Accelerating BGP Configuration Verification through Reducing Cycles in SMT Constraints

Xiaozhe Shao, Zibin Chen, Daniel Holcomb, Lixin Gao, *Fellow, IEEE, ACM*

Abstract—Network verification has been proposed to help network operators eliminate the outage or security issues caused by misconfigurations. Recent studies have proposed SMT-based approaches to verify network properties with respect to network configurations. These approaches translate the verification problem into a Satisfiability Module Theories (SMT) problem. Although these approaches are attractive because of their broad coverage, they can scale to moderate-size networks only.
In this paper, we propose BiNode to accelerate the network verification process. The key idea is to formulate the SMT constraints in a manner that reduces/eliminates the cyclic dependencies between its variables. By doing so, we expedite the solving of the SMT problem. We implement and evaluate the performance of BiNode through an off-the-shelf SMT solver. The experimental results show that BiNode can reduce verification time by an order of magnitude through reducing/eliminating the cyclic dependencies among SMT variables.

Index Terms—Routing Verification, SMT, Network Management

1 Introduction

Network operators typically set configurations manually, and therefore, misconfigurations can occur. Misconfigurations can be costly; they can not only cause downtime in critical systems [11], [28], [36] but also lead to security breaches [37]. Study shows that in 2016 and 2017, 66% of network incidents in Alibaba were due to network misconfigurations [25]. Network verification can enable network operators to check properties and identify configuration bugs.

Data plane verifications [19], [23], [35], [44] take one snapshot from the routing information base and verify properties only on that particular snapshot. Data plane verification is fast. However, networks are dynamic systems where links can be down. Also, peering Autonomous Systems (ASes) might add/withdraw routes to a particular destination by sending announcements via eBGP sessions. These routing dynamics can change the data plane state and invalidate any previously verified properties. The properties must be reverified for every change in the data plane state.

Control plane verifications [1], [5], [14], [16] analyze the configuration files and verify network properties under any possible data plane states generated by the configuration files. Satisfiability Modulo Theories (SMT) based approach [39] translates verification problems into SMT constraints and verify properties using general-purpose SMT solvers. One approach to construct the constraints is to describe the constraints that best routes should satisfy for each router. However, SMT-based methods scale to only moderate size networks.

In this paper, we propose a scalable SMT-based scheme for network control plane verification named BiNode. BiNode accelerates the verification system by reducing the dependency cycles among SMT variables. BiNode achieves cycle-free by incorporating the knowledge of the routing convergence process. In other words, we explicitly account for the route decision process when constructing the SMT constraints.

BiNode can accelerate the verification process through reducing the dependency cycles among SMT variables. BiNode can be applied to one single ISP network where iBGP is the most widely used routing protocol to redistribute routes received from outside networks. Networks consisting of multiple ASes can further benefit from BiNode where eBGP is the default routing protocol used to exchange routing information among ASes. BiNode can be also applied to modern data center networks, which use eBGP as the internal routing protocol [10].

This paper constructs BiNode in the following way. Firstly, we explicitly represent the interaction between iBGP neighbors to reduce the mutual dependencies caused by iBGP sessions. For example, we can avoid mutual dependencies between SMT variables of two peering iBGP routers by exploiting the export rule which only routes learned from eBGP neighbors can be announced to iBGP neighbors. Then, we can explicitly represent the setting of the routing policies between eBGP neighbors in SMT constraints. Finally, when multiple ASes owned by an organization are verified, variables for a pair of eBGP neighbors might depend on each other. In our approach, we explicitly represent the route announcements between the neighboring eBGP routers so that the mutual dependencies between variables of these eBGP sessions are avoided. For example, we can explicitly

The preliminary version of this article titled "Verifying Policy-based Routing at Internet Scale" was published in the Proceedings of the 39th IEEE International Conference on Computer Communications (INFOCOM), Virtual Conference, July 2020. [32]
Xiaozhe Shao is with the Meta Platforms, Inc, USA (e-mail: xiaozheshao@umass.edu). This research were done while the author was a student at University of Massachusetts, Amherst, USA.
Zibin Chen is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, USA (e-mail: zibinchen@umass.edu).
Daniel Holcomb is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, USA (e-mail: dholcomb@umass.edu).
Lixin Gao is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, USA (e-mail: lgao@umass.edu).

represent the export policy that a router announces only the customer (not peer or provider) routes to eBGP neighbors in peer or provider ASes.

We implement the prototype of BiNode and use an off-the-shelf SMT solver, Z3 [9] to solve the SMT constraints. We evaluate the performance of BiNode through verifying network properties, such as reachability property, bounded length property, and equal path length property, on router configurations of networks. The experimental results show that BiNode can reduce the verification time by an order of magnitude. We also compare BiNode with the state-of-the-art control plane verification approach. The experiment results show that BiNode can achieve on average 40% speedup over the state-of-the-art control plane verification approach.

The contributions of this paper are summarized as follows. In this paper, we propose BiNode, an SMT-based approach to verify network properties on control plane. BiNode avoids cyclic dependencies among SMT variables and accelerates the verification process by an order of magnitude. We elaborate on the application of BiNode to various types of networks. In this paper, we show the application of BiNode to single ISP networks, multiple networks, and data center networks.

The rest of this paper is organized as follows. Section 2 discuss the naive SMT-based approach for control plane verification. We discuss the limitations of the naive SMT-based approach and motivate our work. Then we introduce the application of BiNode to single ISP networks, multiple networks, and data center networks in Section 3, Section 4, and Section 5, respectively. We evaluate BiNode in Section 6 and discuss the related work in Section 7. We conclude in Section 8.

2 SMT-Based Approach for Control Plane Verification

To verify network configurations based on SMT, a system of SMT constraints is generated from the routing configurations. Routing configurations are encoded into SMT constraints, describing the relationships between the configurations and the data plane forwarding state. This section first describes the general SMT-based approach for network verification used in Minesweeper [5] and Bagpipe [41]. We refer to the approach as the *topology-based approach*. After that, we discuss the major limitations of the topology-based approach and motivate our work.

2.1 Encoding Routes

We consider two types of routes: candidate routes and best routes. *Candidate routes* are the routes learned from peering neighbors after applying the corresponding import policies. Routing protocols pick the best route from candidate routes according to the preference.

We use $best_i$ to represent the best route of router i. Candidate routes learned from peering neighbors are denoted with the ID of two routers. More specifically, the candidate route of R_i[1] learned from R_j is represented by $C_{i,j}$.

1. In this paper, we use R with subscript to denote a router. For example, R_1 represent the first router in a network.

TABLE 1: Symbolic variables in BiNode.

Variable	Description	Encoding
$r.prefix$	Destination ip prefix for r	[0,2^32)
$r.prefix_length$	Prefix length for r	[1,32]
$r.next_hop$	The next hop router ID of r	[0,2^32)
$r.cost$	Shortest path cost of r	[0,2^16)
$r.length$	AS path length for r	[0,2^16)
$r.pref$	Local preference for r	[0,2^16)
$r.origin$	Origin Type of r	[0,2^2)
$r.med$	BGP MED attribute for r	[0,2^32)
$r.distance$	IGP distance of r	[0,2^32)
$r.rid$	Neighbor router ID for r	[0,2^32)
$r.ibgp$	Whether r is learned via iBGP	1 bit

Routes are encoded into SMT with a set of integers where each integer corresponds to one attribute of the route. Table 1 lists the symbolic variables that can be translated into SMT variables representing the basic information within a route.

2.2 Encoding Route Exchanges

Route exchanges between two routers consist of two stages: export and import. In the export stage, routers can apply export policies to decide whether to announce a route to their neighbors. Then, in the import stage, routers can decide whether to accept the routes in announcements according to the import policies. The import and export policies can be represented with import and export function, respectively. We use f_{exp} and f_{imp} to denote the import function and export function. We use the transformation function (denoted as g) to represent the route exchange process. The transformation function is a mapping between one route to another. g is essentially the composition of f_{imp} and f_{exp}. More specifically, the route exchange process can be represented with the following equation.

$$C_{i,j} = g_{i,j}(best_j) = f^{i,j}_{\text{imp}}(f^{j,i}_{\text{exp}}(best_j)) \quad (1)$$

where $f^{i,j}_{\text{imp}}$ is the import function representing the import policy deployed on R_i for R_j and $f^{j,i}_{\text{exp}}$ is the export function representing the export policy deployed on R_j for R_i.

In general, import and export functions are translated into constant number of SMT constraints depending on the number of attributes the corresponding policy sets. Different policies overwrite attributes in different ways. We capture this by encoding import functions and export functions into SMT constraints where corresponding attributes are overwritten.

As an example, consider that R_1 in AS 45000 might have the following configuration to set local-preference 100 for routes learned from BGP neighbors in AS 50000. Below is a snippet of the configuration on R_1.

Listing 1: A snippet of the configuration on R_1

```
router bgp 45000
   neighbor 192.168.1.3 remote-as 50000
   neighbor 192.168.1.3 route-map
       SET-LOCAL-PREF-PEER in
!
route-map SET-LOCAL-PREF-PEER
   set local-preference 100
```

Suppose the peering BGP router in AS 50000 is denoted as R_2. In this example, the import filter can be encoded as Listing 2.

Listing 2: Translation of import filter

```
C1,2.local_preference = 100
C1,2.length = best2.length + 1
C1,2.next_hop = 2
Ci,2.rank = 100
```

Specially, when export policies do not announce a route to a neighbor, the corresponding export function returns $null$. Any import functions applied to $null$ will return $null$ as well.

2.3 Encoding Route Selection

Routers choose the best route from a set of candidate routes. We use *selection function* to represent the route selection process. We use f_{sel} to denote the selection function. The selection function describes the relationship between the candidate routes and the best route, i.e., the best route is the most preferred candidate route.

$$best_i = f_{\text{sel}}(\{C_{i,j} | R_j \in N(R_i)\}) \tag{2}$$

where $N(R_i)$ denotes the set of peering neighbors of R_i.

We use an integer attribute to represent the level of preference by a protocol at a specific router, namely, *ranking*. The ranking of a candidate route C is assigned at the import stage by the router receiving it. We use $C.rank$ to denote the ranking of C. The ranking typically relates to the attributes of the route. For example, in BGP, the best route is selected by comparing a sequence of attributes such as local preference and AS path length. Therefore, the selection function selects the routes with the highest ranking. That is,

$$f_{\text{sel}}(\{C_{i,j} | R_j \in N(R_i)\}) = \underset{C_{i,j:R_j \in N(R_i)}}{\arg\max} \left(C_{i,j}.rank\right) \tag{3}$$

We encode the selection function into SMT constraints as follows.

$$\begin{aligned} &\bigwedge_{R_j \in N(R_i)} best_i.rank \leq C_{i,j}.rank \\ &\wedge \bigvee_{R_j \in N(R_i)} best_i.rank = C_{i,j}.rank \end{aligned} \tag{4}$$

2.4 Encode Network Properties

In order to verify network properties with SMT, an additional set of constraints representing the properties need to be generated. At this point, we can describe the relationship between the routing table entries and the network configurations. We generate the constraints that describe what kind of routing table entries *violate* the properties. For example, if we want to verify the traffic should always be sent to R_0 at R_1, we have the following constraint.

$$\neg(best_1.next_hop = 0) \tag{5}$$

which is

$$best_1.next_hop \neq 0 \tag{6}$$

After that, we take the *AND* of the constraints encoded from the configurations and the constraints encoded from the network properties. We use an SMT solver to solve the constraints. The satisfiable assignments from the SMT solver are counter-examples to the network properties. As a result, whenever the SMT solver gets a satisfiable assignment, the network properties do *NOT* hold. If we got an *unsat*, we say the properties hold.

(a) A topology example. (b) Variable dependency graph for (a).

Fig. 1: The topology-based approach and its variable dependency graph.

2.5 Limitation of the Topology-Based Approach

Solving constraints for the topology-based approach can be slow because of the cyclic dependencies among variables. Cyclic dependencies between variables happen in the constraints encoded from the route exchange processes. We refer to *variable a depends on variable b* as only when b has a satisfiable assignment, a can have a satisfiable assignment. Obviously, when it is possible for R_j to announce its best route to its neighbor R_i, $best_i$ cannot be determined until $best_j$ is determined. That is, $best_i$ depends on $best_j$. Dependencies happen in both directions when R_i and R_j exchange routes.

Cyclic dependencies in SMT constraints can cause the solving process to be time-consuming. Intuitively, cyclic dependencies make the relationships among variables much more complex and slow down the solving process. When solving the SMT problem, the SMT solver translates the SMT problem into a boolean SAT problem and then rewrites the boolean expression into conjunctive normal form (CNF). CNF is the product (AND operation) of a set of clauses where each clause is a sum of boolean variables (OR operation). The underlying SAT solver tries to assign values for boolean variables such that each clause results in a 1 (true). When the SMT constraints are acyclic, each constraint can be written into equality form with only one unknown variable at each clause. So the SAT solver can easily find assignments for the boolean variables that make all clauses to be 1.

We use a directed graph to visually show the variable dependencies in the SMT constraints encoded from an SMT-based approach. We refer to this graph as *Variable Dependency Graph*. A node in the variable dependency graph is the best route of a router. We build a directed edge from $best_j$ to $best_i$ if $best_i$ depends on $best_j$.

Considering a topology with 3 routers shown in Figure 1(a), where each node is a router, a link between two nodes represents a BGP session between the corresponding routers. Figure 1(b) illustrates the variable dependency graph for the topology-based approach. As we can see from the figure, $best_1$ and $best_2$ mutually depend on each other, and cyclic dependencies exist between $best_1$ and $best_2$.

More generally, if there are cycles in the dependency graph, the dependencies among those variables are cyclic. We refer to the cycles as variable dependency cycles.

