} e_{v_i}(p, [v_{i+1}, \ldots, v_n], v_{i-1}) = 1. \quad (1)$$

With the aforementioned model, one is able to verify if an AS path is policy compliant and also to calculate the total cost of establishing a route $\mathbf{r} = [v_1, \ldots, v_n]$ as follows:

$$c(\mathbf{r}) \triangleq \sum_{i=1}^{n} c_{v_i}([v_i, v_{i+1}, \ldots, v_n]). \quad (2)$$

Client's objective function. An objective function $f(\mathbf{r})$ takes route $\mathbf{r}$ and returns a real number, which represents the objective for the route. Without loss of generality, we focus on functions f that satisfy the following two properties:

- *monotonicity*: Given a route $\mathbf{r}$, $f([u, v] \oplus \mathbf{r}) \geq f(\mathbf{r})$;
- *isotonicity*: Given two routes $\mathbf{r}_1$ and $\mathbf{r}_2$, if $f(\mathbf{r}_1) \geq f(\mathbf{r}_2)$, then $f([u, v] \oplus \mathbf{r}_1) \geq f([u, v] \oplus \mathbf{r}_2)$.

A typical example of the monotone and isotonic objective function is the shortest AS path, where $f(\mathbf{r})$ is the inverse of the AS path length. Other monotone and isotonic objective functions include weighted shortest AS path[2], widest shortest AS path, reachability, etc. A more complete list of monotone and isotonic objective functions can be found in [21].

B. SDI Programmable Network

SDI programming service at an AS. In addition to an actual BGP speaker, each AS $v \in G$ also runs a virtual BGP speaker, which has the same route selection and export policies as the actual BGP speaker of v, and establishes BGP sessions with the virtual BGP speakers of the neighboring ASes of v in G.

Each AS v exposes the following information to clients. First, given a destination IP prefix p specified by a client, it exposes the routing information base (RIB), *i.e.*, all available routes v has to reach p. Second, given a destination IP prefix p and a route $\mathbf{r}$ from v to p specified by a client, it exposes the price for the client to use $\mathbf{r}$. Exposing such information will not expose the private policies (*i.e.*, the route selection / export / pricing policies) of ASes, because inferring these policies based on the exposed information is a constraint acquisition problem, which is computationally intractable [22].

Each AS v provides three programming interfaces to clients. Specifically, it uses a two-phase commit design pattern to allow a client to test the policy-compliance of an end-to-end interdomain route $\mathbf{r}$ before actually using and paying for it. This design has two advantages: (1) it avoids the disruptions and churns in the interdomain network caused by the client using a non-policy-compliant route; (2) it avoids the waste of monetary expense of the client paying for a non-policy-compliant route. The programming interfaces are as follows.

- $selectRoute(match, \mathbf{r})$: This interface is to select a route $\mathbf{r}$ to forward the class of traffic specified by the packet header matching condition $match$ in the virtual BGP speaker. Upon receiving this request from a client, AS v first checks if $\mathbf{r}$ is in its RIB. If so, it selects this route in its virtual BGP speaker, and lets the virtual speaker follow its export policy to announce $\mathbf{r}$ to neighboring virtual BGP speakers.
- $commitRoute(match, \mathbf{r})$: This interface is to install a route $\mathbf{r}$ in the forwarding information base (FIB) of AS v, to forward the class of traffic specified by the packet header matching condition $match$. AS v first checks if $\mathbf{r}$ is in its RIB. If so, it will configure its real BGP speaker to install this route in the FIB of the router, and follow its export policy to announce this route to neighboring BGP speakers.
- $deleteRoute(match, \mathbf{r})$: This interface is to stop using the route r for the class of traffic specified by $match$. AS v will remove the $\mathbf{r}$ from its FIB.

Interaction between client and ASes. A client can use the exposed information and programming interfaces of ASes to achieve flexible, end-to-end interdomain route control.

First, given an end-to-end route $\mathbf{r} = [v_1, v_2, \ldots, v_n]$, a client checks if it is policy-compliant. Specifically, a client can iteratively interact with the ASes along $\mathbf{r}$ in backward order: for any $i = n, \ldots, 2$, the client asks AS v_i to select route $[v_i, \ldots, v_n]$ in its virtual BGP speaker using the $selectRoute$ interface and observes the RIB of AS v_{i-1}. If route $[v_i, \ldots, v_n]$ is observed in the RIB of v_{i-1} for all $i = n, \ldots, 2$, the route $\mathbf{r}$ is policy-compliant.

Second, a client also interacts with the ASes along $\mathbf{r}$ to evaluate the total cost of using $\mathbf{r}$. This can be achieved by asking the price of using route $[v_i, v_{i+1}, \ldots, v_n]$ for each AS v_i along $\mathbf{r}$.

Third, after the client finds a policy-compliant end-to-end route $\mathbf{r} = [v_1, v_2, \ldots, v_n]$ he wants to use for forwarding certain traffic. He can interact with the ASes along $\mathbf{r}$ in a backward order (*i.e.*, from v_n to v_1) to setup the this route. Specifically, for each AS v_i, the client uses the $commiteRoute$ interface to request v_i to install a route $[v_i, v_{i+1}, \ldots, v_n]$ in its forwarding information base to forward the traffic specified by the packet header matching condition $match$.

[2]With all weights being positive

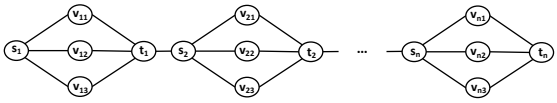


Fig. 2: An illustration of constructing G from an instance of 3-SAT problem.

C. Optimal End-to-End SDI Routing Problem

Based on the aforementioned models, we now formally define the optimal end-to-end SDI routing problem.

