Implications of Autonomy for the Expressiveness of Policy Routing

Nick Feamster
MIT
feamster@csail.mit.edu

Ramesh Johari
Stanford University
ramesh.johari@stanford.edu

Hari Balakrishnan
MIT
hari@csail.mit.edu

ABSTRACT

Thousands of competing autonomous systems must cooperate with each other to provide global Internet connectivity. Each autonomous system (AS) encodes various economic, business, and performance decisions in its routing policy. The current interdomain routing system enables each AS to express policy using *rankings* that determine how each router in the AS chooses among different routes to a destination, and *filters* that determine which routes are hidden from each neighboring AS. Because the Internet is composed of many independent, competing networks, the interdomain routing system should provide *autonomy*, allowing network operators to set their rankings independently, and to have no constraints on allowed filters. This paper studies routing protocol stability under these conditions. We first demonstrate that certain rankings that are commonly used in practice may not ensure routing stability. We then prove that, when providers can set rankings and filters autonomously, guaranteeing that the routing system will converge to a stable path assignment essentially requires ASes to rank routes based on AS-path lengths. We discuss the implications of these results for the future of interdomain routing.

Categories and Subject Descriptors

C.2.6 [**Computer Communication Networks**]: Internetworking; C.2.2 [**Computer Communication Networks**]: Network Protocols—*Routing Protocols*

General Terms

Design, Reliability, Performance, Theory

Keywords

Routing, Internet, policy, autonomy, safety, stability, BGP, protocol

1. Introduction

The Internet's routing infrastructure is made up of thousands of independently operated networks that cooperate to exchange global reachability information using an interdomain routing protocol, the Border Gateway Protocol, Version 4 (BGP) [16]. This cooperation occurs in a landscape where these independent networks, or autonomous systems, compete to provide Internet service. BGP facilitates this "competitive cooperation" by enabling network operators to express routing policies that are consistent with desired economic, business, and performance goals.

Ranking and *filtering* are the two main mechanisms that operators use to implement their policies. Ranking determines which of many possible routes to a destination should be used, thus providing an autonomous system (AS) the freedom to specify preferences over multiple candidate paths to a destination (*e.g.*, specifying a primary and a backup path). Filtering allows an AS to selectively advertise routes to some autonomous systems and hide routes from others, thereby controlling which neighboring autonomous systems send traffic over its infrastructure.

There are two important characteristics of policy routing: *autonomy* and *expressiveness*. Autonomy is the ability of each AS to set its rankings and filters independent of the others. Expressiveness refers to the flexibility of the routing protocol in allowing operators to specify rankings and filters. Ranking expressiveness determines what classes of rankings over routes are permitted by the protocol, while filtering expressiveness determines the range of route filters that are allowed.

The combination of expressiveness and autonomy has, in large part, been the reason for the success of BGP over the past decade. We contend that both autonomy and filtering expressiveness will be *requirements* for policy routing for the foreseeable future. Previous studies of routing stability assume that ASes are willing to compromise some degree of autonomy, filtering expressiveness, or both (see Section 2). However, autonomy preserves each AS's ability to set its policies without coordinating with any other AS. Filtering expressiveness gives an AS flexibility in how it establishes contracts with another AS, a task that should be unconstrained.

Ideally, an interdomain routing system should preserve autonomy, filtering expressiveness, and ranking expressiveness. However, the ability to specify highly expressive rankings comes at considerable cost to system robustness: as has been observed by Varadhan *et al.* and Griffin *et al.*, among others, if there are no constraints on the rankings that an AS can specify, BGP may oscillate forever [12, 18].

