Correctness Properties for Internet Routing*

Nick Feamster
College of Computing
Georgia Tech
feamster@cc.gatech.edu

Hari Balakrishnan
Computer Science & AI Lab
MIT
hari@csail.mit.edu

Abstract

This paper motivates and presents a correctness specification for Internet routing. This specification is based on three properties—route validity, path visibility, and safety. This specification may be of use to people developing tools to check routing configurations, to people designing solutions to specific problems in the current system, and to designers of new protocols and routing architectures, all of whom can benefit from knowing what it means for Internet routing to be "correct".

1 Introduction

Today's Internet routing infrastructure is unacceptably fragile. Among its shortcomings, it converges slowly [17] (and sometimes not at all [14]); it is often misconfigured [8, 18]; it is hard to control and predict [10]; and it has weak security properties [22]. This fragility causes communication on the Internet to be unreliable and unpredictable.

Two important design goals for Internet routing, *policy expressiveness* and *scalability*, lead to this fragility. Each goal is important because Internet service is not the purview of a single administrative entity. Rather, the Internet is composed of thousands of independently operated networks that often compete with one another for customers but must nonetheless cooperate to offer global connectivity to hundreds of millions of hosts. This "competitive cooperation" requires the ability to express *routing policy*, which dictates which routes are advertised to which neighbors, and which paths are used to send packets to any given destination. The Internet's scale (in terms of number of total hosts, number of independently operated networks, and the size of any given network) requires that the Internet's routing protocols also incorporate various scaling techniques.

In practice, multiple routing protocols are used to achieve these goals. First, the Border Gateway Protocol (BGP) [24, 25], which is used in two different modes: a network's egress (or "border") routers use *external BGP (eBGP)* to exchange routes for external destinations (*i.e.*, those that are not contained within the current network) with neighboring networks; *internal BGP (iBGP)* is used to disseminate these routes within any given network. Second, interior routing protocols (also called interior gateway protocols, or IGPs) exchange routes for internal destinations. BGP route selection determines which egress router each router sends traffic to. The IGP is then responsible for determining the internal route from that router to the appropriate egress router.

These two classes of routing protocols facilitate both complex policy expression and scalability, but they also interact with each other in unexpected and undesirable ways. These interactions make reasoning about the behavior of the Internet routing system difficult. The lack of any formal reasoning framework leads to *ad hoc* fixes to observed problems that ultimately only worsen this complexity. Today, network operators (who continually tweak routing configuration and handle problems *reactively* after they occur) and protocol designers (who repeatedly propose "point" solutions to various problems) have no way of reasoning about whether their modifications to Internet routing will operate as intended.

There are several possible ways to improve this situation, including fixing problems in the current architecture, improving the ways in which current networks are operated, and designing a more robust architecture for the future. In all cases, however, we need a *specification* of what it means for Internet routing to be "correct", and on ways to tell if the routing system is performing incorrectly or poorly. This paper presents three fundamental correctness properties and defines them formally, with a view toward developing a high-level correctness specification for large-scale Internet routing.

Using the formal specification of these properties, we derive various constraints that today's Internet routing protocols must satisfy to prevent these properties from being violated. While, in this paper, we apply our correctness specification undesirable interactions between iBGP and IGP, we intend the specification to be useful for analyzing any complex interactions that arise in scalable, policy-based routing. Due to space limitations, we have omitted proofs in this paper; proofs, as well as more detailed discussions, are available in [7].

Our proposal for a routing specification is based on the following properties:

1. *Route validity*, which states that if a router has a

*This research is supported in part by the National Science Foundation under grants 0520241 and 0225660, and by a grant from Cisco Corporation. The views expressed in this paper are not necessarily those of these organizations.

route to a destination, then a usable path corresponding to that route exists in the underlying topology. If route validity is violated, then end users could experience a failure of end-to-end connectivity, because routers could forward packets along nonexistent paths.

2. *Path visibility*, which states that if there is a usable path between two nodes, then the routing protocol will propagate information about that path. A failure of path visibility could disrupt end-to-end communication by preventing two connected nodes from learning routes between one another.
3. *Safety*, which states that the routing protocol converges to a stable route assignment, regardless of the order in which routing messages are exchanged. A routing protocol that violates safety will induce persistent route oscillations, causing routing changes that are unrelated to changes in topology or policy.

This paper defines and investigates these properties and demonstrates how they can deepen our understanding of network routing. This correctness specification addresses *static* properties of network routing, not dynamic behavior (*i.e.*, its response to changing inputs, convergence time, etc.). Internet routing, like any distributed protocol, may experience periods of transient incorrectness in response to changing inputs, but we are concerned with persistent misbehavior.

2 Preliminaries

Before introducing the correctness properties themselves, we first introduce some basic terminology for routing. We explain these terms in the context of Internet routing and BGP. We first define paths and routes in terms of a graph $G = (V, E)$, where the nodes in $V = \{v_1, \ldots, v_N\}$ correspond to IP-level nodes (*i.e.*, routers and end hosts) and the edges in E corresponds to IP-level links between those nodes.

2.1 Paths and Routes

We now define two basic terms—path and route—and explain how they are related.

Definition 2.1 (Path) *A* path *is a sequence of nodes* $P = (v_0, \ldots, v_n)$*, where* $v_i \in G$ *for all* $0 \leq i \leq n$*.*

The definition of a path does not constrain how the sequence of nodes is actually constructed. As such, a path might represent a sequence of directly connected IP-layer nodes or endpoints of a tunnel.[1] Note that deleting some nodes from a path still results in a path.

[1] A *tunnel* is a sequence of nodes that all forward packets to some intermediate node (*i.e.*, the tunnel's "exit"), rather than the ultimate destination. A tunnel may be implemented by a variety of mechanisms, such as IP-in-IP encapsulation [6], Multiprotocol Label Switching (MPLS) [4, 21], etc.

In contrast to a path, a route is *information* that allows nodes in G to construct paths to *destinations*. The *destination* d may refer either to a single node or a group of nodes (named, for example, by an IP prefix). The purpose of a routing protocol is to propagate routes for destinations. Collectively, the routes to d that the nodes in G ultimately select define the *path* from any node in G to that destination. All of our definitions presume that the handle for a destination, d, cannot be manipulated (*e.g.*, our definitions do not consider the effects of IP prefix aggregation [13, 23]).