Inspired by this, we propose BiNode by reducing/eliminating variable dependency cycles in SMT constraints. BiNode eliminates/reduces the variable dependency cycles described above by considering the routing convergence

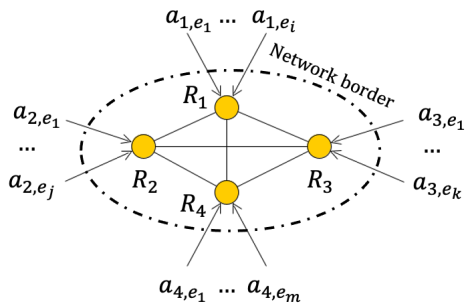


Fig. 2: Fully meshed iBGP architecture.

process and making the protocol rules and the routing policies explicit. In the rest of this paper, we demonstrate the application of BiNode to single ISP networks, multiple networks, and data center networks in Section 3, Section 4, and Section 5, respectively.

3 BiNode for Single ISP Network

In a single ISP network, iBGP is widely used to redistribute routes learned from external networks to the internal. In this section, we show the application of BiNode to single ISP networks. As the name suggests, the key idea of BiNode is to represent each router into two nodes. By introducing an extra node, we are able to eliminate the circular dependencies in the topology-based approach.

To apply BiNode to one single ISP network, we need to encode the iBGP configuration into an SMT problem without variable dependency cycles. In this section, we first introduce the background of iBGP routing protocol and then discuss the application of BiNode to two widely used architectures in ISP networks.

3.1 Background of iBGP Routing System

To avoid blackholes in the routing system, there are two commonly used iBGP configurations: fully meshed iBGP and iBGP with route reflection. We provide the background of these two configurations in the rest of this section.

A fully meshed iBGP routing system redistributes routes received from external networks by establishing iBGP sessions between every pair of BGP routers. Figure 2 illustrates a network with a fully meshed iBGP structure. $a_{p,e}$ represents the route announcements received by R_p from its external BGP peer R_e. Each router only announces the routes learned from eBGP neighbors to iBGP neighbors.

A fully meshed iBGP routing system can end up with a large number of iBGP sessions as the number of BGP routers grows. Route reflection [8] is proposed to limit the number of iBGP sessions while keeping the connectivity.

Route reflection is a hierarchical structure and is divided into clusters. In the hierarchical structure, some BGP routers work as route reflectors (RR) and serve a set of BGP routers called clients. They establish route reflector to client sessions where the clients are considered lower tier. RRs are responsible for redistributing routes collected from clients to other iBGP neighbors and announcing its best routes to clients. In real networks, routers located in the same Point-of-Presence (PoP) [13] form a cluster and redistribute routes to other clients through top-tier routers. Routers in the iBGP routing

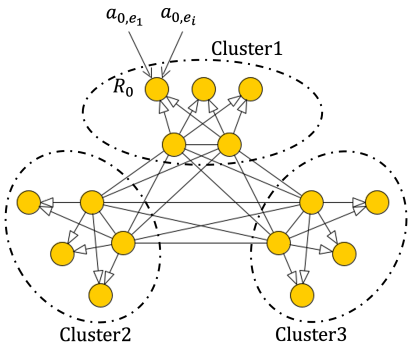


Fig. 3: An iBGP routing system with route reflection.

system with route reflection redistribute routes follow the following rules.

- If the best route is received from non-client iBGP neighbors, announce it to clients neighbors and eBGP neighbors;
- If the best route is received from client iBGP neighbors, announce it to all neighbors except the originator of the route;
- If the best route is received from eBGP neighbors, announce it to all neighbors.

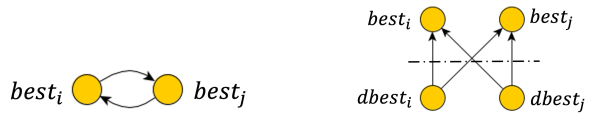
Figure 3 illustrates a simple network divided into three clusters with a two-tier route reflection. Route-reflector-to-client sessions are only established within one cluster, while non-client sessions can be either within one cluster (i.e., connecting with same-tier RRs) or cross clusters (i.e., establishing fully meshed non-client sessions among top tier RRs). Each tier within one cluster might have more than one BGP routers for backup or load balancing purposes. BGP routers at the outmost[2] layer establish eBGP sessions. BGP routers at the top tier are supposed to connect in a fully meshed manner.

3.2 BiNode for Fully Meshed iBGP Routing Systems

As mentioned before, BiNode tries to eliminate the variable dependency cycles in SMT constraints. To do so, we first analyze the dependencies among variables in the topology-based approach. As mentioned in Section 2.5, there are cyclic dependencies between every pair of $best$ variables. However, as per the iBGP, BGP routers announce their best routes to iBGP neighbors only if the best route is learned from eBGP sessions. In other words, for an iBGP session established between R_i and R_j, $best_i$ depends on $best_j$ only if $best_j$ is learned via eBGP sessions. Intuitively, we can represent R_p with two nodes: best routes learned from eBGP sessions (we call it $dbest_p$) and the overall best route of R_p (we call it $best_p$). To this end, instead of having $best_i$ depends on $best_j$, we have $best_i$ depends on $dbest_j$. By representing each router with two nodes, BiNode breaks the cyclic dependencies based on the origin of the routes while still correctly representing the interaction between best routes.

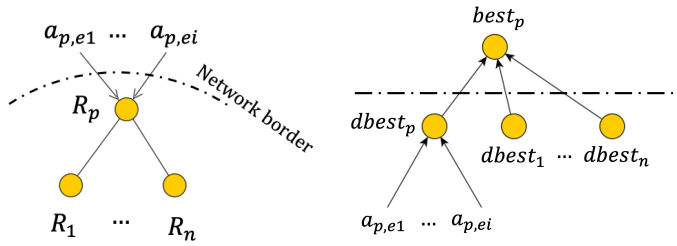
To explicitly show that BiNode indeed eliminates the variable dependency cycles between any pair of BGP routers

2. We refer to the route reflectors connecting across clusters as *top-tier* routers and those who don't have any clients as *outmost* layer routers.



(a) Variabel dependency graph for the topology-based approach. (b) Variable dependency graph for BiNode.

Fig. 4: BiNode avoid variable dependency cycles for fully meshed iBGP architecture.



(a) R_p and its BGP neighbors. (b) Dependency graph of (a).

Fig. 5: BiNode for R_p in a fully meshed iBGP system.

in a fully meshed iBGP system, we show the variable dependency graph for both BiNode and the topology-based approach in Figure 4. Figure 4(a) is the variable dependency graph for the topology-based approach and Figure 4(b) is the corresponding variable dependency graph in BiNode. As we can see from the figure, the dependency graph for the topology-based approach has dependency cycles between $best_i$ and $best_j$ while there is no dependency cycle in BiNode. More generally, consider any R_p in a fully meshed iBGP routing system (as shown in Figure 5,(a)), Figure 5(b) is the variable dependency graph in BiNode near R_p.

3.2.1 SMT Encoding of BiNode for Fully Meshed iBGP

In this section, we derive SMT constraints from variable dependency graph of BiNode. Intuitively, nodes in the graph can be encoded into SMT variables and directed edges can be encoded into SMT constraints. More specifically, $best_p$ in the graph can be encoded into the set of SMT variables in Table 1, and a directed edge from node $dbest_q$ to $best_p$ can be encoded into constraint shown in Equation (1). This is also true for $dbest$ nodes. We extend the candidate routes for $dbest_p$ as all routes learned via eBGP sessions after applying import policies.

More generally, generating constraints for any R_p consists of two steps: (1) generating constraints for $dbest_p$ and (2) generating constraints for $best_p$. The corresponding constraints are as follows.

Generating constraints for $dbest_p$:

$$dbest_p = f_{\text{sel}}(\{C_{p,e} | R_e \in N_e(R_p)\}) \tag{7}$$

where $N_e(R_p)$ represents the set of eBGP neighbors of R_p. $C_{p,e}$ is the candidate route learned from external peer R_e. That is, $C_{p,e}$ can be encoded with Equation (8).

$$C_{p,e} = f^{p,e}_{\text{imp}}(a_{p,e}) \tag{8}$$

Note that f_{sel} in BiNode is exactly the same as that in the topology-based approach, since it is determined by BGP. The selection function can be encoded into SMT constraints with Equation (4).

Generating constraints for $best_p$:

$$best_p = f_{\text{sel}}(\{dbest_p\} \cup \{C_{p,q} | R_p \in N_i(R_q)\}) \tag{9}$$

where $N_i(R_p)$ represents the set of iBGP peers of R_p, i.e., $1 \leq q \leq n, q \neq p$. However, as described above, only routes learned from eBGP neighbors can be redistributed to iBGP neighbors. As a result, different from the topology-based approach, the candidate route is the transformation function applied to $dbest_q$. This relationship can be expressed with Equation (10).

$$C_{p,q} = g_{p,q}(dbest_q) \tag{10}$$

3.2.2 Dependency Graph Representation of BiNode

Now we introduce the Dependency Graph representation of BiNode. As we can see from the previous section, the dependency graph can be translated into SMT constraints and get the problem solved by a general-purpose SMT solver. The dependency graph is a more visually-friendly representation of BiNode.

Now we describe a general approach to deriving SMT constraints from the dependency graph of BiNode. With that, we can visually illustrate BiNode using its dependency graph. We apply similar strategy used in Section 3.2.1. We translate each node in the dependency graph into the constraints of selection functions and transformation functions. $dbest$ from the router can be directly used as candidate routes. More specifically,

$$\forall r \in \mathbf{N}, r = f_{\text{sel}}(\{G_{r,m}(m) | m \in Incoming(r)\}) \tag{11}$$

where $\mathbf{N}$ represents the set of nodes in dependency graph and $Incoming(r)$ represents the set of nodes that have a direct edge to node r. G function can be expressed with the following equation.

$$G_{r,m}(m) = \begin{cases} m & \text{if } id(r) = id(m) \\ g_{id(r),id(m)}(m) & \text{otherwise} \end{cases} \tag{12}$$

where $id(r)$ is the router id of the router where r representing. g is the transformation function described in Section 2.

BiNode is Equivalent to the Topology-Based Approach

For any fully meshed iBGP routing system, Theorem 3.1 states that BiNode is equivalent to the corresponding topology-based approach.

Theorem 3.1. *The best routes derived from BiNode are the same as that from the topology-based approach for any fully meshed iBGP system.*

The proof of Theorem 3.1 can be found in Appendix A.

3.3 BiNode for iBGP with Route Reflection

We now describe BiNode for networks with route reflection. Following the same intuition in Section 3.2, we make the importing and exporting rule in iBGP with route reflection explicit and represent each BGP router in iBGP with route reflection with two nodes. BGP routers in iBGP with route reflection only announce the route from an iBGP neighbor to another iBGP neighbor when the best route is received from clients. Note that BGP routers always announce best routes learned from eBGP neighbors, $best_i$ depends on $best_j$ only when R_j is a client neighbor or eBGP neighbor of R_i.

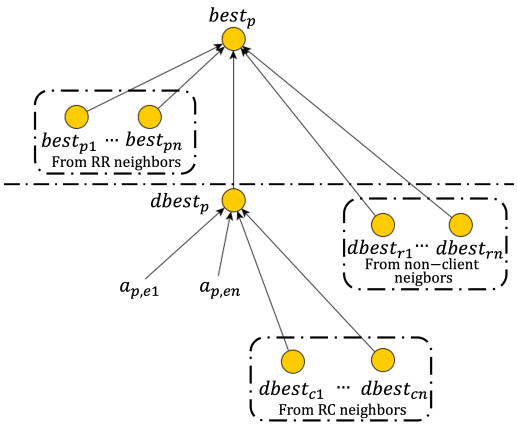


Fig. 6: Variable dependency graph for a BGP router in iBGP with route reflection.

We extend the concept of $dbest$ introduced in Section 3.2 for the hierarchical structure to the best route learned via lower-tier routers (as mentioned before, external routes are considered lower-tier routers). More precisely, for any BGP R_p, $dbest_p$ represents the best route learned from client neighbors and eBGP neighbors.

Following the route redistribution rule mentioned in Section 3.1, we build the variable dependency graph for BiNode for iBGP with route reflection in Figure 6. The figure shows that there is no variable dependency cycle in the variable dependency graph.

BiNode is Equivalent to the Topology-Based Approach

Theorem 3.2 states that BiNode is equivalent to the topology-based approach for any iBGP routing system with route reflection.

Theorem 3.2. *The best routes derived from BiNode are the same as that from the topology-based approach for any iBGP routing system with route reflection.*

We show the SMT constraints for iBGP with route reflection and prove Theorem 3.2 in Appendix B.

4 BiNode for Multiple Networks

A large enterprise/organization might own a number of Autonomous Systems (ASes) peering with each other. For example, Verizon owns 20 ASes, and AT&T owns 26 ASes [7]. Network operators might need to verify properties across these ASes. As a result, we need to translate the configurations of multiple ASes into an SMT problem. When considering multiple ASes, eBGP sessions are used to exchange information across ASes. However, in the topology-based approach, each eBGP session between two border routers can lead to the cyclic dependencies between the corresponding SMT variables for best routes. In this section, we first introduce the background of eBGP and then extend BiNode to eBGP sessions. We show BiNode can avoid the variable dependency cycles introduced by eBGP sessions. Finally, we consider both eBGP sessions and iBGP sessions together and introduce BiNode for multiple networks.

4.1 Background of eBGP

To begin with, we introduce the background of eBGP and the AS relationships in the hierarchical architecture. The network operators set up the configuration of border routers to exchange routing information with neighboring ASes by following the agreements established with the neighboring ASes. Provider-customer (PC) relationship and peer-peer (PP) relationship are two common agreements between ASes. Most ASes set up the routing configuration by following the Gao-Rexford guideline [15], which includes the following two rules.

- *Prefer-customer rule:* an AS prefers customer routes[3] over peer and provider routes. That is, whenever a customer route exists, the best route of an AS is always a customer route.
- *Valley-free rule:* peer and provider routes are exported to customer ASes only.

4.2 Applying BiNode to eBGP Sessions Only

In this section, we apply BiNode to eBGP sessions only and show that BiNode can eliminate the cyclic dependencies among SMT variables. We introduce BiNode for networks whose routing policies follow the Gao-Rexford guideline in this section. Some ASes might not follow the Gao-Rexford guideline [4], [17], [27], [30]. In Appendix C.1, we extend BiNode to accommodate the routing policies that violate the prefer-customer rule and valley-free rule. We consider BiNode at AS level (i.e., ignore iBGP sessions) when introducing BiNode for eBGP sessions. We introduce BiNode for multiple networks when considering iBGP and eBGP sessions together in the next section.