Example 1 Consider Figure 1 [12, 18]. ASes 1, 2, and 3 each prefer the indirect path through their neighboring AS in the clockwise direction over the direct path to the destination, 0. All other paths are filtered. This configuration has no stable path assignment (*i.e.*, a path assignment from which no node would deviate). For example, consider the path assignment $(10, 210, 30)$; in this case, AS 1 has a better path available, 130, so it switches paths. This switch breaks the path 210, causing AS 2 to switch to its second choice,

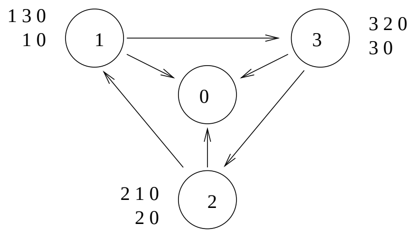


Figure 1: Instability can arise when each AS independently specifies rankings [12, 18]. Each circle represents an AS. AS 0 is the destination. The listing of paths beside each node denotes a ranking over paths.

path 20. The resulting path assignment, $(130, 20, 30)$, is a permutation of the original path assignment: this time, AS 3 has the path 320 available, so it switches. This oscillation continues forever. ■

As the previous example suggests, full autonomy and expressiveness can have undesirable consequences. Routing protocol update messages should reflect actual reachability changes in the network topology or policy. Unfortunately, in BGP, conflicting policies can cause oscillations that produce endless streams of routing updates that are unrelated to changes in topology or policy. This instability creates numerous performance problems, may cause network partitions, and complicates diagnosis and debugging of problems in the routing system. Furthermore, a network operator has no way to guarantee that any given configuration of rankings and filters will not adversely interact with the policies of other ASes. In light of these issues, developing rigorous conditions on policy expressiveness that guarantee routing stability, while preserving autonomy, is crucial.

This paper explores the following question: provided that each AS retains complete autonomy and complete filtering expressiveness, how expressive can rankings be while guaranteeing stable routing? This question is important because ranking autonomy and filtering expressiveness reflect the realities of how ASes specify policies today, and little is known (beyond the results surveyed in Section 2) about the tradeoffs between autonomy and expressiveness as far as routing stability is concerned, particularly under filtering. In particular, our work is the first to develop necessary conditions for stability under realistic assumptions about autonomy and expressiveness and the first to derive necessary conditions for stability in policy routing.

This paper makes three main contributions. First, in Section 4.1, we show that rankings based solely on the immediate next-hop AS en route to the destination may never reach a stable path assignment from an arbitrary initial state; *i.e.*, next-hop rankings, which are common in practice, are *not safe*. Moreover, under unrestricted filtering, a routing system with next-hop rankings may have no stable path assignment. In addition to their operational implications, these results are also somewhat surprising, because next-hop rankings with no route filtering always have one stable path assignment. We also observe that although rankings based on a globally consistent weighting of paths are safe under filtering, even minor generalizations of the weighting function compromise safety (Section 4.2).

Second, we define a *dispute ring*, a special case of the "dispute wheel" (a group of nodes whose rankings have a particular form) of Griffin *et al.* [12], and show that any routing protocol that has a dispute ring is not safe under filtering (Section 5). Using the dispute wheel concept, Griffin *et al.* showed a sufficient condition for safety, proving that if a routing system is unsafe then it must have a dispute wheel. In contrast, to our knowledge, our result is the first known necessary condition for safety under filtering.

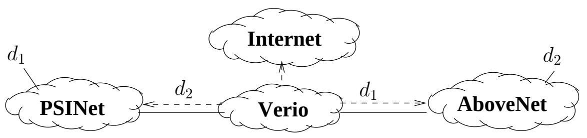


Figure 2: Constraints on filtering and topology are not enforceable.

Third, we show that, providing for complete autonomy and filtering expressiveness, the set of allowable rankings that guarantee safety is effectively ranking based on (weighted) shortest paths. In Section 6, we prove that any routing system that permits paths of length $n + 2$ to be ranked over paths of length n can have a dispute ring, and is thus unsafe under filtering. We also prove that any routing system that permits paths of length $n + 1$ to be ranked over paths of length n can have a dispute wheel. In summary, our results indicate that stable policy routing with provider autonomy and expressive filtering requires tight constraints on rankings.