***Problem** 1 (Optimal End-to-End SDI Routing Problem):* For an interdomain network $G = (V, E)$, where the export policy and the pricing policy of AS v are denoted as $e_v(p, \mathbf{r}, u)$ and $c_v(\mathbf{r})$ respectively, and $f(\mathbf{r})$ denote the global objective function while B denote the cost budget, the optimal policy-compliant route $\mathbf{r}^*$ for a match m from a source AS s to a destination AS t is the solution of the following optimization problem:

$$\text{maximize} \quad f(\mathbf{r})$$

subject to,

$$v_1 = s, v_n = t, e(p, \mathbf{r}) = 1, c(\mathbf{r}) \leq B$$

III. Complexity Analysis

In this section, we analyze the computational complexity of Problem 1. Our main finding is summarized in the following theorem:

***Theorem** 1:* The optimal end-to-end SDI routing problem (Problem 1) is strong NP-hard.

To prove the NP-hardness of Problem 1, we first consider the following problem:

***Problem** 2 (Shortest Policy Compliant AS-Path Problem):* Assume an interdomain network $G = (V, E)$, where ASes not only provide the SDI service described in Section II, but also expose their export policies to the client. Given a source AS v_s, a destination AS v_d, a positive integer K, and a positive real number B, is it possible to find a route $\mathbf{r}$ from v_s to v_d, such that (1) the AS path length of $\mathbf{r}$ is less than or equal to K, (2) $\mathbf{r}$ is policy-compliant, and (3) the total cost of using $\mathbf{r}$ is less than or equal to B.

It is easy to see that Problem 2 is a simplified variant of the optimal end-to-end SDI routing problem. And we propose and prove the following lemma:

***Proposition** 1:* Problem 2 is strong NP-hard.

Proof: We prove the NP-hardness of Problem 2 via a reduction from the 3-SAT problem. Specifically, given an instance of a 3-SAT problem with n clauses $\{C_1, C_2, \ldots, C_n\}$. We use x_{ij} to denote the j-th literal in the i-th clause C_i. For each clause C_i, we construct a graph $G_i = (V_i, E_i)$, where $V_i = \{s_i, t_i, v_{i1}, v_{i2}, v_{i3}\}$, and $E_i = \{(s_i, v_{ij}), (v_{ij}, t_i)|$ where $j = 1, 2, 3.\}$.

We then can construct an instance of Problem 2 as follows: First, we construct the graph G by connecting G_i and G_{i+1} with an edge (t_i, s_{i+1}) for $i = 1, 2, \ldots, n-1$ (Fig. 2). For each AS v_{ij}, we define its export policy as follows: given an AS path $\mathbf{r}$, if $\mathbf{r}$ contains an AS $v_{i'j'}$, whose corresponding literal $x_{i'j'} = \neg x_{ij}$, AS v_{ij} will not announce $\mathbf{r}$ to AS t_i. For each AS s_i, we define its export policy as: announcing every route to every neighbor v_{ij}. For each AS t_i, we define its export policy as: announcing every route to neighbor s_{i+1}. We set $K = 3n - 1$. For each AS in the constructed graph G, we set its pricing policy to be always charging the client $B/3n$ for any route the client want so use. We suppose the client wants to find an end-to-end route $\mathbf{r}$ from s_1 to t_n such that (1) $\mathbf{r}$ policy-compliant, (2) $\mathbf{r}$ has an AS path length less than or equal to K, and (3) the cost of using $\mathbf{r}$ is less than or equal to B. We can see that this process of constructing an instance of Problem 2 from an instance of the 3-SAT problem is polynomial.

After the construction, if the 3-SAT instance is satisfiable, assume the in clause C_i, the j_i-th literal x_{ij_i} is true, then the route $[s_1, v_{1j_1}, t_1, s_2, v_{2j_2}, t_2, \ldots, t_n]$ is a policy-compliant route in the constructed instance of Problem 2. On the other hand, if there exists a policy-compliant route $[s_1, v_{1j_1}, t_1, s_2, v_{2j_2}, t_2, \ldots, t_n]$ in the constructed instance of Problem 2, a satisfiable truth assignment can be found for the original 3-SAT instance by setting literal x_{ij_i} to be true. As such, we complete the proof. ■

With the proof of Proposition 1, the correctness of Theorem 1 is an immediate result.

IV. Blackbox Optimization Algorithm

In this section, we first present a strawman solution based on the Yen's k-shortest-path algorithm [23], and show that its poor performance is caused by 1) the high latency of testing the policy-compliance of routes and 2) the worst case of exponential enumeration complexity. To resolve these problems, we devise an efficient blackbox optimization algorithm, which samples end-to-end routes sequentially and finds a near-optimal policy-compliant end-to-end route with a small number of samples.

A. Naive Route Enumeration Algorithm

A naive solution is to iteratively check the k-th optimal route from v_s to v_t computed by the Yen's algorithm [23] until finding the first policy-compliant route of which cost does not exceed B. Its pseudocode is presented in Algorithm 1.

Algorithm 1: Naive Route Enumeration Algorithm.

```
foreach k = 1, 2, ..., do
    Compute the k-th optimal AS path using the Yen's
      algorithm and denote as r_k;
    Test the policy-compliance of r_k;
    Compute the cost of using r_k;
    if r_k is policy-compliant and c(r_k) ≤ B then
        return r_k;
return null;
```

Specifically, in this solution, Yen's algorithm employs a generalized Dijkstra's algorithm proposed in [21]. In each iteration k, the Yen's algorithm computes a set of deviation paths from the $k-1$-th optimal path $\mathbf{r}_{k-1}$, which shares a

subroute with $\mathbf{r}_{k-1}$. Among the newly computed deviation paths and the computed but unselected deviation paths in previous iterations, the algorithm selects the one with the largest utility $f()$ as the k-th optimal path. As such, this naive solution can always find the optimal end-to-end route that is policy-compliant and does not exceed the client's budget.

However, there are two issues with this solution. First, in each iteration, a client needs to test the policy-compliance of the k-th shortest AS path $\mathbf{r}$. This process incurs a long latency because the client needs to interact with ASes along $\mathbf{r}$ sequentially. Second, in the worst case, the algorithm needs to enumerate all possible routes between v_s and v_t, of which complexity is $O(V!)$.