We analyze the dependencies among variables in the topology-based approach and apply BiNode to eBGP sessions. Similar to Section 3, the key idea of eliminating the cyclic dependencies caused by eBGP sessions is to represent each AS with two nodes. According to the prefer-customer and valley-free rule, customer routes always have a higher ranking than the other routes, and only best routes learned from customer ASes are announced to peer and provider ASes. By making this relationship explicit in BiNode, we can eliminate dependency cycles on networks with policies that follow the Gao-Rexford guideline.

We use two nodes to represent AS i in BiNode. Namely, $best_i$ and $pbest_i$, where $pbest_i$ represents the best customer routes of AS i. $pbest_i$, as defined, depends on best route of the customer ASes only. In the customer ASes, only customer routes are announced to AS i according to the valley-free rule. As a result, $pbest_i$ depends on the best customer routes of its customer ASes.

Similarly, $best_i$ depends on $pbest_i$ and the best routes from peer ASes and provider ASes. Peer ASes announce only their best customer routes to AS i, and provider ASes announce their best routes to AS i according to the valley-free rule.

In general, consider an AS peering with other ASes, Figure 7 illustrates the dependency graph around the variables

3. For simplicity, we name the route learned from customer ASes *customer routes*. Similarly, *peer routes* are referred to routes learned from peering ASes and *provider routes* are referred to routes learned from provider ASes.

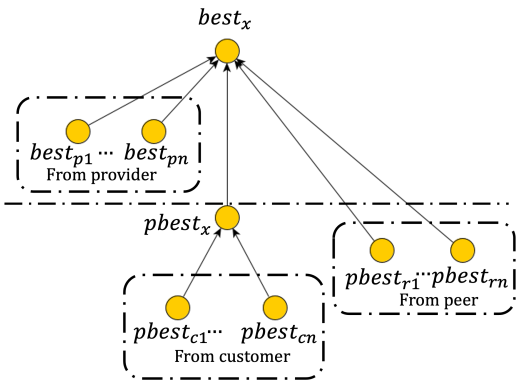


Fig. 7: Dependency graph around $best_p$ and $pbest_p$.

of an AS x, where $p1, ..., pn$ are providers of AS x, $r1, ..., rn$ are peers of x and $c1, ..., cn$ are customers of AS x. The best customer route of x, $pbest_x$, depends on the best customer routes of its customers, *i.e.*, $pbest_{c1}, \ldots, pbest_{cn}$, since the customers of x only announce customer routes to x. The best route of x, $best_x$, depends on the best routes of its providers, *i.e.*, $best_{p1}, \ldots, best_{pn}$, since the providers of x announce all routes to x. In addition, $best_x$ depends on the best customer routes of its peers, *i.e.*, $pbest_{r1}, \ldots, pbest_{rn}$, since the peers of x only announce customer routes to x.

BiNode is Equivalent to the Topology-Based Approach

Theorem 4.1 states that any routing system, when only considering eBGP sessions, BiNode is equivalent to the corresponding topology-based approach.

Theorem 4.1. *The best routes derived from BiNode are the same as that from the topology-based approach when only considering eBGP sessions.*

The proof of the theorem can be found in Appendix C. We show BiNode for ASes violating Gao-Rexford in Appendix C.1.

4.3 Applying BiNode to Both iBGP and eBGP Sessions

Now we apply BiNode to network with both eBGP sessions and iBGP sessions. When verifying properties on a set of connected ASes, it is not sufficient to apply BiNode to iBGP sessions only (Section 3) or eBGP sessions only (Section 4.2). Each of the iBGP and eBGP sessions along can lead to mutual dependencies. In Figure 8, we use a pair of connected ASes to show the mutual dependencies. As Figure 8(a) shows, there are two peering ASes where each AS has two routers. When we apply BiNode to iBGP sessions only, we have the dependency graph shown in Figure 8(b). A cycle is drawn in red among the variables, $best_2$, $dbest_3$, $best_3$ and $dbest_2$ (shown in red). Alternatively, when we apply BiNode to eBGP sessions only, we have the dependency graph shown in Figure 8(c). There are four cycles shown in red derived from iBGP sessions. To avoid these dependency cycles, we extend BiNode to accommodate both iBGP sessions and eBGP sessions as follows. As will describe in the following, to get rid of mutual dependencies, we represent the best route of a router with four nodes. We still name the proposed scheme BiNode as we can treat them as two group of nodes (which will discuss in this section).

4.3.1 Dependency Graph Construction

The basic idea is to use separate variables to represent the best routes among a set of preferred routes. More specifically, we use $pbest$ to present the best customer routes and $pdbest$ to present the best customer routes learned from eBGP sessions.

Mutual dependencies among eBGP routes in different networks can be eliminated by analyzing the relationships and using separate nodes to represent the more preferred routes. More specifically, let's consider the example shown in Figure 9(a). In this example, R_p is a BGP router running both iBGP and eBGP simultaneously. Among all eBGP sessions, $R_{c1}, \ldots, R_{ci}$ are eBGP neighbors in customer ASes, $R_{r1}, \ldots, R_{rk}$ are eBGP neighbors in peer ASes and $R_{p1}, \ldots, R_{pj}$ are eBGP neighbors in provider ASes. We first describe how to eliminate the mutual dependencies among eBGP sessions with customer ASes and then with provider/peer ASes. Customer ASes announce their best routes to R_p only if its best route is learned from their customer ASes. As a result, we introduce separate node $pbest_p$ to represent the best external route learned from customer ASes. However, for network with multiple eBGP speakers, introducing $pbest_p$ can result in mutual dependencies among $pbest$ nodes. To eliminate the variable dependency cycles among $pbest$ nodes, we further introduce node $pdbest_p$ of R_p representing the best route learned via eBGP sessions established with customer ASes.

In summary, for a BGP speaker R_p which establishes BGP sessions with both iBGP neighbors and eBGP neighbors, we use four nodes to represent its best route: $best_p, pbest_p, dbest_p$ and $pdbest_p$. $best_p$ and $dbest_p$ are the same as Section 3. $pbest_p$ represents the best route among customer ASes. $pdbest_p$ represents the best route learned from eBGP sessions established with external BGP speakers within a customer AS.

Now we describe the variable dependencies among the four variables mentioned above. $pdbest_p$ depends on $pbest$s of customer ASes since customer ASes announce their best routes learned from their customer ASes only. In our previous topology shown in Figure 9(a), $pdbest_p$ depends on $pbest_{c1}, \ldots, pbest_{ci}$. Similar to iBGP sessions, $pbest_p$ depends on $pdbest_p$ and $pdbest_1, \ldots, pdbest_k$ where $R_1, \ldots, R_k$ are the iBGP neighbors. $dbest_p$ depends on $pdbest_p$. Dependencies for $best_p$ is the same as that of dependencies discussed in Section 3. Figure 9(b) illustrates the dependency graph among variables for R_p's best route.[4]

BiNode is Equivalent to the Topology-Based Approach

Theorem 4.2 states that for any routing system, BiNode is equivalent to the corresponding topology-based approach.

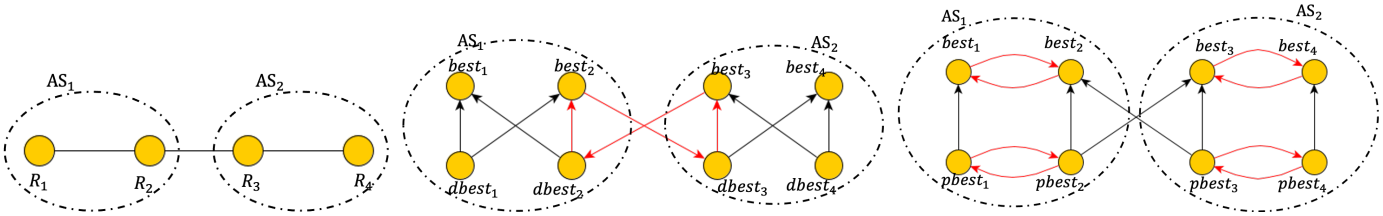
Theorem 4.2. *For any routing system, the best routes derived from BiNode are the same as that from the topology-based approach.*

The proof of the theorem can be found in Appendix D.

4.3.2 BiNode for An Example Topology

Given the configurations of all routers in a set of ASes, we can derive the dependencies of their variables. Then, we

4. We don't show dependencies related with $best_p$ for simplicity, please refer to Section 3.

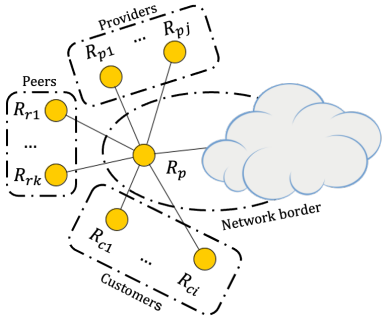


(a) A router-level topology containing two ASes with PP relations.

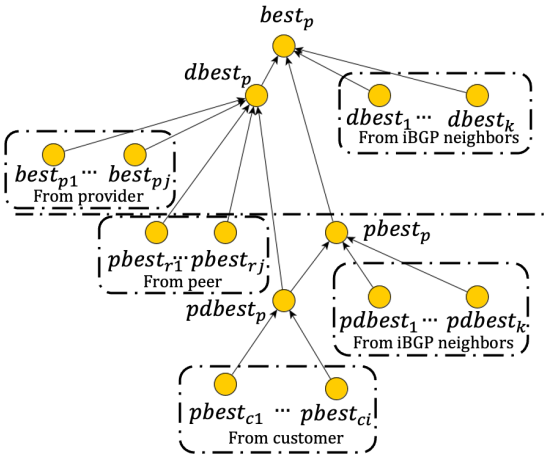
(b) Dependency graph when BiNode is applied to iBGP sessions only.

(c) Dependency graph when BiNode is applied to eBGP sessions only.

Fig. 8: Dependency graphs when BiNode is applied to iBGP sessions or eBGP sessions only.



(a) R_p and its neighbors.



(b) BiNode for R_p shown in (a).

Fig. 9: BiNode for BGP router with both iBGP sessions and eBGP sessions.

can compose all those dependencies into the dependency graph for all pairs of neighboring routers. For example, Figure 10 shows a router-level topology and the associated dependency graph when BiNode is applied.

5 BiNode for Data Center Networks

In this section, we describe BiNode for data center networks. Modern data center networks are designed to have certain characteristics such as scalability, resilience and extensive intra-data center communications [10]. Clos architecture [3], [33] is widely used in large companies' data center networks such as Google Data Center [20] and Facebook Data Center [2]. Figure 11 illustrates a three-tier Clos datacenter network.

Instead of having traditional intra-domain routing protocol, data center networks use eBGP as their internal routing protocol. The AS numbering scheme used inside data centers is typically private AS numbers. Routers within the data center form a hierarchical architecture. In Figure 11, we use directed arrows to represent the hierarchical structure, and arrows point to the lower-tier router. Route selections and redistributions within the data center follow the rules similar to the Gao-Rexford guideline described in Section 4. The cross-tier eBGP sessions can be mapped to Provider-customer relationships where higher-tier routers are considered the provider and lower-tier routers are considered the customer. The corresponding rules in data center networks are summarized as follows.

Network	Size (# of routers)
Data center networks	5-250
Single ISP (Fully meshed iBGP)	10-300
Multiple ISPs (Route reflection iBGP)	108-750

TABLE 2: Network configurations used for evaluation.

- *Prefer-lower-tier rule:* BGP routers prefer routes learned from lower-tier neighbors.
- *Valley-free rule:* best routes learned from higher-tier neighbors are exported to lower-tier neighbors only.

As a result, we can represent a BGP router within the data center with two nodes. We use $dbest_p$ to represent the best route learned from lower-tier neighbors and use $best_p$ to represent the best route of R_p. As a result, we can apply BiNode to data center networks by mapping the cross-tier relationships into Provider-customer relationships.

6 Evaluation

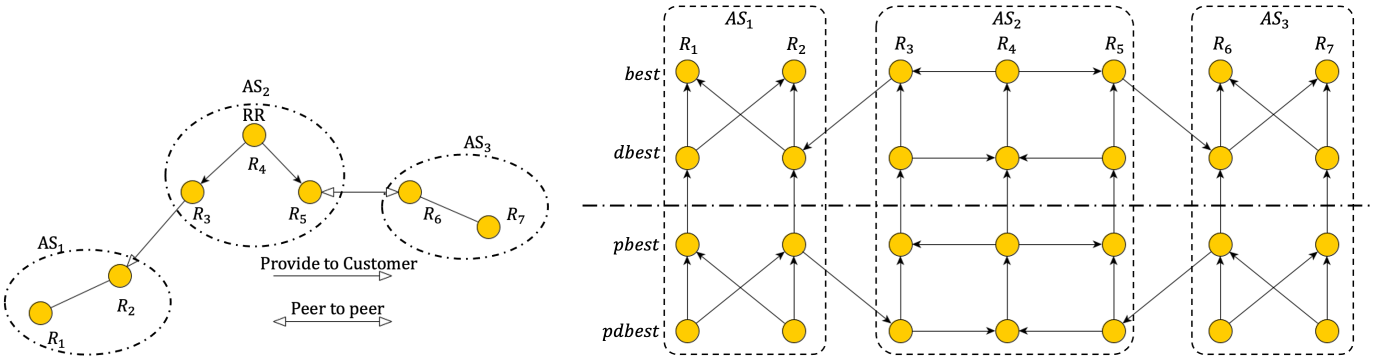
We implement BiNode [31] and evaluate BiNode extensively. We introduce the experiment settings in Section 6.2. In Section 6.3, we compare the performance of BiNode and the topology-based approach based on a series of network configurations. We compare BiNode with the state-of-the-art control plane verification tools in Section 6.4.

6.1 Implementation

We use Batfish [14] to parse router configurations and implement BiNode based on Minesweeper [5]. The verification system will encode the network properties to be verified into an SMT problem and then fed into an SMT solver to get the verification results. We use Z3 [9] as the SMT solver in this work.