Our findings may be interpreted in several ways. The optimist will note that checking a set of rankings to ensure safety is trivial, because all it requires is that BGP routers modify the decision process to consult a route's "local preference" attribute only after considering its AS path length. The pessimist, however, may conclude that guaranteeing safe routing while preserving autonomy may yield constraints on expressiveness that are too constraining. In either case, the results proved in this paper about the fundamental tradeoff between the expressiveness and autonomy may help guide the design of stable interdomain routing protocols in the future.

2. Background and Related Work

A seminal paper by Varadhan *et al.* observed that policy-based interdomain routing protocols could oscillate and defined the concept of safety [18]. Varadhan *et al.* also conjectured that routing systems that allow rankings other than those based on next-hop rankings or shortest path routing may be unsafe [18].

Griffin *et al.* asked how expressive an autonomous, robust routing system can be [11]; our paper addresses this question. Varadhan *et al.* showed that a routing system with an acyclic topology will have at least one stable path assignment if participants can only express next-hop preferences [18]. In this paper, we use a construction due to Feigenbaum *et al.* [6] to show that systems with next-hop rankings always have at least one stable routing. However, we also show that when BGP's protocol dynamics are taken into account, restricting each AS to only next-hop rankings does *not* guarantee that the routing system will be safe.

Gao and Rexford derived sufficient constraints on rankings, filtering, and network topology to guarantee routing stability; they also observe that these constraints reflect today's common practice [7, 8]. They showed that if every AS considers each of its neighbors as either a customer, a provider, or a peer, and obeys certain local constraints on rankings and filtering, and if the routing system satisfies certain topology constraints, then BGP is stable. However, their model does not incorporate ranking independence, as their proposed topological constraints are global. Furthermore, their model restricts filtering; the example below illustrates why these restrictions may sometimes be too strict.

Example 2 Figure 2 shows a situation that occurred in 2001 [2]. When PSINet terminated its peering with AboveNet, AboveNet lost connectivity to PSINet's customers, d_1. To restore connectivity, AboveNet bought "transit" service from Verio (already a peer of PSINet), but only for routes to PSINet and its customers.

Verio does not filter d_1 (or any of PSINet's prefixes) from AboveNet, which is only possible if Verio treats AboveNet as a customer. The constraints imposed by Gao and Rexford state that an AS *must* prefer customer routes over peering routes.[1] This constraint requires Verio to rank AboveNet's route to d_2 over any other available routes to d_2 in order to guarantee stability, which restricts Verio's flexibility in how it can select routes. Establishing a new business relationship (and, hence, altering its filtering policies) *requires* Verio to change its rankings as well. ■

Researchers have previously studied *global* conditions to guarantee the safety of routing systems; global conditions presume that the routing system does not preserve autonomy. Griffin *et al.* showed that, if the rankings of the ASes in a routing systems do not form a *dispute wheel* (a concept that describes global relationship between the rankings of a set of ASes), then the routing system is safe [12]. Griffin *et al.* also examined *robustness*, the property that safety is guaranteed even if arbitrary nodes or edges are removed from the graph. We view robustness as a special case of filtering: removing an edge can be achieved if the ASes incident to that edge filter all routes through that edge; removing a node entails having all ASes filter all routes through that node.

Griffin *et al.* also showed how to modify a BGP-like path vector protocol to detect the existence of a dispute wheel but left unspecified how the ASes should resolve the dispute wheel [13]. Machiraju and Katz defined a global invariant for determining safety when at most one AS deviates from the conditions of Gao and Rexford [15]. Govindan *et al.* proposed a routing architecture where ASes coordinate their policies [9, 10] using a standardized policy specification language [1]. Jaggard and Ramachandran presented global conditions to guarantee safety of routing systems that allow ASes to express only next-hop preferences over routes, and designed centralized and distributed algorithms to check these global conditions [14]. Sobrinho defined concepts that describe global relationships between preferences and incorporated several previous results (including those of both Griffin *et al.* [12] and Gao and Rexford [8]) into a single algebraic framework [17]. In contrast to previous work, this paper recognizes autonomy and ranking expressiveness as requirements and studies the conditions under which a policy-based interdomain routing protocol can be stable.