B. Blackbox Optimization Algorithm

Given the hardness of finding the optimal end-to-end route, we instead design a blackbox optimization algorithm that samples end-to-end routes sequentially and finds a near-optimal policy-compliant end-to-end route with a small number of sampled routes. The key insights behind our algorithms are: First, it leverages the prior belief about the problem to help direct the sampling, and to trade exploration and exploitation of the search space [15], [16]. Second, it leverages important properties from interdomain routing algebra [17], [18] to derive an accurate estimation on the expected improvement of an end-to-end route, which avoids the efficiency loss brought by rounding continuous sample points into discrete sample routes [19].

Reformulation as blackbox optimization problem. We first transfer the formulation of Problem 1 into a blackbox optimization problem. For simplicity, we omit prefix p from function e and use $e(\mathbf{r})$ instead. We move $e(\mathbf{r})$ from the constraint to the objective, and get the blackbox optimization formulation as follows:

$$\text{maximize} \quad u(\mathbf{r}) = e(\mathbf{r})f(\mathbf{r})$$

subject to,

$$v_1 = s, v_n = t, c(\mathbf{r}) \leq B$$

Based on the fact that both $e(\mathbf{r})$ and $c(\mathbf{r})$ are unknown to the client, but can be observed through the interaction between client and ASes, our algorithm utilizes the framework of Bayesian Optimization (BO), a powerful framework for solving optimization problems whose objective functions and constraints are unknown beforehand, but can be observed through experiments [15], [16].

Sampling routes using Bayesian optimization. BO is a sequential model-based approach. Within this framework, we can define a prior belief over the possible objective functions $u(\mathbf{r}) = e(\mathbf{r})f(\mathbf{r})$ and constraints $c(\mathbf{r})$, and then sequentially refines this model as end-to-end routes are sampled via Bayesian posterior updating [16]. To sample efficiently, BO uses an *acquisition function* to determine the next route to sample. As such, BO has the nice property that it typically only needs to sample a small number of routes to find a near-optimal end-to-end route.

Different types of acquisition functions can be used in the BO framework to guide the sequential sampling process. We adopt the Expected Improvement (EI) acquisition function as it has been verified to perform better than other acquisition functions in most cases [24], [25].

Given a route $\mathbf{r}$ that $u(\mathbf{r})$ has not been evaluated yet, we define its improvement function as follows:

$$I(\mathbf{r}) = max\{u(\mathbf{r}) - u(\mathbf{r}^+), 0\}, \tag{3}$$

where $\mathbf{r}^+$ is the route that provides the maximum objective value observed so far. In other words, $I(\mathbf{r})$ is positive when the prediction of $u(\mathbf{r})$ is higher than the best value $u(\mathbf{r}^+)$ so far. Otherwise, $I(\mathbf{r})$ is set to zero.

Next route $\mathbf{r}$ to evaluate is the one that maximizes the expected improvement:

$$\mathbf{r} = \underset{\mathbf{r}}{\operatorname{argmax}}\, E[I(\mathbf{r})|D_t],$$

where D is a set of routes whose value of $u()$ is known.

If we assume $u(\mathbf{r})$ in Equation IV-B is a Gaussian process, we can adopt a covariance function (*i.e.*, the kernel function) to measure the similarity between to different routes (*e.g.*, the $Matern5/2$ kernel [25]), and plug in the kernel function in to the close-form expression of EI [26]. However, in the optimal end-to-end route problem, the search space is discrete.

A strawman approach to tackle the discrete search space is to still use the close-form expression of EI derived in [26] to maximize EI, and then rounding the result to get the next route $\mathbf{r}$. However, it has been reported in recent studies that this rounding approach performs poorly in the BO framework [19]. As such, we need to explore other approaches to compute EI for discrete decision variables (*i.e.*, routes).

Discrete EI maximization leveraging properties from interdomain routing algebra. Our solution to the discrete EI maximization problem is to leverage the following two properties derived from interdomain routing algebra [17], [18] to derive an accurate estimation of $EI(\mathbf{r})$.

***Proposition** 2 (Subroute Policy-Compliance):* If a route $\mathbf{r} = [v_1, v_2, \ldots, v_n]$ is policy-compliant, any route $[v_i, \ldots, v_n]$, where $i = 1, 2, \ldots, n$ is policy-compliant.

***Proposition** 3 (Non-policy-compliance by inclusion):* If a route $\mathbf{r} = [v_1, v_2, \ldots, v_n]$ is not policy-compliant, any route $\mathbf{r}'$ that is constructed by prepending a segment in graph G before $\mathbf{r}$ is not policy-compliant.

These two properties help the client to better predict the probability of a route $\mathbf{r}$ being policy-compliant or not. Specifically, assuming the export policies of ASes are independent from each other, given a route $\mathbf{r} = [v_1, v_2, \ldots, v_n]$, we use $P_{i,i-1}([v_i, \ldots, v_n]|D)$ to represent the probability that AS v_i will announce route $[v_i, \ldots, v_n]$ to AS v_{i-1}, given the current set of sampled routes D.

Then the probability that $\mathbf{r}$ is policy-compliant, represented by $P_{\mathbf{r}}$, is computed as

$$P_{\mathbf{r}} = \prod_{i=n,\ldots,2} P_{i,i-1}([v_i, \ldots, v_n]|D). \tag{4}$$

Then we have

$$E[e(\mathbf{r})|D] = 1 \cdot P_{\mathbf{r}} + 0 \cdot (1 - P_{\mathbf{r}}) = \prod_{i=n,\ldots,2} P_{i,i-1}([v_i, \ldots, v_n]|D). \tag{5}$$

With Equation 5, we can do the following derivation on the expression of expected improvement of a route:

$$\begin{aligned} E[I(\mathbf{r})|D] =& E[\max\{u(\mathbf{r}) - u(\mathbf{r}^+), 0\}|D] \\ =& \max\{E[u(\mathbf{r})|D] - u(\mathbf{r}^+), 0\} \\ =& \max\{\frac{E[e(\mathbf{r})|D]}{f(\mathbf{r})} - u(\mathbf{r}^+), 0\} \\ =& \max\{f(\mathbf{r}) \prod_{i=n,\ldots,2} P_{i,i-1}([v_i, \ldots, v_n]|D) - u(\mathbf{r}^+), 0\} \end{aligned} \tag{6}$$

In addition, as suggested by [27], we also add another term to the right-hand side of Equation 7, changing it to:

$$\begin{aligned} E(I(\mathbf{r})|D) =& P(c(\mathbf{r}) \leq B) \cdot \\ & max\{f(\mathbf{r}) \prod_{i=n,\ldots,2} P_{i,i-1}([v_i, \ldots, v_n]|D) \\ & - u(\mathbf{r}^+), 0\}, \end{aligned} \tag{7}$$

to make the choice of next route to test bias towards one that is likely to satisfy the cost constraint.