6.2 Experiment Setting

To evaluate BiNode for network verification, we use the synthesized configurations of networks of various sizes. We



(a) A router-level topology with AS relations.

(b) Dependency graph for (a).

Fig. 10: Apply BiNode to a topology with both iBGP sessions and eBGP sessions.

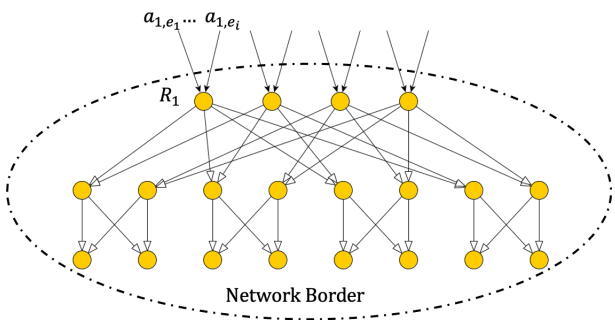


Fig. 11: Clos data center network topology.

consider two types of synthesized networks: data center networks and ISP networks. The data center networks use the fat-tree architecture and BGP as intra-domain routing [3], [10], [33] protocol. ISP networks use a fully meshed iBGP architecture or route reflection configuration. Table 2 summarizes the networks used in the experiments.

We test the verification system through a set of properties on networks. We consider the following properties in the experiment.

- Pair-wise reachability property (RE): Inside of a network, all routers should be able to reach all interfaces of all routers. For the destinations outside of a network, all routers should be able to reach the destination if the border routers learn the BGP routes to the destination.
- Bounded length property (BL): The traffic never traverses a path longer than k number of hops in a network. We use $k = 3$ in the experiment.
- Equal length property (EL): The selected paths have equal length when equal-cost multi-path routing (ECMP) is enabled.
- Forwarding property (FW): The router configurations lead to a stable data plane for packet forwarding.

We test the verification system on a server with 8-core Intel Xeon 2.4 GHz CPU and 32GB memory.

6.3 Compare with Topology-Based Approach

We use Minesweeper as the vanilla implementation of the topology-based approach. We verify properties on networks with various sizes and topologies in the experiments.

6.3.1 *ISP Networks with Fully Meshed iBGP*

Fully meshed iBGP is the trivial topology for the iBGP configuration within an AS. We use BiNode and Minesweeper to verify properties on a series of ISP networks with fully meshed iBGP and illustrate the verification time in Figure 12. As Figure 12 shows, BiNode can significantly reduce the verification time. For a moderate-sized network with more than 100 routers, the reduction of verification time is more than 70%.

6.3.2 *ISP Networks with Route Reflection*

BiNode can also benefit from the cycle-free SMT constraints when verifying properties related to multiple ASes owned by an organization are verified. To show that, we compare BiNode with Minesweeper for verifying multiple ASes. We first synthesize networks containing three ASes with each AS consisting of multiple routers and then synthesize networks containing multiple ASes with each AS consisting of 25 routers. ASes are pairwise connected. We consider that they use route reflection within the AS. We evaluate the performance of BiNode and Minesweeper on these networks.

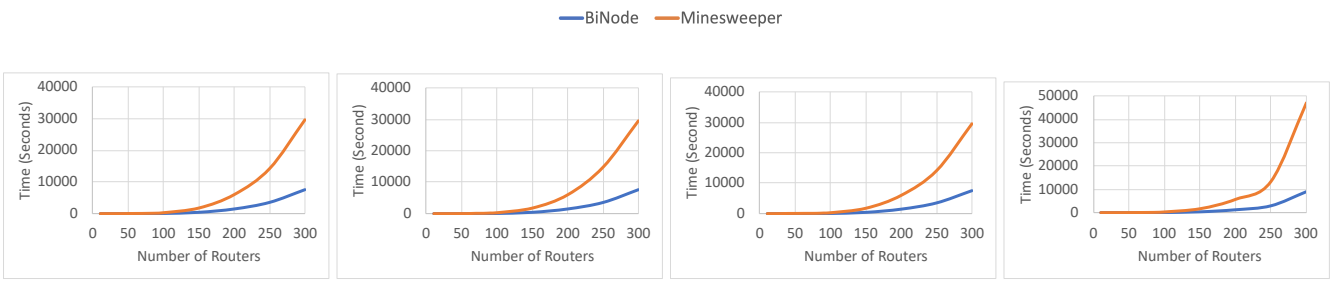
We show the verification time of BiNode and Minesweeper as the number of routers increases in Figure 13 and show the verification time as the number of ASes increases in Figure 14. As Figure 13 shows, BiNode can reduce the verification time by more than 90% when the size of the network grows to hundreds of routers and reduce the verification time by more than 80% when the number of ASes grows to 30.

6.3.3 *Data Center Networks*

Currently, BGP is not only used in inter-domain routing among ASes. It is also used among routers in large-scale data centers for better scalability. [10], [33] Although the routing policies of BGP routers in a data center do not follow any commercial agreements, such as provider-customer and peer-peer, BiNode can still benefit from the hierarchical topology of BGP routers in the data center networks. We evaluate the performance of BiNode and Minesweeper for data center networks and illustrate the verification time in Figure 15. As Figure 15 shows, BiNode can significantly reduce the verification time.

6.3.4 *Verifying Buggy Network Configurations*

We show that BiNode can successfully find bugs in configurations efficiently. We consider the following network



(a) Reachability (RE) (b) Bounded-length (BL) (c) Equal length (EL) (d) Forwarding (FW)

Fig. 12: Verification time for ISP with fully meshed iBGP.

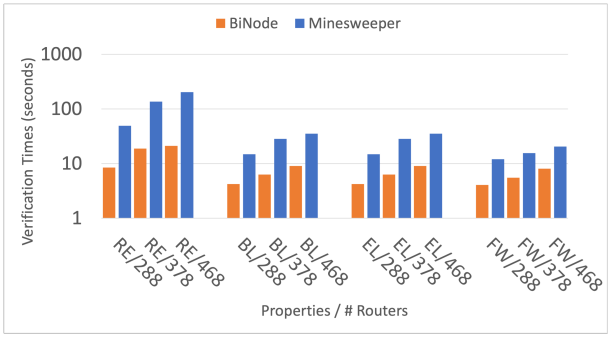


Fig. 13: Verification time for ISP with route reflection.

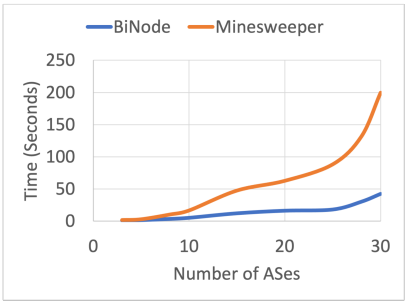
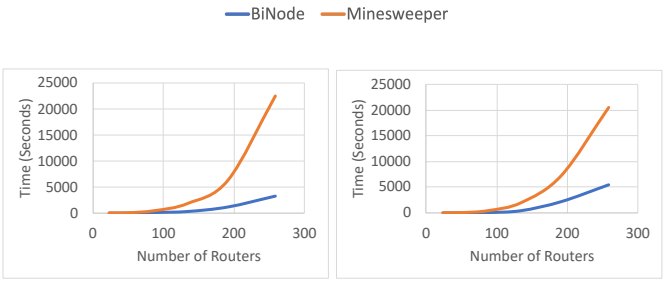


Fig. 14: Verification time for multiple ASes.



(a) Reachability (b) Bounded-length

Fig. 15: Verification time for data center networks.

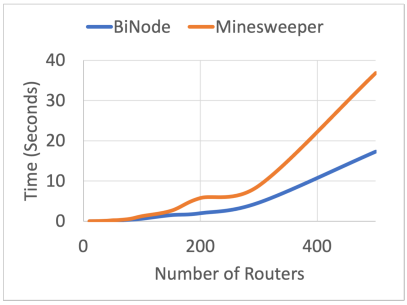


Fig. 16: Verifying reachability property for buggy iBGP configurations.

misconfiguration scenarios. We randomly take down iBGP sessions in fully meshed iBGP architecture and iBGP with route reflection with 10% probability. We synthesize the corresponding configuration and verify the pair-wise reachability property (obviously violated) with both BiNode and Minesweeper under these configurations. We show the verification time of BiNode and that of Minesweeper in Figure 16. The experimental results show that BiNode can reduce the verification time by more than 60% when the size of the network grows to hundreds of routers.

6.3.5 Memory Usage

BiNode achieves the speed up without using extra memories. We also compare the memory usage of BiNode and Minesweeper. We show the memory consumption of both BiNode and Minesweeper in Figure 17 and Figure 18. The results show that BiNode uses 20% to 50% less memory than Minesweeper.

6.4 Compare with State-of-the-Art Control Plane Verification Scheme

We compare the verification time of BiNode with the state-of-art network verification tool. We compare the pair-wise verification time of reachability property on ISP and data center networks with Tiramisu [1]. We use both fully meshed iBGP topology and iBGP with route reflection for generating configurations. Figure 19 shows that BiNode is on average 40% faster than Tiramisu.

We also compare the memory usage between BiNode and Tiramisu. We show the experimental results in Figure 20. The experimental result shows that BiNode can achieve around 40% speedup while saving on average 20% memory compared with Tiramisu. When verifying pair-wise reachability properties on ISP networks with route reflection, BiNode can reduce memory usage by one order of magnitude.

7 Related Work

7.1 Data Plane Verification

Offline data plane verifications [23], [26], [34] perform verification on network properties based on the forwarding behavior collected from the forwarding information base (FIB). Data plane verifications focus on one snapshot of the network state. FIB entries directly reflect the forwarding

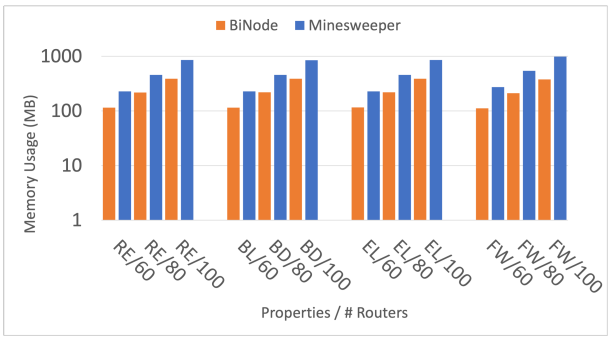
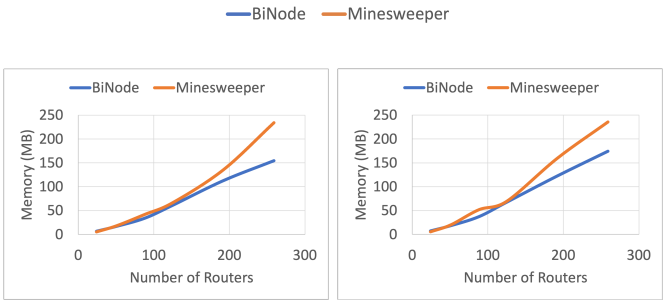


Fig. 17: Memory usage for ISP networks.



(a) Reachability (b) Bounded-length

Fig. 18: Memory usage in verifying data center networks.

behavior in the network, and therefore data plane verifications can verify properties efficiently. However, data plane states change frequently, and therefore offline data plane verifications have limitations. Network properties might not be guaranteed when routing events such as link failures.

Real-time data plane verifications [21], [24], [35], [42], [43] can verify properties under new data plane states within milliseconds, which can be used to detect buggy updates on FIB entries on the fly. Delta-net [19] and HSA [22] have been proposed to serve as a filter layer to filter out changes that can cause violation of properties. Still, this can not guarantee that network properties preserve under certain routing events which are unavoidable in the real world. For example, one cannot prevent a link or device from failure.

7.2 Control Plane Verification

Control plane verifications [5], [12], [14], [41] take the configuration files and network environments as input so that they can verify network properties under all possible data plane states. Control plane verifications cover more network states than data plane verifications and can guarantee the network properties hold for the configuration even under certain routing events. Batfish [14] can derive the data plane from the configuration files and a given environment (*e.g.*, a link-failure scenario). Then, it can check the configuration under the specific environment through analyzing the resulting data plane. ERA [12] compactly represents the control plane messages through the binary decision diagrams (BDDs) for reachability check. However, it is intractable for Batfish and ERA to verify configurations under all possible environments. Bagpipe [41] uses an SMT solver to verify network properties of the inter-domain routing for all possible environments. Further, Minesweeper [5] verifies inter-domain and intra-domain properties through the SMT model. However, it is challenging to scale control plane verifications to large networks since control plane verification needs to verify properties under multiple data plane states at one time.

Over the years, various models [1], [16], [29] have been proposed to improve the scalability of control plane verifications. One successful approach is to use the Simple Path Vector Protocol [18]. Tiramisu [1] proposed TPVP (Tiramisu Path Vector Protocol) and applied property-specific optimization to speed up the verification. Plankton [29] proposed RPVP (Reduced Path Vector Protocol). It used the explicit state model checker to calculate the routing table so that they can incorporate the routing-specific knowledge and emulate the routing protocols and accelerate the verification process. Even though they incorporate the routing-specific information and encode the control plane with a variant of SPVP, there are still many paths to enumerate to get the best route finally.

In this paper, we choose to optimize SMT-based approaches, which furthermore take advantage of routing-specific knowledge and break the best routes into groups to accelerate the solving process. BiNode speeds up the control plane verification and can scale to large networks.