Definition 2.2 (Route) *A* route *is a mapping* $(d \rightarrow v_i)$*, where* d *is a destination, and* $v_i \in G$ *is a node en route to the destination* d*.*

We say that $v_i \in d$ if the destination d includes v_i. A route $(d \rightarrow v_i)$ received by v_j indicates that, if v_j has a packet to send to some node at destination d, it can forward that packet to v_i, which in turn ought to have a route to d (whereupon this process repeats until the data reaches d). One can think of a route $(d \rightarrow v_i)$ being used at node v_j as *inducing* a path, $(v_j, \ldots, v_i)$, where either v_j and v_i are directly connected or where the actual nodes along that path segment are determined by the connectivity between those nodes, as established by the IGP or using tunnels. We will formalize the notion of induced paths in Section 2.2.

Note that Definition 2.2 can apply to any routing protocol, not just to BGP. In an IGP, the node v_i is typically the router that is immediately connected at the IP layer. In BGP, however, (particularly in iBGP) the next hop may be several IP-layer hops away. In BGP, a node that receives a route $(d \rightarrow v)$ but is not directly connected to v must rely on the IGP for reachability to v.

2.2 Induced Paths

We can think of paths as being *induced* by routes. That is, while there exist many sequences of nodes between any node v_0 and a node in some destination d, the path that traffic will actually take from v_0 en route to d is determined by the routes that the nodes in G select. In many routing protocols, including BGP, no single node has knowledge about the entire sequence of nodes that traffic traverses en route to d; rather, the nodes that the routes select collectively *induce* a path to d. We are interested in making statements about those induced paths. We now formalize the concept of induced paths and describe a special class of induced paths called *consistent paths*.

Definition 2.3 (Induced path) *Let* $r_{v_j}(d)$ *be the route that node* v_j *selects en route to* d (i.e., *it is a mapping* $(d \rightarrow v_k)$ *for some other* $v_k \in G$*. Then, the* path induced *by route* $r_{v_0}(d) : (d \rightarrow v_i)$*,* $P_{v_0}(r_{v_0}(d))$*, is:*

$$P_{v_0}(r_{v_0}(d)) = \begin{cases} \phi \text{ if } v_0 \text{ has no route to } d \\ v_0 \text{ if } v_0 \in d \\ (v_0, P_{v_1}(r_{v_1}(d))) \text{ otherwise} \end{cases}$$

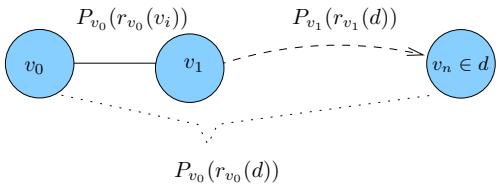

Figure 1: Illustration of an induced path from v_0 to d, collectively induced by the selection of a route to d at every node along the path. The node v_i may be immediately adjacent to v_0 (as shown), but it may also be several hops away.

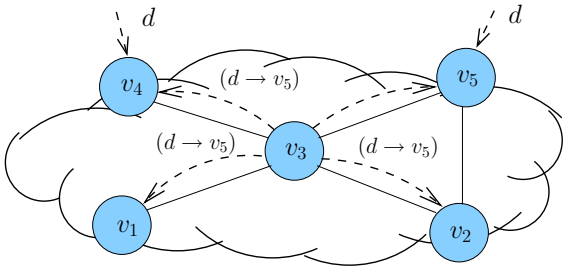

Figure 2: An example that demonstrates how BGP routes induce paths. Dashed lines are iBGP sessions from route reflectors to clients. Solid lines show IGP links.

where v_1 is defined according to $r_{v_0}(v_i) : (d \to v_1)$; that is, v_1 is the next-hop node in v_0's forwarding table for destination v_i.

Figure 1 illustrates an induced path and its constituent subpaths. The node v_i in the induced path may either be adjacent to v_0 in G, or it may be several hops away. When v_i is adjacent to v_0 in G, data traffic can reach v_i from v_0 via a direct IP link. When v_i is several hops away in G, however, v_0 must rely on intermediate nodes to forward traffic to v_i.

2.3 Consistent Paths

In the case of Figure 1, the nodes between v_0 and v_i could use routes that induce paths that never even traverse v_i. In other words, the path that is described by $(v_0, P_{v_0}(r_{v_0}(v_i)))$ may traverse an intermediate node whose induced path to d does not traverse v_i. To precisely classify the types of paths for which this inconsistency does not arise, we define the notion of a consistent path.

Definition 2.4 (Consistent path) *An induced path $P_{v_0}(r_{v_0}(d)) = (v_0, \ldots v_n)$ to d is consistent if, (1) for all $1 \leq i \leq n$, $P_{v_i}(r_{v_i}(d)) = (v_i, v_{i+1}, \ldots, v_n)$; (2) $P_{v_0}(r(v_0(d)))$ contains v_i, where $r_{v_0}(d) : d \to v_i$.*

Consistent paths are an important class of paths because inconsistent paths can sometimes give rise to forwarding loops. A *forwarding loop* is a special case of an inconsistent path where some intermediate node's induced path includes a node that has already appeared on the path.

When v_0 and v_i are not adjacent, ensuring that all intermediate nodes select a route $(d \to v_i)$ will guarantee that an induced path is consistent. If an intermediate node selects some route $(d \to v_j)$ where $v_i \neq v_j$, then the induced path to d may never traverse v_i. If, on the other hand, the intermediate node selects a route $(d \to v_i)$, then the induced path from that node to v_i will be a subpath of $P_{v_0}(r_{v_0}(v_i))$, assuming all nodes use the same function to induce paths (*e.g.*, if the induced path to v_i is based on shortest paths routing, as in an IGP).

We now briefly discuss paths and routes in the context of BGP. To illustrate the distinction between routes and paths, we examine their definitions within the context of BGP routing within a single AS. In this case, a *route* is of the form $(d \to v_i)$, where d is an IP prefix and v_i is the BGP "next hop" (a node that need not be directly connected at the IP layer). The *path* that traffic ultimately takes from some node v_j to the destination d, for which v_j has a route $(d \to v_i)$, depends on how connectivity is established between v_j and v_i. If v_j and v_i are in two different ASes, then they are typically directly connected. If they are in the same AS, however, it is common for v_i to be the IP address of an egress (or "border") router and for v_j to be several IP hops away. The induced path between v_j and v_i may be determined by a tunnel, by a shortest paths routing protocol, using static routes, etc.