Next we derive an estimator on $P_{i,i-1}([v_i, \ldots, v_n]|D)$. In particular, during the sampling process, we keep the record on the policy-compliance test results for previous sampled routes in D. For any two ASes v_i, v_{i-1} in $\mathbf{r}$, we calculate two values:

- $w_{i,i-1}(\mathbf{r})$: to count the number of routes in D such that (1) AS v_i announces its subroute to AS v_{i-1}, and (2) the announced subroute intersects with the subroute $[v_i, \ldots, v_n]$.
- $b_{i,i-1}(\mathbf{r})$: the number of routes in D such that (1) AS v_i does not announce its subroute to AS v_{i-1}, and (2) the unannounced subroute intresetcts with the subroute $[v_i, \ldots, v_n]$.

Then we can approximate $P_{i,i-1}([v_i, \ldots, v_n]|D)$ as:

$$P_{i,i-1}([v_i, \ldots, v_n]|D) = \frac{exp(w_{i,i-1}(\mathbf{r}))}{exp(w_{i,i-1}(\mathbf{r})) + exp(b_{i,i-1}(\mathbf{r}))}. \tag{8}$$

Plugging Equation 8 into Equation 7, we get a close-form estimation on $E(I(\mathbf{r})|D)$, the expected improvement of route $\mathbf{r}$ over sampled set of routes D.

Two-level sequential sampling to quickly find near-optimal end-to-end route. We now present our complete blackbox optimization algorithm for the optimal end-to-end route problem (Algorithm 2). Note that the algorithm also contains several subroutines. Due to the page limit, we only present two most important ones for reference (*i.e.*, Algorithm 4 and Algorithm 3).

In the initialization phase, the algorithm defines 4 empty sets: R_F is the set of policy-noncompliant subroutes found during the test of route policy-compliance; D is the set of sampled routes that have been evaluated for policy-compliance and cost-compliance; R_c is the set of sampled routes that have not been evaluated for policy-compliance or cost-compliance; R_q is the set of routes that are found to be both policy- and cost-compliant during the search process (Line 1).

Algorithm 2: Blackbox Optimization Algorithm.

1 $R_F = \emptyset, D = \emptyset, R_c = \emptyset, R_q = \emptyset$;
2 $\mathbf{r} = genericDijkstraAlg(G, v_s, v_t)$;
3 $e(\mathbf{r}) = PCTest(\mathbf{r}, R_F)$;
4 **if** $e(\mathbf{r}) == 1$ *and* $getCost(\mathbf{r} \leq B)$ **then**
5 **return** $\mathbf{r}$;
6 **else**
7 $\mathbf{r}_{last} = \mathbf{r}$;
8 $u(\mathbf{r}_{last}) = e(\mathbf{r}) \times f(\mathbf{r})$;
9 $D = D \cup (\mathbf{r}_{last}, u(\mathbf{r}_{last}))$;
10 $t_{local} = t_{global} = 0$;
11 **while** $t_{global} >= T_{global}$ *and* $R_q \neq \emptyset$ **do**
12 **if** $t_{local} >= T_{local}$ *and* $R_q == \emptyset$ **then**
13 $t_{local} = 0$;
14 $R = getGlobalCandidates(G, R_F, v_s, v_t)$;
15 **else**
16 $R = getLocalCandidates(G, \mathbf{r}_{last})$;
17 **foreach** $\mathbf{r} \in R$ **do**
18 **if** $isForbidden(\mathbf{r}, R_F)$ **then**
19 $D = D \cup (\mathbf{r}, 0)$;
20 $R = R - \{\mathbf{r}\}$;
21 $R_c = R_c \cup R$;
22 $\mathbf{r}_t = \text{argmax}_{\mathbf{r} \in R_c} E(I(\mathbf{r})|D)$, where $E(I(\mathbf{r})$ is computed in Equation 7;
23 $R_c = R_c - \{\mathbf{r}_t\}$;
24 $e(\mathbf{r}) = PCTest(\mathbf{r}, R_F)$;
25 $D = D \cup (\mathbf{r}_t, u(\mathbf{r}_t^g))$;
26 **if** $e(\mathbf{r}) == 1$ *and* $getCost(\mathbf{r} \leq B)$ **then**
27 $R_q = R_q \cup \{\mathbf{r}\}$;
28 $t_{local} = t_{local} + 1, t_{global} = t_{global+1}$;
29 $\mathbf{r}_{last} = \mathbf{r}_t$;
30 **return** $\text{argmax}_{(r) \in R_q} f(\mathbf{r})$;

As a starting point, the algorithm chooses an optimal end-to-end route computed by the general Dijkstra algorithm [21], and tests its policy-compliance and cost (Line 2-4, and Algorithm 4). If this route is indeed policy-compliant and does not exceed the client's budget, the algorithm simply stops since the optimal solution has been found (Line 5). Otherwise, the algorithm keeps the record of this route in set D (Line 7-9), and enters the sequential sampling process (Line 11).

In each iteration of the sampling process, the algorithm first tries to do a "local exploration", *i.e.*, it samples a set of candidate routes R, each of which shares a subroute with $\mathbf{r}_{last}$, the last route that is tested for policy-compliance and cost-compliance (Line 16 and Algorithm 3). If the algorithm has already sampled locally for a total of T_{local} iterations and still cannot find any route that is both policy-compliant and cost-compliant, it will do a "global exploration" by sampling a set of random candidate routes from v_s to v_t that differs from

the found policy-non-compliant subroutes in R_F as much as possible, and resets the number of local exploration t_{local} (Line 13-14).