7.3 Network Compression for Verification

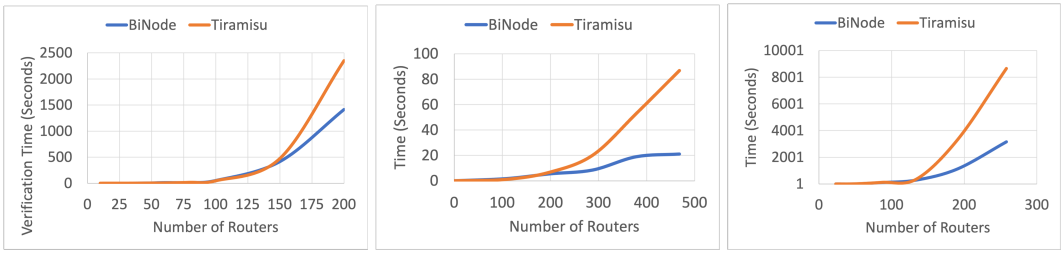
Wang *et al.* [38], [40] propose a network compression method for policy-based routing to accelerate the analysis of convergence behavior. The method reduces the model size by exploiting the duplicate router configuration and the symmetry in the topologies. In contrast, BiNode exploits the intrinsic hierarchy in the routing policy and explicitly reflects the hierarchy in the system to accelerate the formal method. To accelerate the verification, Beckett *et al.* [6] propose a compressed model for a broad range of routing protocols to preserve general path properties, such as reachability, loop freedom and absence of black holes. The network compression method and our method are orthogonal. Our method can be applied after the network compression.

8 Conclusion

In this paper, we propose BiNode to verify network properties through characterizing the verification problem as an SMT problem. Through analyzing the intrinsic characteristics of the routing policies in the network, we can reflect the characteristics in the constraints to accelerate the SMT solving process. We implement the network verification toolkit which exploits the proposed approach. The experimental results show that BiNode can reduce the time it takes to perform verification by an order of magnitude.

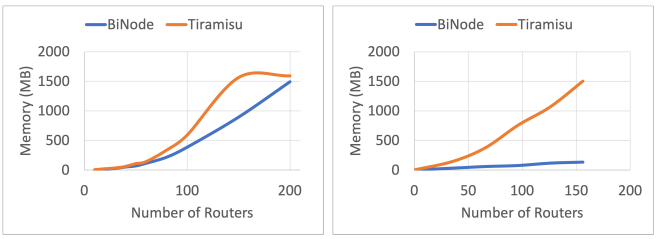
Acknowledgment

The work was supported in part by NSF grants CNS-1900866 and CCF-1918187.



(a) Fully meshed ISP networks. (b) ISP networks with iBGP route reflection. (c) Data center networks.

Fig. 19: BiNode v.s. Tiramisu in verification time.



(a) Fully meshed ISP networks. (b) ISP networks with iBGP route reflection.

Fig. 20: BiNode v.s. Tiramisu in memory usage.

References

[1] A. Abhashkumar, A. Gember-Jacobson, and A. Akella. Tiramisu: Fast multilayer network verification. In 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20), pages 201–219, Santa Clara, CA, Feb. 2020. USENIX Association.

[2] A. Abhashkumar, K. Subramanian, A. Andreyev, H. Kim, N. K. Salem, J. Yang, P. Lapukhov, A. Akella, and H. Zeng. Running BGP in data centers at scale. In 18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21), pages 65–81. USENIX Association, Apr. 2021.

[3] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. In Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, SIGCOMM '08, page 63–74, New York, NY, USA, 2008. Association for Computing Machinery.

[4] R. Anwar, H. Niaz, D. Choffnes, I. Cunha, P. Gill, and E. Katz-Bassett. Investigating interdomain routing policies in the wild. In Proceedings of the 2015 Internet Measurement Conference, IMC '15, pages 71–77, New York, NY, USA, 2015. ACM.

[5] R. Beckett, A. Gupta, R. Mahajan, and D. Walker. A general approach to network configuration verification. In Proceedings of the Conference of the ACM Special Interest Group on Data Communication, SIGCOMM '17, page 155–168, New York, NY, USA, 2017. Association for Computing Machinery.

[6] R. Beckett, A. Gupta, R. Mahajan, and D. Walker. Control plane compression. In Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, SIGCOMM '18, pages 476–489, New York, NY, USA, 2018. Association for Computing Machinery.

[7] CAIDA. Inferred as to organization mapping dataset. https://www.caida.org/catalog/datasets/as-organizations/.

[8] E. Chen, T. J. Bates, and R. Chandra. BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP). RFC 4456, Apr. 2006.

[9] L. De Moura and N. Bjørner. Z3: An efficient smt solver. In Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'08/ETAPS'08, page 337–340, Berlin, Heidelberg, 2008. Springer-Verlag.

[10] D. G. Dutt. BGP in the Data Center. O'Reilly Media, Inc., 2017.

[11] Facebook. Update about the october 4th outage. https://engineering.fb.com/2021/10/04/networking-traffic/outage/.

[12] S. K. Fayaz, T. Sharma, A. Fogel, R. Mahajan, T. Millstein, V. Sekar, and G. Varghese. Efficient network reachability analysis using a succinct control plane representation. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pages 217–232, Savannah, GA, Nov. 2016. USENIX Association.

[13] N. Feamster and J. Rexford. Network-wide prediction of bgp routes. IEEE/ACM Transactions on Networking, 15(2):253–266, 2007.

[14] A. Fogel, S. Fung, L. Pedrosa, M. Walraed-Sullivan, R. Govindan, R. Mahajan, and T. Millstein. A general approach to network configuration analysis. In Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation, NSDI'15, page 469–483, USA, 2015. USENIX Association.

[15] L. Gao and J. Rexford. Stable internet routing without global coordination. IEEE/ACM Transactions on Networking, 9(6):681–692, Dec 2001.

[16] A. Gember-Jacobson, R. Viswanathan, A. Akella, and R. Mahajan. Fast control plane analysis using an abstract representation. In Proceedings of the 2016 ACM SIGCOMM Conference, SIGCOMM '16, page 300–313, New York, NY, USA, 2016. Association for Computing Machinery.

[17] P. Gill, M. Schapira, and S. Goldberg. A survey of interdomain routing policies. SIGCOMM Comput. Commun. Rev., 44(1):28–34, Dec. 2013.

[18] T. G. Griffin, F. B. Shepherd, and G. Wilfong. The stable paths problem and interdomain routing. IEEE/ACM Trans. Netw., 10(2):232–243, apr 2002.

[19] A. Horn, A. Kheradmand, and M. Prasad. Delta-net: Real-time network verification using atoms. In 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17), pages 735–749, Boston, MA, Mar. 2017. USENIX Association.

[20] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat. B4: Experience with a globally-deployed software defined wan. In Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM, SIGCOMM '13, page 3–14, New York, NY, USA, 2013. Association for Computing Machinery.

[21] K. Jayaraman, N. Bjørner, J. Padhye, A. Agrawal, A. Bhargava, P.-A. C. Bissonnette, S. Foster, A. Helwer, M. Kasten, I. Lee, A. Namdhari, H. Niaz, A. Parkhi, H. Pinnamraju, A. Power, N. M. Raje, and P. Sharma. Validating datacenters at scale. In Proceedings of the ACM Special Interest Group on Data Communication, SIGCOMM '19, page 200–213, New York, NY, USA, 2019. Association for Computing Machinery.

[22] P. Kazemian, M. Chang, H. Zeng, G. Varghese, N. McKeown, and S. Whyte. Real time network policy checking using header space analysis. In 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13), pages 99–111, Lombard, IL, Apr. 2013. USENIX Association.

[23] P. Kazemian, G. Varghese, and N. McKeown. Header space analysis: Static checking for networks. In Proceedings of NSDI 2012: 9th USENIX Symposium on Networked Systems Design and Implementation, pages 113–126, 2012.

[24] A. Khurshid, X. Zou, W. Zhou, M. Caesar, and P. B. Godfrey. Veriflow: Verifying network-wide invariants in real time. In Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13), pages 15–27, Lombard, IL, 2013. USENIX.

[25] H. H. Liu, X. Wu, W. Zhou, W. Chen, T. Wang, H. Xu, L. Zhou, Q. Ma, and M. Zhang. Automatic life cycle management of network configurations. In Proceedings of the Afternoon Workshop on Self-Driving Networks, SelfDN 2018, page 29–35, New York, NY, USA, 2018. Association for Computing Machinery.

[26] H. Mai, A. Khurshid, R. Agarwal, M. Caesar, P. B. Godfrey, and S. T. King. Debugging the data plane with anteater. SIGCOMM Comput. Commun. Rev., 41(4):290–301, Aug. 2011.

[27] R. Mazloum, M. O. Buob, J. Augè, B. Baynat, D. Rossi, and T. Friedman. Violation of interdomain routing assumptions. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8362 LNCS:173–182, 2014.

[28] Large route leaks. http://nrl.cs.arizona.edu/projects/lsrl-events-from-2003-to-2009.

[29] S. Prabhu, K. Y. Chou, A. Kheradmand, B. Godfrey, and M. Caesar. Plankton: Scalable network configuration verification through model checking. In 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20), pages 953–967, Santa Clara, CA, Feb. 2020. USENIX Association.

[30] S. Y. Qiu, P. D. McDaniel, and F. Monrose. Toward valley-free inter-domain routing. In 2007 IEEE International Conference on Communications, pages 2009–2016, June 2007.

[31] X. Shao, Z. Chen, D. Holcomb, and L. Gao. BiNode. https://github.com/xiaozheshao/BiNode.

[32] X. Shao and L. Gao. Verifying policy-based routing at internet scale. In IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, page 2293–2302. IEEE Press, 2020.

[33] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, A. Kanagala, J. Provost, J. Simmons, E. Tanda, J. Wanderer, U. Hölzle, S. Stuart, and A. Vahdat. Jupiter rising: A decade of clos topologies and centralized control in google's datacenter network. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM '15, page 183–197, New York, NY, USA, 2015. Association for Computing Machinery.

[34] R. Stoenescu, M. Popovici, L. Negreanu, and C. Raiciu. Symnet: Scalable symbolic execution for modern networks. In Proceedings of the 2016 ACM SIGCOMM Conference, SIGCOMM '16, page 314–327, New York, NY, USA, 2016. Association for Computing Machinery.

[35] B. Tian, X. Zhang, E. Zhai, H. H. Liu, Q. Ye, C. Wang, X. Wu, Z. Ji, Y. Sang, M. Zhang, D. Yu, C. Tian, H. Zheng, and B. Y. Zhao. Safely and automatically updating in-network acl configurations with intent language. In Proceedings of the ACM Special Interest Group on Data Communication, SIGCOMM '19, page 214–226, New York, NY, USA, 2019. Association for Computing Machinery.

[36] Bgp super-blunder: How verizon today sparked a 'cascading catastrophic failure' that knackered cloudflare, amazon, etc. https://www.theregister.com/2019/06/24/verizon_bgp_misconfiguration_cloudflare/.

[37] Verizon 2020 data breach investigations report. https://enterprise.verizon.com/resources/reports/dbir/2020/summary-of-findings/.

[38] A. Wang, A. J. T. Gurney, X. Han, J. Cao, B. T. Loo, C. Talcott, and A. Scedrov. A reduction-based approach towards scaling up formal analysis of internet configurations. In IEEE INFOCOM 2014 - IEEE Conference on Computer Communications, pages 637–645, April 2014.

[39] A. Wang, L. Jia, W. Zhou, Y. Ren, B. T. Loo, J. Rexford, V. Nigam, A. Scedrov, and C. Talcott. Fsr: Formal analysis and implementation toolkit for safe interdomain routing. IEEE/ACM Transactions on Networking, 20(6):1814–1827, 2012.

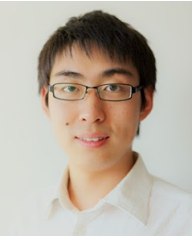
[40] A. Wang, C. Talcott, A. J. T. Gurney, B. T. Loo, and A. Scedrov. Reduction-based formal analysis of bgp instances. In Proceedings of the 18th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS'12, pages 283–298, Berlin, Heidelberg, 2012. Springer-Verlag.

[41] K. Weitz, D. Woos, E. Torlak, M. D. Ernst, A. Krishnamurthy, and Z. Tatlock. Scalable verification of border gateway protocol configurations with an smt solver. SIGPLAN Not., 51(10):765–780, oct 2016.

[42] H. Yang and S. S. Lam. Real-time verification of network properties using atomic predicates. IEEE/ACM Trans. Netw., 24(2):887–900, Apr. 2016.

[43] H. Yang and S. S. Lam. Scalable verification of networks with packet transformers using atomic predicates. IEEE/ACM Transactions on Networking, 25(5):2900–2915, 2017.

[44] P. Zhang, X. Liu, H. Yang, N. Kang, Z. Gu, and H. Li. Apkeep: Realtime verification for real networks. In 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20), pages 241–255, Santa Clara, CA, Feb. 2020. USENIX Association.



Xiaozhe Shao received his Ph.D. degree in Electrical and Computer Engineering from the University of Massachusetts at Amherst, and received BE and MS degrees in Computer Science from University of Science and Technology of China. He is currently working at Meta Platforms, Inc. His current research interests include interdomain routing, network verification and graph data anonymization.

Zibin Chen received his M.S. degree from the University of Massachusetts, Amherst, USA, in 2021. Prior to joining UMass, he received his B.E. degree from Shandong Normal University, China. He is currently pursuing a Ph.D. degree with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, USA.

Daniel Holcomb received the B.S. and M.S. degrees in Electrical and Computer Engineering from the University of Massachusetts Amherst, Amherst, MA, USA, in 2005 and 2007, respectively, and the Ph.D. degree in Electrical Engineering and Computer Sciences from the University of California at Berkeley, Berkeley, CA, USA, in 2013. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Massachusetts Amherst. His research interests are in hardware security and embedded systems.

Lixin Gao is a University Distinguished Professor of Electrical and Computer Engineering at the University of Massachusetts at Amherst. She received a Ph.D. degree in Computer Science from the University of Massachusetts at Amherst. Her research interests include online social networks, Internet routing, network virtualization and cloud computing. Between May 1999 and January 2000, she was a visiting researcher at AT&T Research Labs and DIMACS. She was an Alfred P. Sloan Fellow between 2003-2005 and received an NSF CAREER Award in 1999. She won the best paper award from IEEE INFOCOM 2010, and the test-of-time award in ACM SIGMETRICS 2010. Her paper in ACM Cloud Computing 2011 was honored with "Paper of Distinction". She received the Chancellor's Award for Outstanding Accomplishment in Research and Creative Activity in 2010, College of Engineering Outstanding Senior Faculty Award in 2013, and Outstanding Achievement in Research Award by College of Information and Computer Sciences at the University of Massachusetts in 2015. She is a fellow of IEEE and ACM.