3. Routing Model and Definitions

We now define our routing model. After introducing some basic terminology, we formally define two notions of good behavior for routing protocols: *stability* and *safety*. Finally, we extend each of these two definitions to handle filtering expressiveness.

3.1 Preliminaries

We consider a model consisting of N ASes (nodes)[2], labeled $1, \ldots, N$. Each of these nodes wishes to establish a path (defined below) to a single destination, labeled 0.

Definition 1 (Path) *A* path *from i to j is a sequence of nodes $P = ii_1i_2 \ldots i_mj$ with no repetition,;* i.e.*, such that $i_u \neq i_v$ if $u \neq v$, and $i_u \neq i, j$ for all u.*

[1]Gao and Rexford present a weaker constraint that allows an AS to rank routes learned from customers and peers over those from providers, but does *not* require customer routes to be strictly preferred over routes from peers. This relaxed condition requires that there are no instances where an AS's customer is also a peer of another one of the AS's peers. Of course, Example 2 could also violate this constraint on the topology: PSINet is Verio's customer for d_1, but it would be reasonable for PSINet to peer with another of Verio's peers, since all are "tier-1" ISPs.

[2]In this paper, we use the terms "AS" and "node" interchangeably.

We denote the number of hops in a path P as $length(P)$; note that a path with n nodes has $n-1$ hops. In addition, given an AS k, we will write $k \in P$ if node k appears in P. For clarity, given a path P from i to j, we will often denote P by iPj; furthermore, if P is a path from i to j, and Q is a path from j to k, then we will denote the concatenation of P and Q by $iPjQk$.

We denote the set of *all* paths from i to 0 (*i.e.*, all paths on the complete graph) using the nodes $1, \ldots, N$ by $\mathcal{P}_i^N$. Given the set of nodes $\{1, \ldots, N\}$, each AS i will choose a *ranking* $\prec_i$ over the set of all paths $\mathcal{P}_i^N$, defined as follows.

Definition 2 (Ranking) *A* ranking $\prec_i$ *for node i is a total ordering over the set of all paths $\mathcal{P}_i^N$; thus, given any two paths $P, Q \in \mathcal{P}_i^N$, either $P \prec_i Q$ (i prefers Q to P) or $P \succ_i Q$ (i prefers P to Q).*

An AS may always choose the *empty path*, ε, which is equivalent to total disconnection from the destination node 0. Thus, we have $\varepsilon \in \mathcal{P}_i^N$ for all i and N. Furthermore, we assume that every AS strictly prefers connectivity to disconnectivity, so that $P \succ_i \varepsilon$ for all $P \in \mathcal{P}_i^N$.

Note that all paths may not be available to node i, due to both topological constraints and filtering by other nodes. We will use $\mathcal{F}_i \subseteq \mathcal{P}_i^N$ to denote the set of paths actually available for use by node i. The empty path is always available; *i.e.*, $\varepsilon \in \mathcal{F}_i$.

A *routing system* is specified by the rankings of the individual nodes, together with the paths available to the individual nodes. Observe that we have decoupled the "routing policy" of each AS i into two components: the rankings $\prec_i$ of AS i over route advertisements received; and a determination of which paths are filtered from other ASes. The filtering decisions of all nodes, together with physical constraints on the network, yield the sets $\mathcal{F}_1, \ldots, \mathcal{F}_N$. We thus have the following formal definition of a routing system.

Definition 3 (Routing system) *A* routing system *is a tuple $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$, where node i has ranking $\prec_i$ over the set $\mathcal{P}_i^N$, and $\mathcal{F}_i$ is the set of paths available to node i.*

A routing system specifies the input to any interdomain routing protocol we might consider. Given this input, the protocol should converge to a "routing tree": that is, an assignment of a path to each AS, such that the routes taken together form a spanning tree rooted at 0. To formalize this notion, we must define path assignments and consistent paths.