Algorithm 3: Local exploration of candidate routes: $getLocalCandidates(G, \mathbf{r})$.

1 $R_c = \emptyset$;
2 $\mathbf{r} = [v_1, v_2, \ldots, v_n]$;
3 **foreach** $i = 2, \ldots, n-1$ **do**
4 $\quad E_G^i = E_G - \cup_{j=1,\ldots,i}\{(v_j, v_{j+1})\}$;
5 $\quad V_G^i = V_G$;
6 $\quad \mathbf{r}_i = [v_1, \ldots, v_i] \oplus genericDijkstra((V_G^i, E_G^i), v_i, v_n)$;
7 $\quad R_c = R_c \cup \{\mathbf{r}_i\}$;
8 **return** R_c;

After exploring candidate routes, the algorithm leverages Proposition 3 to only keep candidate routes that do not contain any subroute in R_F (Line 17-20), merges them with candidate routes left before in R_c (Line 21), and selects $\mathbf{r}$ from the remaining sampled routes whose expected improvement is the largest among all the sampled, yet unevaluated routes R_ci (Line 22). After that, the algorithm evaluates the policy-compliance and cost-compliance of $\mathbf{r}$, and added it to R_q if it is both policy-compliant and cost-compliant (Line 24-27).

The sequential sampling process terminates after a total number of T_{global} iterations or R_q becomes non-empty. Then the route which has the largest utility value $f()$ is the resulting near-optimal, end-to-end route (Line 30).

Algorithm 4: Policy-compliance test of $\mathbf{r}$: $PCTest(\mathbf{r}, R_F)$.

1 Input: $\mathbf{r} = [v_1, v_2, \ldots, v_n]$;
2 **foreach** $i = n, \ldots, 2$ **do**
3 $\quad$ Call $selectRoute(match, [v_i, \ldots, v_n])$ at AS v_i;
4 $\quad$ Get RIB_{i-1} at AS v_{i-1};
5 $\quad$ **if** $[v_i, \ldots, v_n] \notin RIB_{i-1}$ **then**
6 $\quad\quad R_F = R_F \cup [v_{i-1}, v_i, \ldots, v_n]$;
7 $\quad\quad$ **return** 0;
8 **return** 1;

Discussion. Due to the combinatory nature of Problem 1, it is very challenging to derive an explicit convergence bound of our blackbox optimization algorithm. In order to guarantee that at least one policy-compliant and cost-compliant route is found, we explicitly specify it as part of the stopping condition. As such, in the worst case, our blackbox optimization algorithm still needs to enumerate a large number of routes. However, in practice, this worst case would rarely happen. This is because, with the help of the expected improvement acquisition function, our algorithm can smartly decide the next route to test for policy and cost compliance, substantially reducing the numbers of sample routes needed. In addition, by leveraging Proposition 3, our algorithm substantially reduces the ratio of sampled routes that require policy-compliance test between the client and ASes, which is the most expensive part of enumeration. We validate our argument on the efficiency and efficacy of our algorithm using real-world data. As we will show in the next section, in an interdomain network with over 60000 ASes and over 320000 AS-level links, in 80% experiment cases, the blackbox optimization algorithm can find a near-optimal policy-compliant end-to-end route by sampling less than 33 routes.

V. Evaluation

In this section, we conduct extensive evaluations aiming to answer the following questions: 1. How fast can the Bayesian Optimization framework find the (near-)optimal end-to-end route? 2. How likely can the Bayesian Optimization framework find a (near-)optimal end-to-end route in a reasonable time?

A. Experiment Setting

Network Topology. To evaluate our approach, we use the AS-level Internet topology derived from the CAIDA dataset [28] which includes 63361 ASes and 320978 AS-level links.

Initial RIB. Each AS follows the derived AS business relationship to set its local preference for BGP route selection by default, which provides the initial RIB information.

Export Policies. Each AS configures the following types of route export policies to determine whether announcing a received route to a neighbor AS: 1. *Neighboring Business Relationship*: An AS never exports a non-customer route to a non-customer neighbor; 2. *Blackhole*: For some security reason, *e.g.*, BGP hijacking prevention, an AS may not export any routes across a set of suspected ASes; 3. *Forbiden Segment*: For some security or business reason, an AS may not export any routes including some specific segments.

Searching Algorithms. We implement the following end-to-end route search algorithms: 1. *Shortest Path Enumeration* (SPE): Use YEN's algorithm to iterate K-shortest path in order until the first policy-compliant route is found; 2. *Software-Defined Internetworking Bayesian Optimization* (SDIBO): Use the Bayesian Optimization framework with the discrete EI acquisition function and the two-level sequential sampling approach to finding the near-optimal policy-compliant route.

Intents. We generate global end-to-end intents based on Internet traffic dataset [28]: we choose the top 10 ASes which receive the most traffic volume as destinations. For each selected destination AS, we choose top 200 source ASes sending the most traffic volume to this destination AS. We generate 2000 end-to-end intents in total and check the connectivity[3] in the derived AS-level Internet topology. Finally, we get 1620 effective end-to-end intents.

Experiment Workflow. We implement a BGP simulator to simulate the RIB computation and route announcement, and a prototype SDI client to find the optimal end-to-end route. We design the experiment as follows: First, we run our customized BGP simulator for this AS-level Internet topology until it converges, and collect the RIB of each AS as the initial prior knowledge. Then we feed the converged topology with

[3]We suspect the reason of disconnectivity is because the inferred AS relationship is incomplete.

the BGP RIBs to our prototype SDI client. After that, the prototype SDI client proposes a global end-to-end intent and tries to find the optimal policy-compliant route within the minimal number of policy-compliance tests by using one of end-to-end route search algorithms above.