If the induced path between v_j and v_i is not defined by a tunnel, then the nodes between v_j and v_i will use their own routes for forwarding data to d. In this case, the induced path to d is actually determined by "stitching together" these constituent induced paths. If all nodes between v_j and v_i select routes that indicate that traffic to d should be sent via v_i, then the induced path between v_j and v_i will be consistent. Otherwise, the path could be arbitrary; in fact, it might never traverse v_i.

Figure 2 shows an example that illustrates this distinction. In this example, all routers in the AS are clients of the route reflector, v_3; solid lines show the edges in the IGP graph, and all edges have a cost of 1. Suppose that v_3 learns two routes to d and selects the route that it receives from v_5. In this case, v_3 propagates that *route* (*i.e.*, $(d \to v_5)$) to all of its clients, as shown. Using that route, each node ultimately uses a different *path* to the egress router, v_5. For example, v_1's shortest IGP path to v_5 is v_1, v_3, v_5, whereas v_2's shortest path to v_5 is v_2, v_5. Even if a node, say v_1 selects a BGP route with the "next hop" v_5, there is no guarantee that the resulting *induced path* will traverse v_5. If an additional node, v_6, had been on the path between v_1 and v_3, and had instead selected a route $(d \to v_4)$, then v_1's path to d through the AS could have in fact been v_1, v_6, v_4.

2.4 Policy

A noteworthy aspect of Internet routing is that it is *policy-based*. The job of the routing protocol is not to propagate complete information about the topology, but

to only propagate information about paths that comply with the various economic and policy goals of each AS. We must therefore qualify paths in the topology according to those that comply with such these policies and those that do not.

Definition 2.5 (Policy) *A* policy *is a function* $\mathcal{P}(s, v_{i-1}, v_i, v_{i+1}, d) \rightarrow (0, 1)$*, where* s *is a source,* v_{i-1}*,* v_i*, and* v_{i+1} *are nodes on a path* $(v_0, v_1, \ldots, v_n)$*,* d *is a destination, and* $\mathcal{P}$ *is defined as follows:*

$$\mathcal{P}(s, v_{i-1}, v_i, v_{i+1}, d) = \begin{cases} 1 \text{ if } i = 0 \text{ and } v_0 \text{ forwards packets from source } s \text{ destined for } d \\ 1 \text{ if } 0 < i < n \text{ and } v_i \text{ forwards packets with source } s \text{ from } v_{i-1} \text{ destined for } d \text{ via } v_{i+1} \\ 1 \text{ if } i = n \text{ and } v_n \text{ forwards packets with source } s \text{ destined for } d \\ 0 \text{ otherwise} \end{cases}$$

The function $\mathcal{P}(v_{i-1}, v_i, v_{i+1}, d)$ is not expressive enough to capture all policies, but, as we will see, it is general enough to capture the policies that are commonly expressed in Internet routing. Other routing protocols may require more expressive policy functions. Our intent here is not to define a policy function that captures all policies, but rather to allow us to define a policy-conformant path in the context of Internet routing.

Definition 2.6 (Policy-conformant path) *A path* $(v_0, v_1, v_2, \ldots, v_n)$ *is policy-conformant for source* s *and destination* d *if* $\mathcal{P}(s, v_{i-1}, v_i, v_{i+1}, d) = 1$ *for all* $0 \leq i \leq n$*.*

For simplicity, we assume that paths for which the source, destination, and all nodes in between the source and destination are in the same AS are policy-conformant.

Although the policy function is defined at the level of nodes, it is in fact expressive enough to capture many AS-level policies that network operators commonly want to express. For example, suppose an operator wants to express that AS Y should not forward traffic between two other ASes, X and Z, for some destination d. Recall that a path with some nodes removed still constitutes a path. As such, it is possible to express this policy in terms of the nodes in ASes X and Z along the path. For example, the policy can be expressed as $\mathcal{P}(s, v_X, v_i, v_Z, d) = 0$, where v_X is a router that has an eBGP session to AS X, and so forth. In a more complicated scenario, if AS Y has multiple nodes that are adjacent to nodes in ASes X and Z, the AS-level policy would be expressed as an enumeration over node-level policies.

3 Route Validity

In this section, we motivate and describe *route validity*. Informally, route validity says that any route that the routing protocol propagates should correspond to a usable path in the topology. Route validity concerns the properties of the paths induced by the routes that the routing protocol propagates.

Definition 3.1 (Route validity) *A route for a destination* d *is valid if, and only if, the path induced by the route (1) is consistent, (2) is policy-conformant for all sources that use the route, and (3) terminates at* d*. We say that a routing protocol satisfies route validity if the protocol propagates only valid routes for all destinations.*

Because a source v_0 and a destination d may be in different ASes, guaranteeing that BGP satisfies route validity is difficult in practice. Determining both the induced path to d and determining whether that path is policy-conformant requires knowledge of the configurations of multiple ASes. Fortunately, the properties of route validity (*i.e.*, consistency and policy-conformance) are composable.

Observation 3.1 *Composing a path by concatenating two consistent, policy-conformant paths results in a new consistent, policy-conformant path.*

This observation implies that if the routing protocols in each AS en route to a destination induce only consistent and policy-conformant paths to some destination d, then BGP will only induce consistent, policy-conformant paths for that destination d. For the purposes of this paper, we assume that all paths are policy-conformant, because detecting violations of policy are difficult to verify in practice. We also assume that all eBGP sessions are point-to-point (*i.e.*, immediately connected at the IP layer), which is almost always the case in practice: service providers typically apply policies at AS boundaries, rather than on paths within an AS. Finally, we assume that the IGP already satisfies route validity; detecting route validity faults in internal routing protocols is beyond the scope of this work.

Modulo policy-conformance, guaranteeing that BGP satisfies route validity boils down to ensuring that iBGP always induces consistent paths within each AS. Guaranteeing this property is the focus of the remainder of this section. We first focus on how to guarantee that "full mesh" iBGP configurations (and protocol modifications that would allow every router to the complete set of "best" eBGP-learned routes) always induce consistent paths; we then derive conditions on iBGP configurations that use route reflection that guarantee that iBGP only induces consistent paths.