Definition 4 (Path assignment) *A* path assignment *for the routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ is a vector of paths $\boldsymbol{P} = (P_1, \ldots, P_N)$ such that, for all i, $P_i \in \mathcal{F}_i$.*

Thus, a path assignment is an assignment of a feasible path to each node i, where feasibility is determined by the set of paths $\mathcal{F}_i$. Even though each node has a path assigned, these paths may not be *consistent*: node i may be assigned a path $P_i = ij\hat{P}_j0$, where j is the first node traversed on P_i, and where $\hat{P}_j$ is a path from j to 0. However, the path $\hat{P}_j$ may not be the same as the path P_j assigned to j in the path assignment $\boldsymbol{P}$; in fact, $\hat{P}_j$ may not even be in the set of feasible paths $\mathcal{F}_j$. For example, a node or link along the path $\hat{P}_j$ may experience a failure, causing the routing protocol to withdraw the path; if j has heard such a withdrawal but i has not, then it is possible that $P_i = ij\hat{P}_j0$ until node i learns that $\hat{P}_j$ no longer exists. To formally capture such situations, we define consistent paths and consistent path assignments.

Definition 5 (Consistent path) *Given a path assignment $\boldsymbol{P}$, a path $\hat{P}_i$ for node i is* consistent *with $\boldsymbol{P}$ if one of the following holds:*

1. *$\hat{P}_i = \varepsilon$; or*
2. *$\hat{P}_i = i0$; or*
3. *$\hat{P}_i = ijP_j0$, for some $j \neq i$.*

Definition 6 (Consistent path assignment) *A* consistent path assignment *for the routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ is a path assignment vector $\boldsymbol{P} = (P_1, \ldots, P_N)$ such that for all i, P_i is consistent with $\boldsymbol{P}$.*

A routing protocol where packets are forwarded solely on destination should ultimately assign paths that are consistent with each other.

3.2 Stability and Safety

Informally, a path assignment is *stable* if it is consistent, and no node has a more preferred consistent path available.

Definition 7 (Stable path assignment) *Given a routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$, and a consistent path assignment $\boldsymbol{P}$, we say that $\boldsymbol{P}$ is* stable *if for all nodes i, and all paths $\hat{P}_i \in \mathcal{F}_i$ that are consistent with $\boldsymbol{P}$, $\hat{P}_i \prec_i P_i$.*

Definition 8 (Stable routing system) *The routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ is* stable *if there exists at least one stable path assignment $\boldsymbol{P}$.*

The stability of a routing system does not indicate whether a routing protocol will converge *regardless* of the initial path assignment. The *safety* property, which states that a protocol eventually converges, regardless of the initial path assignment and ordering of the routing messages, captures this notion.

In defining safety, we will consider a simplified abstraction of BGP. We model the process by which nodes receive route advertisements from other nodes and subsequently update their own route decisions. In this paper, we will consider a protocol dynamic where at each time step only a single AS is *activated*; when activated, an AS immediately processes all pending incoming route advertisements, and then makes a route decision. Formally, this model will translate into a path assignment sequence where exactly one node (the "activated" node) changes its route at any given time step.

A routing system is safe if no oscillation occurs regardless of the order in which nodes are activated.

Definition 9 (Fair activation sequence) *The sequence $i_1, i_2, \ldots$ is a* fair activation sequence *if each node $i = 1, \ldots, N$ appears infinitely often in the sequence.*

This definition of fair activation sequence is similar to that presented by Gao and Rexford [8], except that in our definition we only activate one node at a time. This distinction is not major: we can interpret the Gao and Rexford dynamics as a model where outstanding routing messages may be in flight when a particular node is activated.