Policy-compliance Tests. There are two methods to test whether a path is policy-compliant as a whole or at a certain AS as follows: 1. *Active Test*: The client actively sends a test request of route $\mathbf{r} = [v_1, \ldots, v_n]$ to the remote virtual BGP speaker at v and observes whether $[v] \oplus \mathbf{r}$ appears in a given neighbor u. 2. *Local Test*: If the client has ever actively tested the visibility of a route for a node, it will cache the result. Next time when the client has to check it again, the client only need to locally check it using the cache.

B. Searching Efficiency

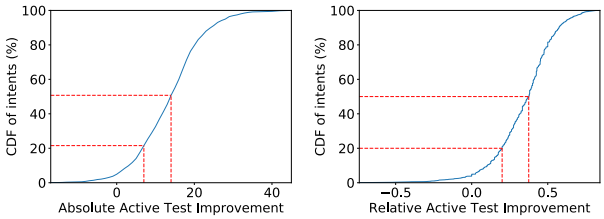
In this section, we evaluate the searching efficiency of different search algorithms, *i.e.*, how fast they can find the (near-)optimal end-to-end route.

Metric. We use the following metrics to measure the searching efficiency of an algorithm: 1. *Number of Active Tests*: It represents the number of active test requests sent before finding the solution. 2. *Number of Local Tests*: It represents the number of local tests (*i.e.*, cache lookups). 3. *Average Number of Active/Local Tests*: It represents the average number of active/local tests on a path for an intent. 4. *Absolute Active/Local Test Improvement*: It is calculated by subtracting the number of active/local tests for SPE by the number of active/local tests for SDIBO. 5. *Relative Active Test Improvement*: It is calculated by dividing the absolute active test improvement by the number of active tests for SPE.

Early Termination Thresholds. The route optimization problem is NP-hard. So in the worst case, whatever the algorithm is adopted, the client has to enumerate all simple paths from the source to the destination. It is not realistic in practice so we set two thresholds to terminate the searching: 1. *Maximum Number of Active Tests*: In practice, an active test action usually takes a long time. So we set a threshold (60) to limit the maximum number of such actions for each intent. 2. *Maximum Number of Local Tests*: Local test action is usually quick. However, in the worst case, one will enumerate all routes. Considering the scale of the Internet, it still may take intolerant time. So we also set a threshold (2500) to limit the maximum number of local tests for each intent.

Results. We first demonstrate the performance gain of SDIBO for each intent. Fig. 3 demonstrates the CDF curve of the absolute active test improvement and the relative active test improvement. As we can see, **SDIBO improves the active test in most cases and is likely to get substantial improvement**: for 80% of the requests, SDIBO can reduce more than 7 active tests and more than 20% of active tests made by SPE; for 50% requests, SDIBO can reduce more than 15 active tests and by 40% to 80% of active tests made by SPE, which yields an improvement of 1.6x to 5x on the execution time.

To further understand the performance of SDIBO, we draw a scatter figure where the x-coordinate represents the absolute active test improvement and y-coordinate represents the absolute local test improvement. As we can see in Fig. 4, most dots fall into the lower right region ($x > 0$, $y < 0$, red color), where SDIBO takes less active tests but more local tests. This indicates that **SDIBO searches routes more effectively and fully leverages the local information**. We also see that many dots even fall into the upper right region ($x > 0$, $y > 0$, green color), where SDIBO uses less active tests and fewer local tests, which further demonstrates the benefits of SDIBO's intelligent searching method.



(a) Absolute Improvement **(b)** Relative Improvement

Fig. 3: CDF of Active Test Improvement

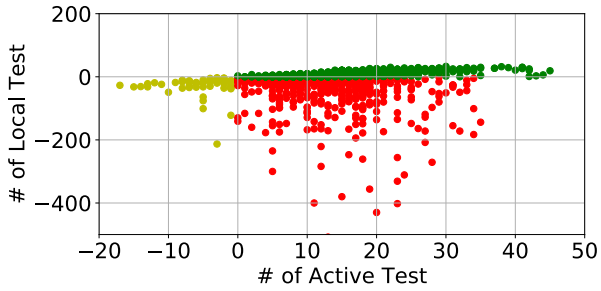


Fig. 4: Absolute Active/Local Test Improvement

C. Searching Effectiveness

In this section, we demonstrate the search effectiveness of different search algorithms, *i.e.*, whether they can find a (near-)optimal end-to-end route in a reasonable time and how optimal the route is.

Metrics. We use the following metrics: 1. *Proportion of Effective Search*: It represents the proportion of intents where at least one policy-compliant end-to-end route is found before early termination. 2. *Proportion of Optimal Search*: It represents the number of intents for which the SDIBO finds the optimal policy-compliant end-to-end route, *i.e.*, the route has the same length as the one found by SPE if any.

Early Termination Thresholds. We use the same early termination thresholds as in the last section.

Results. Fig. 5 demonstrates the proportion of successful search of SPE and SDIBO with the aforementioned setting. We can see that within a given time scale (bounded by the thresholds), SDIBO can substantially improve the chance that a policy-compliant end-to-end route is found: 61% (SDIBO) to 24% (SPE), which yields a 2.5x improvement.

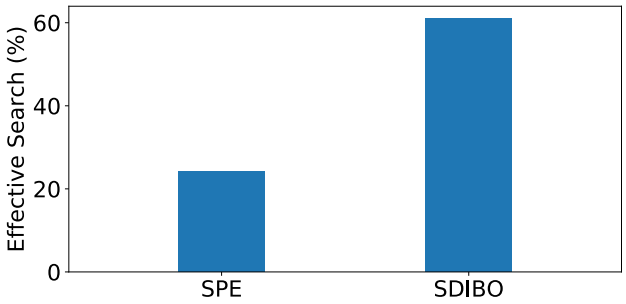


Fig. 5: Proportion of Effective Search.