APPENDIX A
BINODE FOR FULLY MESHED IBGP

We show BiNode is equivalent to the topology-based approach in this section. We summarize the constraints of both BiNode and the topology-based approach for fully meshed iBGP and then prove Theorem 3.1. Consider a fully meshed iBGP system with n BGP routers. According to Section 2, the constraints derived from the topology-based approach are as follows.

Deriving topology-based constraints for Fully Meshed iBGP

```
for each BGP router R_p, construct the
    constraints for the following equation:
  best_p = f_sel({C_{p,e} | R_e ∈ N_e(R_p)}
           ∪{C_{p,q} | 1 ≤ q ≤ n, q ≠ p})
  for q ∈ [1, p−1] ∪ [p+1, n], construct the
      constraints for the following equation:
    C_{p,q} = f^{p,q}_imp(f^{q,p}_exp(best_q))
  for R_e ∈ N_e(R_p), construct the constraints
      for the following equation:
    C_{p,e} = f^{p,e}_imp(a_{p,e})
```

where $N_e(R_p)$ denotes the external BGP neighbor set of R_p, and $f^{p,q}_{\text{imp}}$ represent the import policy deployed on router p for router q, $f^{q,p}_{\text{exp}}$ represent the export policy deployed on router q for router p. The corresponding constraints from BiNode are as follows.

Deriving Constraints from BiNode for Fully Meshed iBGP

```
for each BGP router R_p, construct the
    constraints for the following equation:
  best_p = f_sel({dbest_p} ∪ {C_{p,q} | 1 ≤ q ≤ n, q ≠ p})
  for R_q ∈ [1, p−1] ∪ [p+1, n], construct the
      constraints for the following equation:
    C_{p,q} = g_{q,p}(dbest_q)
  dbest_p = f_sel({C_{p,e} | R_e ∈ N_e(R_p)})
  for R_e ∈ N_e(R_p), construct the constraints
      for the following equation:
    C_{p,e} = f^{p,e}_imp(a_{p,e})
```

We now prove Theorem 3.1.

Theorem 3.1. *The best routes derived from BiNode are the same as that from the topology-based approach for any fully meshed iBGP system.*

We prove Theorem 3.1 by proving the following lemmas.

Lemma 1. *Given satisfiable assignments for SMT variables in the topology-based approach for a fully meshed iBGP routing system, i.e., $best^T_p$, there are satisfiable assignments for SMT variables in BiNode, i.e., $best^B_p$, where for any R_p, $best^T_p = best^B_p$.*

Proof. Given satisfiable assignments for variables in the topology-based approach for a fully meshed iBGP routing system, we construct the satisfiable assignments for variables in BiNode and show the solutions satisfy the constraints of the selection functions and import and export policies.

$$\forall R_p, best^B_p = best^T_p \tag{13}$$

We derive the solutions to variables for $dbest$ in BiNode as follows.

$\forall R_p$,

$$dbest^B_p = \begin{cases} best^T_p & \text{if } best^T_p \text{ is learned via eBGP.} \\ \text{Equation (7)} & \text{otherwise} \end{cases} \tag{14}$$

#	Attribute	Preference	Consistent?
1	$r.pref$	Highest local preference.	Y
2	$r.length$	Shortest AS path length.	Y
3	$r.med$	Lowest MED value.	N
4	$r.ibgp$	Prefer eBGP over iBGP.	N
5	$r.distance$	Lowest next hop IGP cost.	N
6	$r.rid$	Lowest router ID.	-

TABLE 3: BGP decision precess.

We then show the above derived assignments for $best$ and $dbest$ are satisfiable assignments to constraints in BiNode. We consider two cases for $best^T_p$ mentioned in Equation (14).

Case 1: $best^T_p$ is learned via eBGP. In this case, $best^B_p = dbest^B_p$ where

$$best^B_p = dbest^B_p = f_{\text{sel}}(\{C_{p,e} | R_e \in N_e(R_p)\}) \tag{15}$$

When $best^T_p$ is learned via eBGP session, we have the external route announcements applying import policies as candidate routes. That is,

$$best^B_p = best^T_p = f_{\text{sel}}(\{f^{p,e}_{\text{imp}}(a_{p,e}) | R_e \in N_e(R_p)\}) \tag{16}$$

Apparently, $best^B_p$ satisfies the constraints of selection function and import policies. Similarly, $dbest^B_p$ satisfies the constraints of selection function and import policies.

Case 2: $best^T_p$ is learned via iBGP. Similar to first case, we have $dbest^B_p$ satisfies the constraints of selection function and import policy, because in this case, we have

$$dbest^B_p = f_{\text{sel}}(\{f^{p,e}_{\text{imp}}(a_{p,e}) | R_e \in N_e(R_p)\}) \tag{17}$$

We now show $best^B_p$ and $C_{p,q}$ satisfies the constraints of selection function and import policy and export policy. As mentioned before, the ranking of a route is related to a specific router. That is because the ranking of a route is inconsistent over routers. [13] Table 3 summarizes the attributes used to calculate a ranking of a BGP route. As we can see from the table, local preference and AS path length are consistent over the network. We use $c.rank_e$ to represent the first two rankings of a BGP route (i.e., ranking only consider the attributes set by eBGP), since they are consistent over the network, no subscripts are needed. Inspired by [13], we use set $\mathbf{S}$ to denote the set of routers whose external best route has a significant lower preference in the entire network. More specifically,

$$\mathbf{S} = \{R_q | dbest_q.rank_e < \max_R(dbest_k.rank_e)\} \tag{18}$$

We consider candidate routes from BGP neighbors of R_p, say R_q.

- If $R_q \notin \mathbf{S}$, we have $best^T_q$ satisfies case 1, $C_{p,q}$ satisfies the constraints of import policy and export policy.
- If $R_q \in \mathbf{S}$, there exists $R_k \notin \mathbf{S}$, $C_{p,k}.rank_e > C_{p,q}.rank_e$. As a result, $C_{p,k}.rank > C_{p,q}.rank$. $C_{p,q}$ satisfies the constraints of import policy and export policy. Since the ranking of $C_{p,q}$ at R_p is smaller than the ranking of $C_{p,k}$ and can never be picked by $best^B_p$.

As a result, $best^B_p$ satisfies the constraints of selection function and import policies.

In summary, the derived solutions to $best_p$ and $dbest_p$ are satisfiable assignments for variables in BiNode. Lemma 1 is proved. □

Lemma 2. *Given satisfiable assignments for variables in BiNode for fully meshed iBGP routing system, i.e., $best_p^B$, there are satisfiable assignments for variables in the corresponding topology-based approach, where for any R_p, $best_p^T = best_p^B$.*

Proof. Given satisfiable assignments for variables in BiNode for a fully meshed iBGP routing system, we construct the satisfiable assignments for variables in the topology-based approach and show the solution satisfies the selection function and import and export policies.

$$\forall R_p, best_p^T = best_p^B \tag{19}$$

where $best_p^B$ can be expressed with Equation (9) and Equation (10). Similarly, $best_p^B$ satisfies the import function and selection function. We can show this is true by considering two cases similar to Lemma 1. Obviously, if $best_p^B$ is learned via eBGP session, i.e., $dbest_p^B$, $C_{p,e}$ is the route announcements applying the same import policy and $best_p$ satisfies the constraints of import policy and export policy. If $best_p^B$ is learned via iBGP, we consider set **S** and show $C_{p,q}$ satisfies the constraints of import policy and export policy in the following two cases.

- If $R_q \notin \mathbf{S}$, we have $best_q^B$ is learned via eBGP sessions, in this case, $C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(best_q))$. C_pq satisfies the constraints of import policy and export policy.
- If $R_q \in \mathbf{S}$, there exists $R_k \notin \mathbf{S}$, $C_{p,k}.rank > C_{p,q}.rank$. $C_{p,q}$ cannot be the best route, since in fully meshed iBGP routing system, $C_{p,k}$ is a candidate route of $best_k$. As a result, $C_{p,q}$ satisfies the constraints of import policy and export policy.

As a result, $best_p^T$ satisfies the constraints of selection function and import policies and export policies.

In summary, the derived solutions are satisfiable assignments for variables in the topology-based approach. Lemma 2 is proved.

□

As a result, BiNode for fully meshed iBGP is equivalent to the topology-based approach.

APPENDIX B
BINODE FOR IBGP WITH ROUTE REFLECTION

In this section, we first derive the SMT constraints for BiNode for iBGP with route reflection and then prove BiNode for iBGP with route reflection is equivalent to the topology-based approach.

B.1 Deriving Constraints for iBGP with Route Reflection

The corresponding constraints for any BGP router R_p consist of two parts, the constraints for $best_p$ and the constraint for $dbest_p$. More specifically, the constraints for $best_p$ and $dbest_p$ are as follows.

Constraint for $dbest_p$:

Constraint for $dbest_p$ are similar to that of fully meshed iBGP system. That is,

$$dbest_p = f_{\text{sel}}(\{C_{p,e} | R_e \in N_e(R_p)\} \cup \{C_{p,q} | R_q \in N_c(R_p)\}) \tag{20}$$

where $C_{p,e}$ represents candidate routes learned from external BGP peer R_e. As a result, $C_{p,e}$ can be expressed with

$$C_{p,e} = f_{\text{imp}}^{p,e}(a_{p,e}) \tag{21}$$

$C_{p,q}$ represent the candidate routes learned from iBGP neighbor R_q. As a result, $C_{p,q}$ can be expressed with

$$C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(best_q)) \tag{22}$$

Constraint for $best_p$:

$$best_p = f_{\text{sel}}(\{dbest_p\} \cup \{C_{p,q} | R_q \in N_n(R_p) \cup N_r(R_p)\}) \tag{23}$$

where $N_n(R_p)$ represents the set of iBGP neighbors at the same tier establishing non-client sessions with R_p and $N_r(R_p)$ represent the set of RR neighbors of R_p. As a result, $C_{p,q}$ for $R_q \in N_n(R_p)$ can be expressed with

$$C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(dbest_q)) \tag{24}$$

$C_{p,q}$ for $R_q \in N_r(R_p)$ can be expressed with

$$C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(best_q)) \tag{25}$$

In summary, for iBGP with route reflection, we build the constraints for BiNode for iBGP routing system with route reflection as follows.

Deriving constraints for BiNode for iBGP with Route Reflection

```
for each BGP router R_p, construct the
    constraints for the following equation:
  best_p = f_sel({dbest_p} ∪ {C_{c,q} | R_q ∈ N_r(R_p) ∪ N_n(R_p)})
  for R_q ∈ N_r(R_p), construct the constraints
      for the following equation:
    C_{p,q} = f_imp^{p,q}(f_exp^{q,p}(best_q))
  for R_q ∈ N_n(R_p), construct the constraints
      for the following equation:
    C_{p,q} = f_imp^{p,q}(f_exp^{q,p}(dbest_q))
  dbest_p = f_sel({C_{p,e} | R_e ∈ N_e(R_p)} ∪ {C_{p,q} | R_q ∈ N_c(R_p)})
  for R_e ∈ N_e(R_p), construct the constraints
      for the following equation:
    C_{p,e} = f_imp^{p,e}(a_{p,e})
  for R_q ∈ N_c(R_p), construct the constraints
      for the following equation:
    C_{p,q} = f_imp^{p,q} f_exp^{q,p}(dbest_q))
```

B.2 BiNode is Equivalent to the Topology-Based Approach

We show for any iBGP system with route reflection, BiNode is equivalent to the corresponding topology-based approach. We show they are equivalent by proving Theorem 3.2.

Theorem 3.2. *The best routes derived from BiNode are the same as that from the topology-based approach for any iBGP system with route reflection.*

We can prove Theorem 3.2 by proving Lemma 3 and Lemma 4. We show BiNode is correct under realistic routing systems discussed in Section 3.3.

Lemma 3. *Given satisfiable assignments for variables in the topology-based approach for iBGP routing system with route reflection, i.e., $best_p^T$, there are satisfiable assignments for SMT variables in BiNode, where for any R_p, $best_p^T = best_p^B$.*

Proof. Given satisfiable assignments for variables in the topology-based approach, we construct satisfiable assignments for variables in BiNode for iBGP routing system with route reflection as follows. We first construct the satisfiable assignments for variables representing best routes as follows.

$$\forall R_p, best_p^B = best_p^T \tag{26}$$

We then derive the solutions to variables for $dbest$ in BiNode as follows.

$\forall R_p$,

$$dbest_p^B = \begin{cases} best_p^T & \text{if } best_p^T \text{ is learned from down nodes.} \\ \text{Equation (20)} & \text{otherwise} \end{cases} \tag{27}$$

Similarly, we show the derived assignments for BiNode are satisfiable assignments by considering three cases of $best_p^T$.

Case 1: $best_p^T$ is learned from down node R_q of R_p. We have $dbest_p^B = best_p^B$ and $C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(dbest_q))$. We extend set $\mathbf{S}$ described in Lemma 1 to $\mathbf{S}_d$ to denote the set of routers whose client best route has significant lower preference in the entire network. More specifically,

Case 2: $best_p^T$ is learned from same tier neighbor of R_p, for example, R_q. In this case, we have $best_p^B = f_{\text{sel}}(\{C_{p,q}|R_q \in N_d(R_p)\})$ where $C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(dbest_q))$.

$$\mathbf{S}_d = \{R_q| \max_{R_e,R_c:N_l(R_p)}(a_{c,e}.rank_e) < \max_{R_e,R_p}(a_{p,e}.rank_e)\} \tag{28}$$

where $N_l(R_p)$ represents the set of lower tier routers of R_p. We show $C_{p,q}$ satisfies the constraints of import/export policy in this case by considering two cases of R_q.