3.1 Case #1: Every router learns all "best" eBGP routes.

If different routers within an AS receive different sets of candidate routes for some destination d, then the routers along a path from v_0 to v_i may not ultimately select the route $(d \rightarrow v_i)$. It turns out that the default iBGP configuration, where every eBGP-speaking router has an iBGP

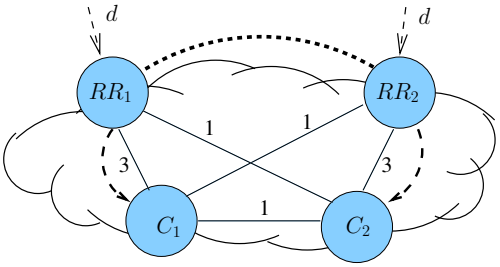

Figure 3: The interaction of IGP and route reflection may cause forwarding loops [5].

session with every other eBGP-speaking router in the AS (*i.e.*, a "full mesh" iBGP configuration) satisfies route validity.

Theorem 3.1 *If (1) every router learns the BGP routes selected by the complete set of eBGP-speaking routers, and (2) iBGP-speaking routers do not modify route attributes (*i.e.*, local preference, origin type, MED, or next hop), then all paths induced by iBGP (within the AS) will be consistent.*

A full mesh iBGP configuration can guarantee the first condition of Theorem 3.1. Another approach to ensuring that every router learns the set of routes selected by the complete set of eBGP-speaking routers is to alter how route reflectors readvertise routes to their clients. By a similar argument as in the proof of Theorem 3.1, such a modification would cause iBGP to induce only consistent paths. Such a configuration not only guarantees consistent paths, but it also prevents certain types of persistent route oscillation [1]. Unfortunately, implementing such a configuration in practice requires modifying the large deployed base of BGP routers. Alternatively, an architecture such as either the Routing Control Platform (RCP) [3, 9] or the recent proposal for more versatile route reflectors [2] could implement this type of iBGP configuration.

3.2 Case #2: Each router learns only some "best" eBGP routes

If full mesh iBGP were the only intra-AS BGP configuration, guaranteeing that iBGP satisfied route validity would be easy. Unfortunately, this technique does not scale to large ASes because it requires $O(|R|^2)$ iBGP sessions, where $|R|$ is the number of routers in the AS. Large ASes typically use a technique called route reflection, where a single route reflector selects a route on behalf of its client routers.

Guaranteeing route validity in an iBGP topology with route reflectors is not easy. Previous work has observed that the interactions between the IGP topology and an iBGP topology with route reflectors can give rise to route validity violations [5]. Figure 3 shows one such example. Route reflectors RR_1 and RR_2 both receive a route to destination d and have clients C_1 and C_2 respectively. Hence, C_1 may receive and select the route $(d \rightarrow RR_1)$, and C_2 may receive and select the route $(d \rightarrow RR_2)$. If the shortest IGP path (*i.e.*, the induced path) between A and RR_1 is via B, and the shortest IGP path between B and RR_2 is via A, then traffic en route to d that traverses either router A or B will be caught in a persistent *forwarding loop*: that is, traffic destined for d will never reach d but instead will repeatedly visit a cycle of two or more nodes. A forwarding loop is simply a special case of a route validity violation.

Our goal is to detect whether a configuration of route reflectors and clients induces only consistent paths with a simple algorithm that examines only the static iBGP and IGP topologies. One such sufficient condition that guarantees this property requires that the iBGP topology be *RR-IGP-Consistent*, defined as follows:

Definition 3.2 (RR-IGP-Consistent) *A route reflector configuration is* RR-IGP-Consistent *if, for all nodes, every shortest IGP path between that node and its possible egress nodes (*i.e.*, the set of eBGP-speaking routers) traverses that node's route reflector before any other node's route reflector and the egress node's route reflector before the egress node itself.*

Theorem 3.2 *If an iBGP configuration is* RR-IGP-Consistent*, then all paths induced by iBGP are consistent.*

Although this result is a helpful sufficient condition, it does not guarantee that route validity will be satisfied when arbitrary links fail, thus causing shortest IGP paths to change. Designing an *RR-IGP-Consistent* iBGP topology that is robust to link failures is a difficult task. Recent work has proposed using graph separators as a way of efficiently placing route reflectors in an iBGP topology to guarantee that route validity is satisfied [27].

4 Path Visibility

Path visibility says that if there exists one or more policy-conformant paths between two nodes, then the routing protocol should propagate routes that induce at least one of those paths. Path visibility is an important property for a number of reasons. First, if a routing protocol satisfies this property, then every node is guaranteed to have the necessary information to reach all other nodes.

Definition 4.1 (Path visibility) *A routing protocol satisfies* path visibility *if, for all $v_0 \in V$ and for all destinations d, the existence of a policy-conformant path $P = (v_0, \ldots, v_n)$ implies that v_0 learns a valid route $(d \rightarrow v_j)$ for some $0 \leq j \leq n$.*

Path visibility states that if there is a policy-conformant path from v_0 to d, then v_0 should learn *at least one* valid route to d. Note that the definition does not require v_0 to learn all routes to d, nor does it require that v_0 learn the "best" route to d by any metric.

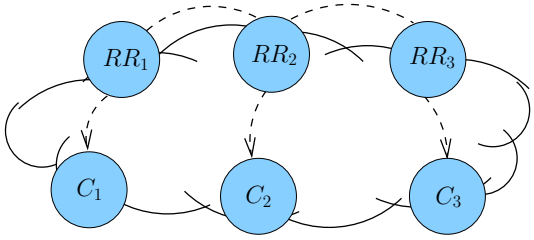

Figure 4: An iBGP topology that violates path visibility.

Path visibility also does not require that the route that v_0 learns correspond to the actual path that traffic takes from v_0 to d.

By definition, path visibility violations result when some router fails to propagate usable routes. These failures in route propagation range from the mundane (*e.g.*, misconfigured filters that fail to install or advertise routes for a policy-conformant path) to the subtle (*e.g.*, errors in iBGP configuration).