We have also calculated the proportion of optimal search. Surprisingly, the proportion is 100%, *i.e.*, SDIBO is able to find a policy-compliant route with the same optimal length as SPE if any. While the result may be related to the selected objective function, this still demonstrates that **SDIBO has a very high probability to find an optimal policy-compliant end-to-end route within a reasonable time.**

VI. Related Work

BGP mechanisms for end-to-end route control. The main mechanisms in BGP to support end-to-end route control are selective announcement [29], AS path prepending [30], BGP FlowSpec [2] and BGP communities [31]. However, they are limited due to the destination-based forwarding nature and the peering model of BGP. Selective announcement, AS path prepending and BGP FlowSpec are mainly used for an AS to control its inbound traffic. And BGP communities only enable the interaction between two peering ASes. As illustrated in the motivating example in Section I, BGP does not support a source AS or end user to affect the route selection of remote ASes.

Recent interdomain routing protocols and systems. Many interdomain routing protocols [32]–[38] and systems [3]–[7], [9], [10], [39], [40] have been proposed to provide more mechanisms and interfaces for end-to-end route control. Their design can be categorized into three classes. The first one is third-party composition [3]–[7], [41]. An important representative system of this category is SDX [5] and its variants [6], [7], [41]. A major limitation of SDX is that a client's routing control actions are only used within the exchange point without being announced to other ASes. Hence, enforcing clients' routing policies at different SDXes can cause correctness issues such as persistent forwarding loops. To resolve this, forwarding loop detection approaches (*e.g.*, [7], [41]) have been proposed, but they still have a problem with a large number of false-positive alerts.

The second category is tunnel-based overlay [9], [10], [42]–[44]. MIRO [9], ARROW [10] and RCS [43] are the most recent systems in this category. The basic idea is to let a stub AS interact with a remote AS to select routes different from the BGP route, and then build a tunnel between stub and remote ASes to utilize the negotiated routes. As such, these systems have datapath overhead such as tunneling processing on each data packet. In addition, to ensure the convergence of interdomain routing, tunnels built and used in these systems are not announced to other ASes. This use-announcement inconsistency is usually not preferred by network operators for security reasons, *e.g.*, violation of data traffic regulation may happen without being detected.

Third, Google and Facebook [39], [40] also develop flexible peering systems to take route selection back to edge by overriding the BGP. However, these systems can only control the next-hop AS of traffic forwarding.

In contrast, in this paper, we propose SDI, a novel, systematic, low datapath overhead route control model, in which a network exposes a programmable interface to allow clients to define the interdomian routes of the network.

Bayesian optimization. Our blackbox optimization algorithm is built on the Bayesian optimization framework [15], [16], [19]. Bayesian optimization is a powerful framework for optimizing objective functions that are expensive to evaluate. However, when the decision variables are discrete, it is shown that the commonly rounding approach has a negative impact on the efficiency of BO [19]. A recent work proposes to use the graph to encode the discrete search space and applies spectral graph theory to solve the corresponding expected improvement maximization problem [45]. However, this design has severe scalability issues. In contrast, we leverage important properties from interdomain routing algebra to derive an accurate estimation of the expected improvement of an end-to-end route. As demonstrated in Section V, our EI estimation is accurate and fast even in large networks.

Routing algebra. Routing algebra has been used to analyze the stability, correctness and optimality of routing protocols [17], [18], [21], [34]. To the best of our knowledge, we are the first to leverage properties derived from interdomain routing algebra to improve the efficiency of blackbox end-to-end route optimization.

VII. Conclusion

In this paper, we provide the first, systematic formulation of the software-defined internetworking model, in which a network exposes a programmable interface to allow clients to define the interdomain routes of the network, but still maintains the control of its export policies. We define the optimal end-to-end SDI routing problem, prove its NP-hardness, and develop a blackbox optimization algorithm to efficiently find the near-optimal, policy-compliant end-to-end route with a small number of sample routes. We evaluate our algorithm extensively using real interdomain network topologies to demonstrate its efficiency and efficacy.

Acknowledgment

The authors thank Shenshen Chen, Haizhou Du, Linghe Kong, Alan Liu, Ennan Zhai and Yan Zhu for their help during the preparation of this paper. The authors also thank the anonymous reviewers for their valuable comments. This research is supported in part by NSFC grants #61702373, #61672385, #61701347, and #61902266; NSF award #1440745; Facebook Research Award, Google Research Award, and the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001.

REFERENCES

[1] Y. Rekhter, S. Hares, and D. T. Li, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271, Jan. 2006. [Online]. Available: https://rfc-editor.org/rfc/rfc4271.txt

[2] P. R. Marques, J. Mauch, N. Sheth, B. Greene, R. Raszuk, and D. R. McPherson, "Dissemination of Flow Specification Rules," RFC 5575, Aug. 2009. [Online]. Available: https://rfc-editor.org/rfc/rfc5575.txt

[3] V. Kotronis, X. Dimitropoulos, and B. Ager, "Outsourcing the routing control logic: better internet routing based on sdn principles," in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*. ACM, 2012, pp. 55–60.

[4] K. Lakshminarayanan, I. Stoica, S. Shenker, and J. Rexford, *Routing as a Service*. Computer Science Division, University of California Berkeley, 2004.

[5] A. Gupta, L. Vanbever, M. Shahbaz, S. P. D. B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett, "SDX: A Software Defined Internet Exchange," in *Proceedings og SIGCOMM 2014*. IEEE, August 2014, pp. 233–239.

[6] A. Gupta, R. MacDavid, R. Birkner, M. Canini, N. Feamster, J. Rexford, and L. Vanbever, "An industrial-scale software defined internet exchange point." in *NSDI*, vol. 16, 2016, pp. 1–14.

[7] R. Birkner, A. Gupta, N. Feamster, and L. Vanbever, "Sdx-based flexibility or internet correctness?: Pick two!" in *Proceedings of the Symposium on SDN Research*. ACM, 2017, pp. 1–7.

[8] G. Asharov, D. Demmler, M. Schapira, T. Schneider, G. Segev, S. Shenker, and M. Zohner, "Privacy-preserving interdomain routing at internet scale," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 3, pp. 147–167, 2017.

[9] W. Xu and J. Rexford, "Multi-path interdomain routing," in *In SIGCOMM*. Citeseer, 2006.

[10] S. Peter, U. Javed, Q. Zhang, D. Woos, T. Anderson, and A. Krishnamurthy, "One tunnel is (often) enough," in *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4. ACM, 2014, pp. 99–110.