We now define our simplified model of the routing protocol dynamics: that is, starting from an initial path assignment $\boldsymbol{P}_0$, and given a fair activation sequence of nodes $i_1, i_2, \ldots$, what is the resulting observed sequence of path assignments $\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots$? To formalize the dynamics of our model, we consider an abstraction of the BGP decision process described in Figure 3. At each time t, a node i_t is activated, and chooses its most preferred available path consistent with the path assignment $\boldsymbol{P}_{t-1}$. All other nodes' paths remain unchanged. It is clear that this decision process yields a sequence of path assignments $\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots$.

Routing protocol dynamics

At time $t-1$, the current path assignment is $\boldsymbol{P}_{t-1}$; *i.e.*, each node i has currently selected path $P_{i,t-1}$ to the destination 0. At time t:

1. A given node i_t is activated.
2. Node i_t updates its path to be the *most preferred path* (according to $\prec_{i_t}$) consistent with $\boldsymbol{P}_{t-1}$. That is,
   (a) $P_{i_t,t} \in \mathcal{F}_{i_t}$ is consistent with $\boldsymbol{P}_{t-1}$, and
   (b) $P_{i_t,t} \succ_{i_t} \hat{P}_{i_t} \; \forall \; \hat{P}_{i_t} \in \mathcal{F}_{i_t}$ consistent with $\boldsymbol{P}_{t-1}$.
3. All other nodes leave their paths unchanged.

Figure 3: The routing protocol dynamics, given an activation sequence $i_1, i_2, \ldots$. The process starts from an initial path assignment $\boldsymbol{P}_0$.

After any given activation step t, the overall path assignment $\boldsymbol{P}_t$ may not be consistent. Inconsistencies reflect the fact that a node only updates its path assignment in response to the receipt of a route advertisement. If, at time t_0, a node i is using a path that traverses some other node j that has since changed paths, then node i would obliviously continue to use (and advertise) that *inconsistent* path until it receives a routing update that reflects that the path through j has disappeared or changed. When activated, say, at time $t > t_0$, node i would discover that the path it was using was inconsistent with $\boldsymbol{P}_t$ and would then select its highest-ranked path that was consistent with $\boldsymbol{P}_t$. The activation of a node at some time t corresponds to that node receiving all available routing information in the system up to that time.

With the definition of our protocol dynamics in hand, we can define protocol *safety*. Given a routing system and an activation sequence, we say that the system has converged if, after some finite time, the path assignment remains invariant for all future time. A protocol is *safe* if it converges to a stable path assignment, regardless of the initial path assignment and fair activation sequence.

Definition 10 (Safety) *A routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ is* safe *if for any initial path assignment $\boldsymbol{P}_0$ and fair activation sequence $i_1, i_2, \ldots$, there exists a finite T such that $\boldsymbol{P}_t = \boldsymbol{P}_T$ for all $t \geq T$.*

Because the activation sequences are fair in the preceding definition, if a routing system converges to $\boldsymbol{P}_t$, then the resulting path assignment to which the system converges must be both consistent and stable. If not, at least one node would change its path assignment eventually.

3.3 Filtering

In this paper, we are interested in the stability and safety of systems that result when nodes are allowed to filter routes from other nodes. We thus require conditions stronger than stability and safety, known as *stability under filtering* and *safety under filtering*. Informally, a routing system is stable (respectively, safe) under filtering if, under any choices of filters made by the ASes, the resulting routing system is always stable (respectively, safe).