Because of the way iBGP readvertises routes, an arbitrary iBGP configuration is not guaranteed to satisfy path visibility. In fact, even the very simple iBGP topology in Figure 4 does not satisfy path visibility: if the route reflector RR_1 (or its client, C_1) receives a route for some destination via an eBGP session, then neither RR_3 nor C_3 will receive a route to the destination, and vice versa.

Path visibility is important because it ensures that, if the network remains connected at lower layers, the routing protocol does not create any new network partitions. Path visibility also reduces the likelihood of suboptimal routing. For example, in Figure 4, even if all clients learned *some* route to the destination via eBGP, the clients would not be guaranteed to discover the *best* route to the destination (*e.g.*, if a client of the route reflector on the far left learned a route with a shorter AS path, neither the route reflector on the far right nor its clients would learn it). As such, it is important that an AS's iBGP configuration satisfy path visibility. In the remainder of this section, we derive the constraints on the iBGP configuration that must be satisfied to guarantee path visibility. We first consider iBGP topologies that do not employ route reflection.

Theorem 4.1 *For an iBGP topology without route reflectors, satisfying path visibility requires a full mesh iBGP configuration.*

In large networks, a route reflector may itself be a client of another route reflector. Any router may also have "normal" (*i.e.*, peer) iBGP sessions with other routers. We use the set of reflector-client relationships between routers in an AS to define a graph $\mathcal{I}$, where each router is a node and each session is either a directed or undirected edge: a client-reflector session is a directed edge from client to reflector, and peer iBGP sessions are undirected edges. We say that $\mathcal{I}$ is *acyclic* if $\mathcal{I}$ has no sequence of directed and undirected edges that form a cycle. In typical iBGP hierarchy designs, $\mathcal{I}$ is acyclic (previous work states that $\mathcal{I}$ should be acyclic to prevent protocol oscillations [15]—and it is a good design decision anyway—although we will see in Section 5 that this constraint is unnecessary). We now define the topological constraints on $\mathcal{I}$ to guarantee path visibility.

Theorem 4.2 *Suppose that the graph defined by an AS's iBGP relationships, $\mathcal{I}$, is acyclic. Then, $\mathcal{I}$ does not have a signaling partition if, and only if, the eBGP-speaking routers that are not route reflector clients form a clique.*

The results in this section suggest that path visibility can be guaranteed by checking relatively simple constraints on the iBGP topology, which can be determined by analyzing the static configuration files alone. Although, in the long term, architectural changes could be made to guarantee that no configuration ever violates path visibility [2, 3, 9], relatively simple checks against routing configuration can guarantee path visibility today.

5 Safety

Violations of safety can cause the routing protocol to continually send routing updates that do not reflect changes in the underlying topology.

Definition 5.1 (Safety) *A routing protocol satisfies* safety *if and only if, given no changes to available paths after time t_0, then, at some finite time $t_s > t_0$, each node $v \in G$ selects some route r and does not select a route $r' \neq r$ for any $t > t_s$.*

Safety is an important property because it guarantees that changes in *routes* (*i.e.*, routing update messages) correspond directly to changes in available *paths*. This invariant is important for several reasons. First, if the routing protocol causes routers to change routes unnecessarily (*i.e.*, when the paths are in fact stable), the protocol itself may cause performance degradations, such as lost or reordered packets. Second, if routing changes do not correspond to changes in the actual topology, then debugging the cause of an oscillation becomes more difficult, because an operator cannot determine whether routing changes reflect problems with infrastructure (*e.g.*, flaky or failing equipment) or the routing protocol itself.

Safety problems arise for two reasons:

1. Conflicting route selections within the same AS, caused by interactions between BGP route attributes and the IGP (iBGP safety).
2. Conflicting rankings, caused by conflicting policies between ASes (eBGP safety).

In both cases, guaranteeing safety is hard. The rest of this section focuses on the constraints for guaranteeing safety in iBGP. Guaranteeing that eBGP satisfies safety

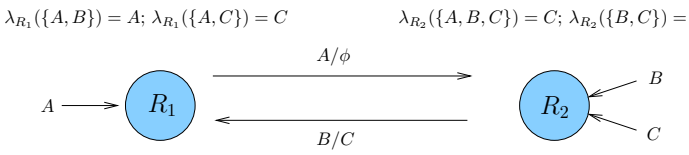

Figure 5: λ_{R_2} does not satisfy determinism. This violation can causes a safety violation.

requires either knowing the rankings of ASes across the *global* Internet (not a realistic requirement, because ASes typically insist on keeping their rankings private) or placing restrictions on how each AS can specify rankings and filters. This problem is the focus of other recent work [11].

Safety violations in iBGP occur because BGP's route selection process does not satisfy *determinism*. Determinism essentially says that the route each router ultimately selects should not depend on either (1) the presence or absence of routes that would not be selected in the first place or (2) the order in which messages arrive. We formally define determinism and explain why guaranteeing this property is difficult in practice.

Definition 5.2 (Selection function) *A selection function at router r, λ_r, takes as input a set of routes $R_d = \{r_1, \ldots, r_n\}$ for some destination d and produces a single route $r_i \in R_d$. The route r_i is often called the router's "best route" to d.*

Definition 5.3 (Determinism) *A routing protocol satisfies* determinism *for destination d if, for all routers r, if r has a set of routes R_d to d, $\lambda_r(R_d)$ satisfies the following two properties:*

1. *$\lambda_r(R_d) = \lambda_r(R'_d)$, where R'_d is any subset of R_d that contains $\lambda_r(R_d)$, and*
2. *$\lambda_r(R_d)$ does not depend on the order in which the routes in R_d arrived at router r.*

Determinism depends only on the selection function, λ_r, for all routers r. Thus, we may also discuss a single selection function, λ_r, in terms of whether it satisfies determinism.

Determinism is important for predictability; moreover, as the following observation shows, violations of determinism can induce safety violations, even when the selection function of only one router violates determinism.

Observation 5.1 *If the selection function of even one router, λ_r, violates determinism, then the routing protocol may also violate safety.*

The following example illustrates this point.