- $R_q \notin \mathbf{S}_d$. Obviously, $dbest_p$ is the most preferred candidate route of R_q, since $dbest_q$ has lowest next hop IGP cost. As a result, in this case, $C_{p,q}$ satisfies the constraints of the import/export policy.
- $R_q \in \mathbf{S}_d$. In this case, $best_q \neq dbest_q$. Suppose there exist $R_k \notin \mathbf{S}$, such that $best_q = C_{q,k}$. As a cluster in an iBGP system with route reflection are typically physically located in the same point-of-presence (PoP) and therefore client route typically has two characteristics: same MED value among customer routes and lower next hop IGP cost than non-client route. The only way $C_{q,k}$ has have a higher ranking than $dbest_p$ is that $C_{q,k}.rank_e > dbest_q.rank_e$. As a result, there must be $C_{p,t}$ which is a candidate route for R_q such that $C_{p,t}.rank_e = C_{q,k}.rank_e > C_{p,q}.rank_e$. As a result, $C_{p,q}$ satisfies the constraints of import/export policy.

Case 3: $best_p^T$ is learned from higher tier neighbor of R_p, say, R_q. In this case, we have $best_p^B = f_{\text{sel}}(\{C_{p,q}|R_q \in N_d(R_p)\})$ where $C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(best_q))$. Similar to the first case, we have $C_{p,q}$ satisfies constraints of import policy and export policy.

In summary, the derived solutions to $best_p$ and $dbest_p$ are satisfiable assignments for variables in BiNode. Lemma 3 is proved. □

Now we prove Lemma 4.

Lemma 4. *Given satisfiable assignments for variables in BiNode for iBGP routing system with route reflection, i.e., $best_p^B$, there are satisfiable assignments for variables in the corresponding topology-based approach, where for any R_p, $best_p^T = best_p^B$.*

Proof. Given satisfiable assignments for variables in BiNode for an iBGP routing system with route reflection, we construct the satisfiable assignments for variables in the topology-based approach and show the solution satisfies the selection function and import and export policies.

$$\forall R_p, best_p^T = best_p^B \tag{29}$$

where $best_p^B$ can be expressed with Equation (23), Equation (24) and Equation (25). Similarly, $best_p^B$ satisfies the import function and selection function. We can show this is true by considering three cases similar to Lemma 3. Obviously, if $best_p^B$ is learned via eBGP session, $C_{p,e}$ is the route announcements applying the same import policy and $best_p$ satisfies the constraints of import policy and export policy. If $best_p^B$ is learned via higher tier iBGP neighbors, $C_{p,q}$ satisfies constraints of import/export policies. If $best_p^B$ is learned via same tier iBGP neighbors, we consider set $\mathbf{S}_d$ and show $C_{p,q}$ satisfies the constraints of import policy and export policy in the following three cases.

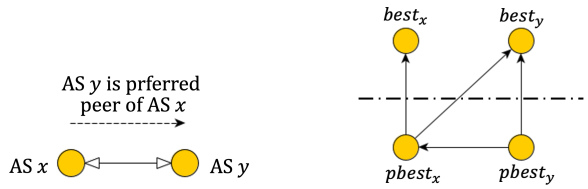
- If $R_q \notin \mathbf{S}_d$, we have $best_q^B$ is learned via lower-tier iBGP or eBGP neighbor. In this case, $best_q = dbest_q$. $C_{p,q}$ satisfies the constraints of import/export policies.
- If $R_q \in \mathbf{S}_d$, we have $best_q^B$ is learned via same or higher tier iBGP neighbors. In this case, $best_q \neq dbest_q$. In this case, suppose there exist $R_k \notin \mathbf{S}$, such that $best_q = C_{q,k}$. As a cluster mentioned above, client route typically has a lower next hop IGP cost than non-client routes. The only way $C_{q,k}$ can have have a higher ranking than $dbest_p$ is that $C_{q,k}.rank_e > dbest_q.rank_e$. As a result, there must be $C_{p,t}$ which is a candidate route for R_q such that $C_{p,t}.rank_e = C_{q,k}.rank_e > C_{p,q}.rank_e$. As a result, $C_{p,q}$ satisfies the constraints of import/export policy.

As a result, $best_p^T$ satisfies the constraints of selection function and import policies and export policies.

In summary, the derived solutions to $best_p$ is a satisfiable assignments for variables in the topology-based approach. Lemma 2 is proved. □

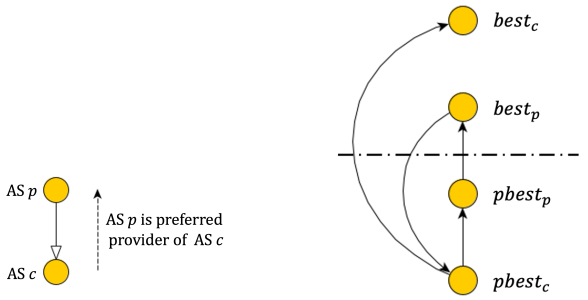
APPENDIX C
BINODE FOR EBGP

This section first illustrates BiNode for ASes violating the Gao-Rexford guideline and then derive constraints from BiNode for eBGP sessions. We prove that BiNode is equivalent to the topology-based approach in Appendix.



(a) An example policy. (b) Dependency graph for (a).

Fig. 21: BiNode for an AS and its preferred peer.



(a) An example policy. (b) Dependency graph for (a).

Fig. 22: BiNode for an AS and its preferred provider.

C.1 BiNode for ASes Violating the Gao-Rexford Guideline

C.1.1 BiNode for Policies Violating the Prefer-Customer Rule

We now consider applying BiNode to networks with routing policies that do not always follow the prefer-customer rule. We still use two nodes in the dependency graph to represent each AS. We can derive the dependency according to the import policies between each pair of neighboring routers.

There are two possible scenarios where an AS can violate the prefer-customer rule. First, an AS might prefer peer routes over customer routes. Second, an AS might prefer provider routes over customer routes. We apply BiNode to these two scenarios as follows.

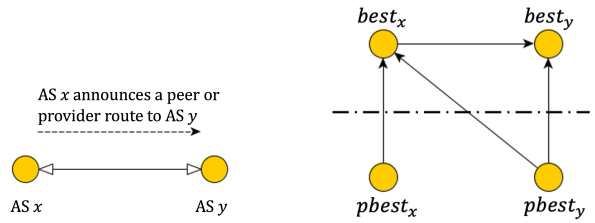
C.1.1.1 Prefer peer routes over customer routes: When an AS prefers routes from a peer over its customer routes, we refer to those peers as *preferred peers*. Figure 21(a) illustrates a pair of ASes, where y is a preferred peer of x. The corresponding dependencies among $best_x$, $pbest_x$, $best_y$ and $pbest_y$ are shown in Figure 21(b).

C.1.1.2 Prefer provider routes over customer routes: When an AS prefers routes from a provider over its customer routes, we refer to those providers as *preferred providers*. Figure 22(a) illustrates a pair of nodes, where p is a preferred provider of c. The corresponding dependencies among $best_p$, $pbest_p$, $best_c$ and $pbest_c$ are shown in Figure 22(b).

C.1.2 BiNode for Policies Violating the Valley-Free Rule

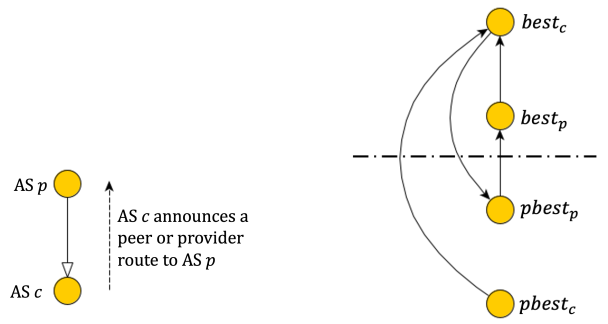
In this section, we consider how to apply BiNode to networks when routing policies do not always follow the valley-free rule. We still use two nodes to represent each AS. According to the export policies between each pair of neighboring routers, we can derive the dependency between them.

There are two possible scenarios where an AS can violate the valley-free rule. First, an AS might export peer or provider routes to its peers. Second, an AS might export peer or provider routes to providers. BiNode for these two scenarios are as follows.



(a) An example policy. (b) Dependency graph for (a)

Fig. 23: BiNode for an AS and its peer to which the AS announces its peer or provider routes.



(a) An example policy. (b) Dependency graph for (a)

Fig. 24: BiNode for an AS and its provider to which the AS announces its peer or provider routes.

C.1.2.1 Exporting peer or provider routes to peers: An AS might announce peer or provider routes to its peers. Figure 23(a) illustrates two ASes, x and y, with PP relationship, where AS x announces peer and provider routes to AS y and the corresponding dependencies among $best_x$, $pbest_x$, $best_y$ and $pbest_y$ are shown in Figure 23(b).

C.1.2.2 Exporting peer or provider routes to providers: An AS might announce peer or provider routes to its providers. Figure 24(a) illustrates two ASes, p and c, with PC relationship, where the customer, AS c, announces peer and provider routes to its provider, AS y, and the corresponding dependencies among $best_p$, $pbest_p$, $best_c$ and $pbest_c$ are shown in Figure 24(b).

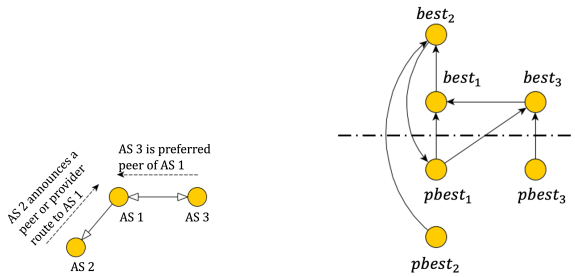
We derive constraints BiNode for eBGP routing system for all three cases discussed in Section 4. We derive the constraints for each of the three cases in two steps. We first derive the constraints for $best_p$ and then derive the constraints for $pbest_p$.

C.1.3 BiNode for An Example Topology

Given the routing policies of ASes, we can derive the dependency of their variables. After that, we can compose all dependencies into one graph for all pairs of neighboring ASes. For example, Figure 25 illustrates BiNode for an AS-level topology in which ASes do not always follow the Gao-Rexford guideline.

C.2 Deriving Constraints for eBGP

We first derive the constraints for eBGP when ASes follow the Gao-Rexford guideline and then derive constraints for



(a) An example policy. (b) Dependency graph of (a).

Fig. 25: BiNode for a topology in which ASes violate the Gao-Rexford guidelines.

ASes violates the Gao-Rexford guideline using BiNode.

C.2.1 BiNode for Policies Following the Gao-Rexford Guideline

Constraints for $pbest_p$:

$$pbest_p = f_{\text{sel}}(\{C_{p,e}|R_e \in N_e(R_p)\} \cup \{C_{p,q}|R_q \in N_c(R_p)\}) \tag{30}$$

where $N_c(R_p)$ represent the set of customer neighbors of R_p. As a result, $C_{p,e}$ and $C_{p,q}$ can be expressed with

$$\begin{aligned} C_{p,e} &= f^{p,eq}_{\text{imp}}(a_{p,e}) \\ C_{p,q} &= f^{p,q}_{\text{imp}}(f^{q,p}_{\text{exp}}(pbest_q)) \end{aligned} \tag{31}$$

Constraints for $best_p$:

$$best_p = f_{\text{sel}}(\{pbest_p\} \cup \{C_{p,q}|R_q \in N_p(R_p)\} \cup \{C_{p,q}|R_q \in N_r(R_p)\}) \tag{32}$$

where $N_p(R_p)$ represents the set of provider BGP neighbors of R_p and $N_r(R_p)$ represents the set of peer BGP neighbors of R_p. As a result, for $R_q \in N_p(R_p)$, $C_{p,q}$ can be expressed with the following equation.

$$C_{p,q} = f^{p,q}_{\text{imp}}(f^{q,p}_{\text{exp}}(best_q)) \tag{33}$$

and for $R_q \in N_r(R_p)$, $C_{p,q}$ can be expressed with the following equation.

$$C_{p,q} = f^{p,q}_{\text{imp}}(f^{q,p}_{\text{exp}}(pbest_q)) \tag{34}$$

In summary, we build the constraints for BiNode for eBGP routing system when policies follow the Gao-Rexford guidelines as follows.

Deriving constraints BiNode for eBGP

```
for each BGP router R_p, construct the
    constraints for the following equation:
  best_p = f_sel({pbest_p} ∪ {C_{p,q}|R_q ∈ N_p(R_p)})
           ∪{C_{p,q}|R_q ∈ N_r(R_p)}
  for R_q ∈ N_p(R_p), construct the constraints
      for the following equation:
      C_{p,q} = f^{p,q}_imp(f^{q,p}_exp(best_q))
   for R_q ∈ N_r(R_p), construct the constraints
       for the following equation:
      C_{p,q} = f^{p,q}_imp(f^{q,p}_exp(pbest_q))
  pbest_p = f_sel({C_{p,e}|R_e ∈ N_e(R_p)} ∪ {C_{p,q}|R_q ∈ N_c(R_p)})
  for R_e ∈ N_e(R_p), construct the constraints
      for the following equation:
     C_{p,e} = f^{p,e}_imp(a_{p,e})
  for R_q ∈ N_c(R_p), construct the following
      constraint:
      C_{p,q} = f^{p,q}_imp(f^{q,p}_exp(pbest_q))
```

C.2.2 BiNode for Policies Violating the Prefer-Customer Rule

We build the constraints for R_p as follows if R_p prefers peer/provider routes over client routes. We use set $N_v(R_p)$ to denote the set of preferred peers/providers.

Constraints for $pbest_p$:

$$pbest_p = f_{\text{sel}}(\{C_{p,e}|R_e \in N_e(R_p)\} \cup \{C_{p,q}|R_q \in N_c(R_p)\} \cup \{C_{p,v}|R_v \in N_v(R_p)\}) \tag{35}$$

As a result, candidate routes $C_{p,v}$ can be expressed with

$$C_{p,v} = f^{p,v}_{\text{imp}}(f^{v,p}_{\text{exp}}(pbest_v)) \tag{36}$$

The remaining candidate routes remain the same and can be expressed with Equation (31).

Constraints for $best_p$:

$$best_p = f_{\text{sel}}(\{pbest_p\} \cup \{C_{p,q}|R_q \in N_p(R_p)\backslash N_v(R_p)\} \cup \{C_{p,q}|R_q \in N_r(R_p)\backslash N_v(R_p)\}) \tag{37}$$

$C_{p,q}$ can be expressed with Equation (33) and Equation (34) accordingly.