[11] D. N. Bauer, D. Dechouniotis, C.-X. Dimitropoulos, and A. Kind, "Valley-free shortest path method," Mar. 15 2011, uS Patent 7,907,596.

[12] Z. M. Mao, L. Qiu, J. Wang, and Y. Zhang, "On as-level path inference," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1. ACM, 2005, pp. 339–349.

[13] L. Gao and J. Rexford, "Stable internet routing without global coordination," *IEEE/ACM Transactions on Networking (TON)*, vol. 9, no. 6, pp. 681–692, 2001.

[14] P. Gill, M. Schapira, and S. Goldberg, "A survey of interdomain routing policies." *Computer Communication Review*, vol. 44, no. 1, pp. 28–34, 2014.

[15] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[16] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[17] T. G. Griffin and J. L. Sobrinho, "Metarouting," in *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4. ACM, 2005, pp. 1–12.

[18] J. L. Sobrinho, "Network routing with path vector protocols: Theory and applications," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2003, pp. 49–60.

[19] E. C. Garrido-Merchán and D. Hernández-Lobato, "Dealing with integer-valued variables in bayesian optimization with gaussian processes," *arXiv preprint arXiv:1706.03673*, 2017.

[20] T. G. Griffin, F. B. Shepherd, and G. Wilfong, "The stable paths problem and interdomain routing," *IEEE/ACM Transactions on Networking (ToN)*, vol. 10, no. 2, pp. 232–243, 2002.

[21] J. L. Sobrinho, "Algebra and algorithms for qos path computation and hop-by-hop routing in the internet," in *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No. 01CH37213)*, vol. 2. IEEE, 2001, pp. 727–735.

[22] C. Bessiere, F. Koriche, N. Lazaar, and B. O'Sullivan, "Constraint acquisition," *Artificial Intelligence*, vol. 244, pp. 315 – 342, 2017, combining Constraint Solving with Mining and Learning. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0004370215001162

[23] J. Y. Yen, "Finding the k shortest loopless paths in a network," *management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[24] O. Alipourfard, H. H. Liu, J. Chen, S. Venkataraman, M. Yu, and M. Zhang, "Cherrypick: Adaptively unearthing the best cloud configurations for big data analytics," in *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*, 2017, pp. 469–482.

[25] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[26] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[27] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, "Bayesian optimization with inequality constraints." in *ICML*, 2014, pp. 937–945.

[28] CAIDA, "As relationships and internet traffic dataset," http://www.caida.org/data/as-relationships/, 2016.

[29] B. Quoitin, C. Pelsser, L. Swinnen, O. Bonaventure, and S. Uhlig, "Interdomain traffic engineering with bgp," *IEEE Communications magazine*, vol. 41, no. 5, pp. 122–128, 2003.

[30] R. K. Chang and M. Lo, "Inbound traffic engineering for multihomed ass using as path prepending," *IEEE network*, vol. 19, no. 2, pp. 18–25, 2005.

[31] B. Donnet and O. Bonaventure, "On bgp communities," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 55–59, 2008.

[32] R. R. Sambasivan, D. Tran-Lam, A. Akella, and P. Steenkiste, "Bootstrapping evolvability for inter-domain routing with d-bgp," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 2017, pp. 474–487.

[33] N. Kushman, S. Kandula, D. Katabi, and B. M. Maggs, "R-bgp: Staying connected in a connected world." USENIX, 2007.

[34] J. L. Sobrinho, D. Fialho, and P. Mateus, "Stabilizing bgp through distributed elimination of recurrent routing loops," in *Network Protocols (ICNP), 2017 IEEE 25th International Conference on*. IEEE, 2017, pp. 1–10.

[35] T. Holterbach, S. Vissicchio, A. Dainotti, and L. Vanbever, "Swift: Predictive fast reroute," in *Proceedings of the 2017 conference on ACM SIGCOMM 2017 Conference*. ACM, 2017, pp. 460–473.

[36] R. Mahajan, D. Wetherall, and T. E. Anderson, "Mutually controlled routing with independent isps." in *NSDI*, 2007.

[37] P. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet routing," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 111–122, 2009.

[38] R. Chandra, Y. Rekhter, T. J. Bates, and D. Katz, "Multiprotocol Extensions for BGP-4," RFC 4760, Jan. 2007. [Online]. Available: https://rfc-editor.org/rfc/rfc4760.txt

[39] B. Schlinker, H. Kim, T. Cui, E. Katz-Bassett, H. V. Madhyastha, I. Cunha, J. Quinn, S. Hasan, P. Lapukhov, and H. Zeng, "Engineering egress with edge fabric: steering oceans of content to the world," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 2017, pp. 418–431.

[40] K.-K. Yap, M. Motiwala, J. Rahe, S. Padgett, M. Holliman, G. Baldus, M. Hines, T. Kim, A. Narayanan, A. Jain *et al.*, "Taking the edge off with espresso: Scale, reliability and programmability for global internet peering," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 2017, pp. 432–445.

[41] A. Dethise, M. Chiesa, and M. Canini, "Prelude: Ensuring interdomain loop-freedom in˜ sdn-enabled networks," *arXiv preprint arXiv:1806.09566*, 2018.

[42] X. Yang, D. Clark, and A. W. Berger, "Nira: a new inter-domain routing architecture," *IEEE/ACM Transactions on Networking (ToN)*, vol. 15, no. 4, pp. 775–788, 2007.

[43] Y. Wang, J. Bi, and K. Zhang, "A sdn-based framework for fine-grained inter-domain routing diversity," *Mobile Networks and Applications*, vol. 22, no. 5, pp. 906–917, 2017.

[44] H. Wang, Y. R. Yang, P. H. Liu, J. Wang, A. Gerber, and A. Greenberg, "Reliability as an interdomain service," in *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4. ACM, 2007, pp. 229–240.

[45] C. Oh, J. M. Tomczak, E. Gavves, and M. Welling, "Combinatorial bayesian optimization using graph representations," *arXiv preprint arXiv:1902.00448*, 2019.