Example 5.1 Consider Figure 5. Router R_1 selects route A given the choices $\{A, B\}$ and selects route C given choices $\{A, C\}$. This selection function satisfies determinism. On the other hand, router R_2's selection function violates determinism: $\lambda_{R_2}(\{A, B, C\}) = C$, but $\lambda_{R_2}(\{B, C\}) = B$. The interaction of the two selection functions creates the following oscillation:

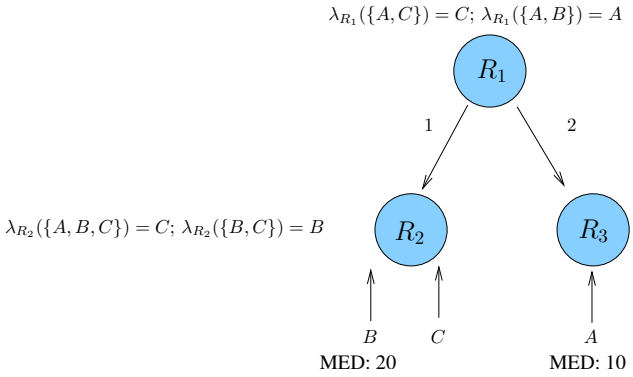

Figure 6: Instantiation of Figure 5 in a BGP configuration. Router 1 is a route reflector with two clients, R_2 and R_3. Costs on edges are IGP path costs. Router R_2 prefers route B over route C due to a tiebreak.

1. Router R_1 receives only route A, selects it, and advertises this route to router R_2.
2. Router R_2 has received $\{A, B, C\}$, selects route C, and advertises it to router R_1.
3. Router R_1 has received $\{A, C\}$, selects route C, and sends a withdrawal (ϕ) for route A to router R_2.
4. Router R_2 selects B from the set $\{B, C\}$ and advertises it to router R_1, implicitly withdrawing route C.
5. Router R_1 now has to select a route from the set $\{A, B\}$, selects route A, and advertises it to router R_2.

This process repeats forever, violating safety. ■

5.1 Determinism Violations in BGP: The MED Attribute

It turns out that the above scenario can occur in BGP, because the MED attribute causes each router's selection function to violate determinism. The addition of a third router, as shown in Figure 6, gives rise to the oscillation from the previous example. In this case, router R_1 is a route reflector for two clients: routers R_2 and R_3, with IGP costs as shown. Routes A and B are advertised by the same AS, and route A has a lower MED value (and, hence, is preferred to B). In this setup, the selection functions are exactly as described in the from Figure 5: when router R_2 learns $\{A, B, C\}$, route B is eliminated due to MED, and route C is selected because it is an eBGP-learned route. When router R_2 learns only $\{B, C\}$, on the other hand, it prefers route B over route C due to the router ID tiebreak. Similarly, router R_1 prefers route C over route A due to IGP, but router A over router B due to MED. The routing system in this example oscillates in the same fashion as the one shown in Figure 5.

As the above example shows, the interaction between the MED attribute and route reflection can cause BGP to violate safety. Note that this example satisfies the guidelines that were specifically proposed to avoid these

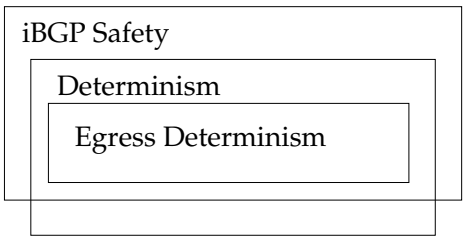

Figure 7: The relationship between determinism, egress determinism, and safety.

types of oscillations [20]. Even though not all safety violations are caused by violations of determinism, eliminating BGP's determinism problem can eliminate all oscillations that do not involve cyclic preferences over routes caused by setting the local preference attribute. Specifically, by making the MED attribute comparable across *all* routes, rather than just those from the same AS, each router's selection function can be made to satisfy determinism. We now formally show this result.

Lemma 5.1 *If a router's selection function compares the MED attribute across all routes to a destination (rather than just across those from the same neighboring AS), then its selection function satisfies determinism.*

We explore how comparing the MED attribute across all routes affects protocol operation, as well as how this might be done in practice, in recent work [12]. In short, the primary benefit of making the route selection function deterministic is that a set of routers *within a single AS* may violate safety if determinism is not satisfied. Although determinism prevents safety violations such as those shown in Figures 5 and 6, it does not prevent *all* violations of safety. For that, we require a stronger notion of determinism, which we call *egress determinism*.

5.2 Egress Determinism Violations in BGP: Route Reflection

Even if determinism is satisfied, an AS's iBGP topology can still cause a routing protocol to violate safety. In particular, we can construct an oscillation that involves the interaction between an AS's route reflector topology and its IGP topology. To better understand this interaction, we first define the notion of *egress determinism*. Egress determinism is a stronger condition than determinism, as shown in Figure 7; essentially, it states that, given a set of routes learned at *any* egress router in the AS, a router's preference between any pair of those routes should not depend on either the order in which those routes arrive or the presence or absence of other routes. Egress determinism implies determinism, but it also states that every router's selection function should satisfy determinism for all routes learned at *any* router in the AS, not just those learned locally at that router.

Definition 5.4 (Egress Determinism) *Let E_d be the set of routes for destination d learned at* any *router in the AS. Then, a routing protocol satisfies* egress determinism *for destination d if $\lambda_r(E_d)$ satisfies the following two properties:*

1. *$\lambda_r(E_d) = \lambda_r(E'_d)$, where E'_d is any subset of E_d that contains $\lambda_r(E_d)$, and*
2. *$\lambda_r(E_d)$ does not depend on the order in which the routes in E_d arrived at router r.*

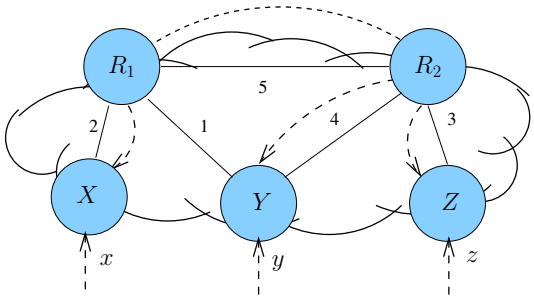

Figure 8: The interaction of IGP and iBGP can cause a violation of egress determinism. λ_{R_1} is equal to either x or y depending on whether $z \in E_d$.

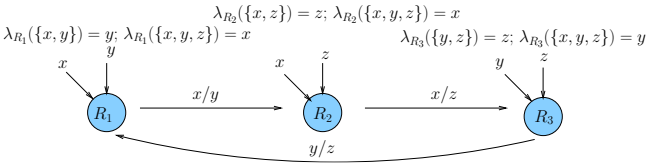

Figure 9: Violations of egress determinism can also cause the routing protocol to violate safety.