C.2.3 BiNode for Policies Violating the Valley-Free Rule

We build the constraints for R_p as follows if R_p has a customer/peer neighbors violating valley-free rule. We use set $N_v(R_p)$ to denote the set of violated neighbors.

C.2.3.1 BGP neighbors in customer ASes violate valley-free rule: We first consider the case where BGP neighbors in customer ASes violate the valley-free rule. That is, $N_v \subset N_c(R_p)$.

Constraints for $pbest_p$:

$$pbest_p = f_{\text{sel}}(\{C_{p,e}|R_e \in N_e(R_p)\} \cup \{C_{p,q}|R_q \in N_c(R_p)\} \cup \{C_{p,v}|R_v \in N_v(R_p)\}) \tag{38}$$

As a result, candidate routes $C_{p,v}$ can be expressed with

$$C_{p,v} = f^{p,v}_{\text{imp}}(f^{v,p}_{\text{exp}}(best_v)) \tag{39}$$

The remaining candidate routes remain the same and can be expressed with Equation (31). The constraints for $best_p$ remain the same.

C.2.3.2 BGP neighbors in peer ASes violate valley-free rule: We then consider the case where BGP neighbors in peer ASes violate the valley-free rule. That is, $N_v \subset N_r(R_p)$. In this case, the constraints for $pbest_p$ remain the same.

Constraints for $best_p$:

$$best_p = f_{\text{sel}}(\{pbest_p\} \cup \{C_{p,q}|R_q \in N_p(R_p)\} \cup \{C_{p,q}|R_q \in N_r(R_p)\} \cup \{C_{p,v}|R_v \in N_v(R_p)\}) \tag{40}$$

The candidate routes $C_{p,v}$ can be expressed with

$$C_{p,v} = f^{p,v}_{\text{imp}}(f^{v,p}_{\text{exp}}(best_v)) \tag{41}$$

C.3 BiNode for eBGP is Equivalent to the Topology-Based Approach

We prove BiNode for eBGP is equivalent to the topology-based approach. We prove they are equivalent by proving Theorem 4.1.

Theorem 4.1. *The best routes derived from BiNode is the same as that from the topology-based approach for eBGP routing system when policies following Gao-Rexford guideline.*

We prove Theorem 4.1 by proving Lemma 5 and Lemma 6.

Lemma 5. *Given satisfiable assignments for variables in the topology-based approach for eBGP routing system where policies following Gao-Rexford guideline, i.e., $best_p^T$, there are satisfiable assignments for variables in the corresponding BiNode, where for any R_p, $best_p^T = best_p^B$.*

Proof. Given a satisfiable assignment of the topology-based approach, we can construct a satisfiable assignment for BiNode as follows. We first derive the variables of the best routes in BiNode from that of the topology-based approach.

$$best_i^B = best_i^T \tag{42}$$

$$pbest_i^B = \begin{cases} best_i^T & \text{if } best_i^T \text{ is a customer route} \\ \text{Equation (32)} & \text{Otherwise} \end{cases} \tag{43}$$

In the following, we show that the derived assignment for SMT variables is a satisfiable assignment for the constraints of BiNode. We first consider that the assignment for BiNode satisfies the constraints of import policy, Equation (32).

- If $best_i^T$ is learned from a customer neighbor R_j, then $best_i^B = pbest_i$ and $C_{i,j}$ is the incoming edge of the downside router of R_i. $pbest_i$ and $C_{i,j}$ satisfy the constraints of import policy, since $C_{i,j}$ is the most preferred customer route of node i. Following the same reason, $best_i$ satisfies the constraints of import policy.
- If $best_i^T$ is learned from a peer neighbor or a provider neighbor R_j, then node i does not receive any customer route and $C_{i,j}$ is the incoming edge of the upside node of node i. Apparently, $pbest_i$ satisfies the constraints of import policy. $best_i^b$ and $C_{i,j}$ satisfy the constraints of import policy, since $C_{i,j}$ is the most preferred route of node i.

In summary, in all the above three cases, the derived solution for $pbest_p^B$ and $best_p^B$ satisfies the constraints in BiNode. Lemma 5 is proved. □

We now prove Lemma 6.

Lemma 6. *Given satisfiable assignments for variables in the topology-based approach, for eBGP routing system where policies following Gao-Rexford guideline, i.e., $best_p^T$, there are satisfiable assignments for variables in BiNode, where for any R_p, $best_p^T = best_p^B$.*

Proof. Similarly, we can derive a satisfiable assignment for variables in a topology-based approach from that in BiNode as follows.

$$best_i^T = best_i^B \tag{44}$$

Apparently, $best_i^T$ and the variables of the route announcements received by node i satisfy the constraints of import policy, Equation (30), since $best_i^B$ is the best route among all route announcements received by node i.

Given a pair of neighboring BGP routers, R_i and R_j, we show that $C_{i,j} = f_{\text{exp}}^{i,j}(best_i^T)$ as follows.

- If $C_{i,j}$ is the outgoing edge of node $pbest_i$, then R_i announces only customer routes to R_j and $C_{i,j} = f_{\text{imp}}^{i,j}(f_{\text{exp}}^{j,i}(pbest_i))$. In that case, R_i announces only customer routes to R_j. If $pbest_i$ is a customer route, then $pbest_i = best_i^B$ and $C_{i,j} = f_{\text{imp}}^{i,j}(f_{\text{exp}}^{j,i}(pbest_i)) = f_{\text{imp}}^{i,j}(f_{\text{exp}}^{j,i}(best_i^T))$. Otherwise, $C_{i,j} = null = f_{\text{imp}}^{i,j}(f_{\text{exp}}^{j,i}(best_i^T))$.
- If $C_{i,j}$ is the outgoing edge of node i, then AS i can announce peer routes or provider routes to AS j and $C_{i,j} = R_{i,j}^b = f_{export}^{i,j}(best_i^b) = f_{export}^{i,j}(best_i^T)$.

In summary, in all the above three cases, $best_p^T$ satisfies the constraints in the topology-based approach. Lemma 6 is proved. □

Similarly, BiNode for eBGP is equivalent to the topology-based approach when policies violate Gao-Rexford guideline when the corresponding accommodation is applied as described in Appendix C.1.1 and Appendix C.1.2.

APPENDIX D
BINODE FOR MULTIPLE NETWORKS

In this section, we derive the constraints for BiNode for multiple networks and prove BiNode for multiple networks is equivalent to the topology-based approach.

D.1 Deriving Constraints from BiNode for Multiple Networks

As shown in Section 4.3, BiNode for multiple networks consists of four parts. In this section, we derive the constraints for each variable from the DGR shown in Figure 9.

Constraints for $pdbest$:

$$pdbest_p = f_{\text{sel}}(\{C_{p,c} | R_c \in N_c(R_p)\}) \tag{45}$$

where $N_c(p)$ represents the set of routers that are eBGP peers of router p and belong to a customer AS.

$$C_{p,c} = f_{\text{imp}}^{p,c}(f_{\text{exp}}^{c,p}(pbest_c)) \tag{46}$$

Constraints for $pbest$:

$$pbest_p = f_{\text{sel}}(\{pdbest_p\} \cup \{C_{p,q} | R_q \in N_i(R_p)\}) \tag{47}$$

where $N_i(R_p)$ represents the set of internal BGP peers of R_p. As a result,

$$C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(pdbest_q)) \tag{48}$$

Constraints for $dbest_p$:

$$dbest_p = f_{\text{sel}}(\{pdbest_p\} \cup \{C_{p,q} | R_q \in N_p(R_p)\}) \tag{49}$$

where $N_p(R_p)$ represents the set of routers that are eBGP peers of R_p and belong to a provider AS.

$$C_{p,q} = f_{\text{imp}}^{p,q}(f_{\text{exp}}^{q,p}(best_q)) \tag{50}$$

Constraints for $best_p$:

$$best_p = f_{\text{sel}}(\{dbest_p\} \cup \{pbest_p\} \cup \{C_{p,q}|R_q \in N_i(R_p)\}) \tag{51}$$

where $N_p(R_p)$ represents the set of routers that are eBGP peers of R_p and belong to a provider AS.

$$C_{p,q} = f^{p,q}_{\text{imp}}(f^{q,p}_{\text{exp}}(dbest_q)) \tag{52}$$

D.2 BiNode is Equivalent to the Topology-Based Approach

We prove BiNode is equivalent to the topology-based approach. We prove they are equivalent by proving the following theorem.

Theorem 4.2. *The best routes derived from BiNode are the same as that from the topology-based approach for multiple networks.*

Similarly, we prove Theorem 4.2 by proving the following two Lemmas.

Lemma 7. *Given satisfiable assignments for variables in the topology-based approach, i.e., $best^T_p$, there are satisfiable assignments for variables in BiNode, where for any R_p, $best^T_p = best^B_p$.*

Proof. Consider any R_p, the best route derived from topology-based approach can be expressed with

$$best^T_p = f_{\text{sel}}(\{C_{p,q}|R_q \in N_e(R_p)\} \cup \{C_{p,k}|R_k \in N_i(R_p)\}) \tag{53}$$

Let's consider two cases of $best^T_p$, $best^T_p$ can be the best route learned via eBGP session or via iBGP session.

Case 1: $best^T_p$ is learned via eBGP sessions:

In this case,

$$best^T_p = f_{\text{sel}}(\{C_{p,q}|R_q \in N_e(R_p)\}) \tag{54}$$

That is

$$\begin{aligned} best^T_p &= f_{\text{sel}}(\{C_{p,q}|R_q \in N_e(R_p)\}) \\ &= f_{\text{sel}}(\{C_{p,q}|R_q \in N_c(R_p)\} \cup \{C_{p,k}|R_k \in N_p(R_p)\}) \end{aligned} \tag{55}$$

where $N_c(R_p)$ represents the set of routers that are eBGP peers of R_p and belong to a customer AS and $N_p(R_p)$ represents the set of routers that are eBGP peers of R_p and belong to a provider or peer AS. We first elaborate $C_{p,q}$ for $R_q \in N_c(R_p)$.

$$C_{p,q} = f^{p,q}_{\text{imp}}(f^{q,p}_{\text{exp}}(best_q)) \tag{56}$$

Note that R_p is a peer or provider of R_p, R_q only announces its best route to R_p if its best route is learned from a customer AS. If so, $best_q = pbest_q$, in this case, $C_{p,q} = C_{p,q}$. Otherwise, $pbest_q = null$ and $C_{p,q} = null$, in this case, we still have $C_{p,q} = C_{p,q}$. As a result, $best^T_p$ follows the following constraint.

$$\begin{aligned} best^T_p = f_{\text{sel}}(&\{C_{p,q}|R_q \in N_c(R_p)\} \\ &\cup \{C_{p,k}|R_k \in N_p(R_p)\}) \end{aligned} \tag{57}$$

Note that by definition, $\{C_{p,q}|R_q \in N_c(R_p)\}$ is learned from $pdbest_p$, therefore, we have $best^T_p$ follows the following constraint.

$$\begin{aligned} best^T_p &= f_{\text{sel}}(\{pdbest_p\} \cup \{C_{p,k}|R_k \in N_p(R_p)\}) \\ &= f_{\text{sel}}(\{dbest_p\}) \end{aligned} \tag{58}$$

Recall that in this particular case, $best^T_p$ is learned via eBGP sessions, as a result, we have

$$\forall R_q \in N_i(R_p), dbest_p.rank > C_{p,q}.rank \tag{59}$$

and

$$dbest_p.rank > pdbest_p.rank \tag{60}$$

To show $best^T_p$ satisfies constraints in BiNode, we only need to show

$$\begin{aligned} best^T_p = f_{\text{sel}}(&\{dbest_p\} \cup \{pbest_p\} \\ &\cup \{C_{p,q}|R_q \in N_i(R_p)\}) \end{aligned} \tag{61}$$

which is true because we can add candidate route with a lower ranking without changing the output of the selection function.

Case 2: $best^T_p$ is learned via eBGP sessions:

We have already proved this in previous sections. If R_p is in a fully meshed iBGP system, we have proved $best^T_p$ satisfies constraints in BiNode in Appendix A. If R_p is in a iBGP system with route reflection, we have proved $best^T_p$ satisfies constraints in BiNode in Appendix B. □

Lemma 8. *Given satisfiable assignments for variables in BiNode, i.e., $best^B_p$, there are satisfiable assignments for variables in the corresponding topology-based approach, where for any R_p, $best^T_p = best^B_p$.*

Proof. Consider the best routes derived from BiNode for R_p, we have

$$best^B_p = f_{\text{sel}}(\{dbest_p\} \cup \{pbest_p\} \cup \{C_{p,q}|R_q \in N_i(R_p)\}) \tag{62}$$

Let's consider three cases of $best^B_p$, $best^B_p$ can be $dbest_p$, $pbest_p$ or one of the candidate route from $\{C_{p,q}|R_q \in N_i(R_p)\}$.

Case 1: $best^B_p = dbest_p$

In this case, suppose $best^B_p$ is learned from $best_k$, we have

$$\forall R_q \in N_i(R_p), C_{p,k}.rank > C_{p,q}.rank \tag{63}$$

and

$$best_k.rank_e > pbest_p.rank_e \tag{64}$$

As a result, to show $best^B_p$ satisfies constraints in topology-based approach, we have to show

$$best^B_p = f_{\text{sel}}(\{C_{p,q}|R_q \in N_i(R_p)\} \cup \{C_{p,k}|R_k \in N_i(R_q)\}) \tag{65}$$

which is true because the selection function won't change the input when adding lower ranking candidate routes.

Case 2: $best^B_p = pbest_p$

Similar to case 1, $best^B_p$ satisfies constraints in topology-based approach in this case.

Case 3: $best^B_p$ is one of the candidate route from $\{C_{p,q}|R_q \in N_i(R_p)\}$

We have already proved this in previous sections. If R_p is in a fully meshed iBGP system, we have proved $best^B_p$ satisfies constraints in BiNode in Appendix A. If R_p is in a iBGP system with route reflection, we have proved $best^B_p$ satisfies constraints in BiNode in Appendix B. □