Note that egress determinism is a stronger condition than determinism because it states properties that λ_r must satisfy over the set of routes learned by all routers in the AS, not just the routes learned at r.

If every router in the AS always learned all routes in E_d, then violations of egress determinism would never cause oscillations: given a fixed set of routes E_d, every router would always see that set and select the same route. In an iBGP topology with route reflectors, however, most routers will see some subset of E_d, which means that violations of egress determinism may cause safety violations. Consider Figure 8: X is a route reflector client of R_1, and Y and Z are clients of route reflector R_2. Suppose that routers X, Y, and Z all learn routes for some destination d with equal local preference, AS path length, origin type, and MED attributes, causing routers within the pictured AS to resort to preferring eBGP routes over iBGP routes, and, that being equal, to prefer routes with the shortest IGP path cost. If $E_d = \{x, y, z\}$, then $\lambda_{R_1}(E_d) = x$: R_2 selects z due to its shorter IGP path cost to the next hop, and R_1, having learned x and z, selects route x. If, on the other hand, $E_d = \{x, y\}$, then $\lambda_{R_1}(E_d) = y$: R_2 selects y, and R_1, having learned both x and y, selects y due to the shorter IGP path cost. Thus, the first condition of egress determinism is violated.

Like determinism violations, egress determinism violations can cause the routing protocol to violate safety.

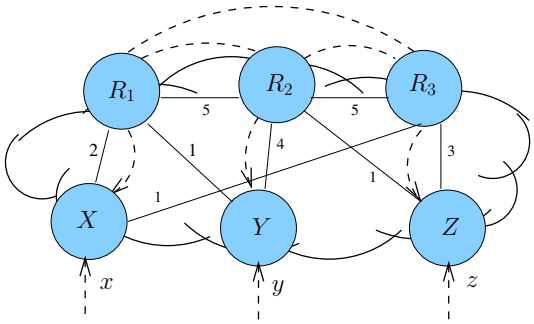

Figure 10: The instantiation of Figure 9 in BGP.

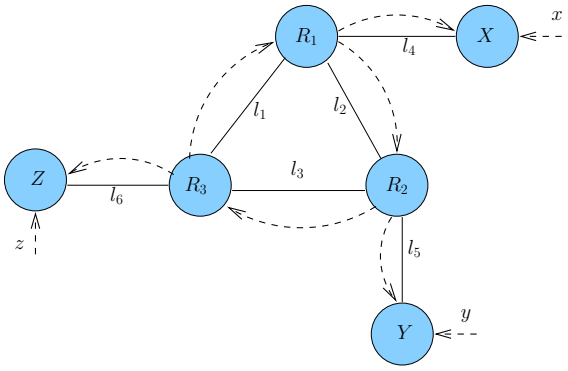

Figure 11: When iBGP violates safety but satisfies egress determinism, the only way a cyclic iBGP topology can violate safety is if the IGP allows negative edge weights.

Consider three routers whose selection functions violate egress determinism, as shown in Figure 9; R_1's selection function is identical to that in Figure 8. Each router prefers one route or the other depending on the presence or absence of a third route. In this case, there is no stable assignment of routes x, y, and z to routers R_1, R_2, and R_3. For example, if R_1 selects x, then R_2 selects z and R_3 selects y, prompting R_1 to select y, and so on. This very scenario can be realized in BGP today if three routers' route selection functions fail to satisfy egress determinism, as shown in Figure 10. Previous work has also observed that violations of this type could occur [16] but did not observe that these could be constructed in general by composing egress determinism violations.

Lemma 5.2 *If an AS's iBGP topology is* RR-IGP-Consistent*, and every router's selection function satisfies determinism, then every router's selection function also satisfies egress determinism.*

We now state the conditions for iBGP to satisfy safety using our results involving determinism and egress determinism. Specifically, we show that if MED is compared across all routes (*i.e.*, every router's selection function satisfies determinism) and if the iBGP topology is *RR-IGP-Consistent* (*i.e.*, egress determinism is satisfied), then iBGP satisfies safety.

Theorem 5.1 *If every router's selection function compares MED attribute across all routes and the iBGP topology is* RR-IGP-Consistent*, then iBGP satisfies safety.*

Our definitions have allowed us to derive sufficient conditions on safety that are significantly weaker (and therefore, the result is stronger) than in previous work [15]. In particular, *our results show that assuming that the relationships between route reflectors and their clients are acyclic is unnecessary* (although a cyclic topology may make an oscillation more likely if egress determinism is violated). It turns out that the only way for a cyclic iBGP topology to cause oscillations would be for either the iBGP topology to not be *RR-IGP-Consistent* or for some IGP edges to have negative edge weights.

To understand why cycles in the iBGP hierarchy do not cause problems if the topology is *RR-IGP-Consistent*, see Figure 11. In this example, if egress determinism is satisfied, then the only case where an oscillation might result is where R_1 prefers route y over route x, R_2 prefers route z over y, and R_3 prefers x over z. All other cases where oscillation might occur (*i.e.*, those caused by violations of egress determinism) require some shortest IGP path between a router and another egress to not traverse that router's route reflector. For safety to be violated in this example, routes x, y, and z must all have equal local preference, AS path length, origin type and MED (otherwise, all routers would select the most preferable route or routes). Presuming that all routes are equally good up to the step in route selection involving the IGP tiebreak, then the only way for such a situation to occur is if the following inequalities were satisfied:

$$\begin{aligned} l_1 + l_4 &< l_6 \\ l_2 + l_5 &< l_4 \\ l_3 + l_6 &< l_5 \end{aligned}$$

which implies that $l_1 + l_2 + l_3 < 0$, or that some IGP edge weights must be negative.

Theorem 5.1 is significant because the conditions on the iBGP topology that are required to guarantee safety are identical those for guaranteeing route validity, as stated in Theorem 3.2. Furthermore, because there are now known techniques for *generating* these configurations [27], our results are prescriptive, since this technique that was designed to generate iBGP topologies that guarantee route validity also happens to generate topologies that guarantee safety.

6 Conclusion

Proposals to fix the various ills in Internet routing have ranged from point solutions to a specific problem (*e.g.*, [1, 19]) to those advocating large-scale architectural change (*e.g.*, [3, 9, 26]). All of these efforts will benefit from a specification of what it means for Internet routing to be "correct". This paper motivated and defined three aspects of correctness that are fundamental to any routing

protocol: route validity, path visibility, and safety. These properties provide a foundation for tools, protocols, and architectures to improve network routing and, more generally, can help both theoreticians and practitioners study and evaluate the properties of any routing protocol. Although we have explored our correctness specification in the context of today's Internet routing system, we hope that it will prove useful for evaluating the behavior of any large-scale policy-based network routing system.

References

[1] A. Basu et al. Route Oscillations in IBGP with Route Reflection. In *Proc. ACM SIGCOMM*, pages 235–247, Pittsburgh, PA, Aug. 2002.

[2] O. Bonaventure, S. Uhlig, and B. Quoitin. *The Case for More Versatile BGP Route Reflectors*. Internet Engineering Task Force, July 2004. `http://totem.info.ucl.ac.be/publications/draft-bonaventure-bgp-route-reflectors-00.html` Work in progress, expired November 2004.

[3] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and K. van der Merwe. Design and Implementation of a Routing Control Platform. In *Proc. 2nd Symposium on Networked Systems Design and Implementation (NSDI)*, pages 15–28, Boston, MA, May 2005.

[4] B. Davie and Y. Rekhter. *MPLS: Technology and Applications*. Academic Press, San Diego, CA, 2000.

[5] R. Dube. A Comparison of Scaling Techniques for BGP. *ACM Computer Communications Review*, 29(3):44–46, July 1999.

[6] D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina. *Generic Routing Encapsulation (GRE)*. Internet Engineering Task Force, Mar. 2000. RFC 2784.

[7] N. Feamster. *Proactive Techniques for Correct and Predictable Internet Routing*. PhD thesis, Massachusetts Institute of Technology, Feb. 2006.

[8] N. Feamster and H. Balakrishnan. Detecting BGP Configuration Faults with Static Analysis. In *Proc. 2nd Symposium on Networked Systems Design and Implementation (NSDI)*, pages 43–56, Boston, MA, May 2005.

[9] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and K. van der Merwe. The Case for Separating Routing from Routers. In *ACM SIGCOMM Workshop on Future Directions in Network Architecture*, pages 5–12, Portland, OR, Sept. 2004.

[10] N. Feamster, J. Borkenhagen, and J. Rexford. Guidelines for Interdomain Traffic Engineering. *ACM Computer Communications Review*, 33(5):19–30, Oct. 2003.

[11] N. Feamster, R. Johari, and H. Balakrishnan. The Implications of Autonomy for Stable Policy Routing. In *Proc. ACM SIGCOMM*, pages 25–36, Philadelphia, PA, Aug. 2005.

[12] N. Feamster and J. Rexford. Modeling BGP route selection within an AS. *IEEE/ACM Transactions on Networking*. To appear. Earlier version appears in *ACM SIGMETRICS 2004*.

[13] V. Fuller, T. Li, J. Yu, and K. Varadhan. *Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy*. Internet Engineering Task Force, Sept. 1993. RFC 1519.

[14] T. Griffin, F. B. Shepherd, , and G. Wilfong. The Stable Paths Problem and Interdomain Routing. *IEEE/ACM Transactions on Networking*, 10(1):232–243, 2002.

[15] T. Griffin and G. Wilfong. A Safe Path Vector Protocol. In *Proc. IEEE INFOCOM*, pages 490–499, Tel Aviv, Israel, Mar. 2000.

[16] T. Griffin and G. Wilfong. On the Correctness of IBGP Configuration. In *Proc. ACM SIGCOMM*, pages 17–29, Pittsburgh, PA, Aug. 2002.

[17] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet Routing Convergence. *IEEE/ACM Transactions on Networking*, 9(3):293–306, June 2001.

[18] R. Mahajan, D. Wetherall, and T. Anderson. Understanding BGP Misconfiguration. In *Proc. ACM SIGCOMM*, pages 3–17, Pittsburgh, PA, Aug. 2002.

[19] Z. M. Mao, R. Govindan, G. Varghese, and R. Katz. Route Flap Damping Exacerbates Internet Routing Convergence. In *Proc. ACM SIGCOMM*, pages 221–233, Pittsburgh, PA, Aug. 2002.

[20] D. McPherson, V. Gill, D. Walton, and A. Retana. *Border Gateway Protocol (BGP) Persistent Route Oscillation Condition*. Internet Engineering Task Force, Aug. 2002. RFC 3345.

[21] Multiprotocol Label Switching (MPLS). `http://www.ietf.org/html.charters/mpls-charter.html`.

[22] S. Murphy, A. Barbir, and Y. Yang. *Generic Threats to Routing Protocols*. Internet Engineering Task Force, Oct. 2004. `http://www.ietf.org/internet-drafts/draft-ietf-rpsec-routing-threats-07.txt`, expired April 2005.

[23] Y. Rekhter and T. Li. *An Architecture for IP Address Allocation with CIDR*. Internet Engineering Task Force, Sept. 1993. RFC 1518.

[24] Y. Rekhter and T. Li. *A Border Gateway Protocol 4 (BGP-4)*. Internet Engineering Task Force, Mar. 1995. RFC 1771.

[25] Y. Rekhter, T. Li, and S. Hares. *A Border Gateway Protocol 4 (BGP-4)*. Internet Engineering Task Force, Oct. 2004. `http://www.ietf.org/internet-drafts/draft-ietf-idr-bgp4-26.txt` Work in progress, expired April 2005.

[26] L. Subramanian, M. Caesar, C. T. Ee, M. Handley, M. Mao, S. Shenker, and I. Stoica. HLP: A Next-generation Interdomain Routing Protocol. In *Proc. ACM SIGCOMM*, Philadelphia, PA, Aug. 2005.

[27] M. Vutukuru, P. Valiant, S. Kopparty, and H. Balakrishnan. How to Construct a Correct and Scalable iBGP Configuration. Technical Report MIT-LCS-TR-996, MIT Computer Science and Artificial Intelligence Laboratory, Aug. 2005. `http://www.lcs.mit.edu/publications/specpub.php?id=1787`.