Definition 11 (Stable under filtering) *The routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ is* stable under filtering *if, for all* *choices of available paths $\hat{\mathcal{F}}_i \subseteq \mathcal{F}_i$ for $i = 1, \ldots, N$, the routing system $(N, \prec_1, \ldots, \prec_N, \hat{\mathcal{F}}_1, \ldots, \hat{\mathcal{F}}_N)$ is stable.*

Definition 12 (Safe under filtering) *The routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ is* safe under filtering *if, for all choices of available paths $\hat{\mathcal{F}}_i \subseteq \mathcal{F}_i$ for $i = 1, \ldots, N$, the routing system $(N, \prec_1, \ldots, \prec_N, \hat{\mathcal{F}}_1, \ldots, \hat{\mathcal{F}}_N)$ is safe.*

We interpret these definitions as follows. The set of available paths $\mathcal{F}_i$ gives the set of paths that are physically possible for node i to use, given the current network topology. Once all nodes have chosen their route filters, $\hat{\mathcal{F}}_i$ gives the set of paths that can ever be used by node i in route advertisements. Because we allow arbitrary choice of filters, the resulting routing system should be stable and safe regardless of the choices of $\hat{\mathcal{F}}_1, \ldots, \hat{\mathcal{F}}_N$ that are made.

4. Ranking Classes and Safety

In this section, we study two natural ranking classes under which ASes retain autonomy in setting rankings over paths. First, in Section 4.1, we study the rankings where each AS is allowed to rank paths solely based on the immediate next-hop AS, called "next-hop rankings". We show that (1) there are routing systems where each node has only a next-hop ranking that are not safe; and (2) even though all routing systems where nodes have next-hop rankings are stable, there exist some routing systems of this form that are not stable under filtering.

In Section 4.2, we study the properties of routing systems where each node is allowed to choose a weight for all its outgoing links, and rankings are derived from a "total" weight associated to each path. The total weight of a path is defined as the weight of the first link on that path, plus a discounted sum of the weights of all remaining links on that path. We show that if the discount factor is anything other than 1 (which corresponds to shortest path routing), then there exist weight configurations that yield a routing system that is not safe.

4.1 Next-Hop Rankings

One natural set of rankings for a routing system is one where each AS can express rankings over paths solely based on the next-hop AS in the path. Such a class of rankings makes sense because an AS establishes bilateral contracts with its immediate neighbors and, as such, will most often wish to configure its rankings based on the immediate next-hop AS en route to the destination. For example, an AS will typically prefer sending traffic via routes through its neighboring customer ASes over other ASes, since those customer ASes are paying based on traffic volume. We formally define next-hop rankings as follows:

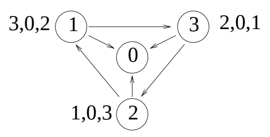
Definition 13 (Next-hop ranking) *Given N, $\prec_i$ is a* next-hop ranking *if, for all nodes j, k with i, j, k distinct, we have:*

$$ijP_j0 \prec_i ikP_k0 \;\Rightarrow ijP'_j0 \prec_i ikP'_k0, \tag{1}$$

for all $P_j, P'_j \in \mathcal{P}_j^N$, and $P_k, P'_k \in \mathcal{P}_k^N$. (Here we interpret $\mathcal{P}_0^N = \{\varepsilon\}$.)

Thus, $\prec_i$ ranks paths based only on the first hop of each path.

Such a restriction on policy would still be sufficiently rich to achieve most traffic engineering goals, since most policies are based on the immediate next-hop AS [4]. Additionally, this class of rankings is expressive enough for most current policy goals, because most current routing policies are dictated according to the AS's business relationship with its immediate neighbor. In this section,



(a) Routing system

Activate	1	2	3
—	(1 0)	(2 0)	(3 2 0)
2	(1 0)	**(2 1 0)**	(3 2 0)
1	**(1 3 2 0)**	(2 1 0)	(3 2 0)
3	(1 3 2 0)	(2 1 0)	**(3 2 1 0)**
2	(1 3 2 0)	**(2 0)**	(3 2 1 0)
1	**(1 0)**	(2 0)	(3 2 1 0)
3	(1 0)	(2 0)	(3 2 0)

(b) Activation sequence

Figure 4: Next-hop rankings are not safe in this routing system. AS 1 prefers all paths through AS 3 over the direct path to the destination 0 (with ties broken deterministically) and prefers the direct path over all paths through AS 2. Similarly, AS 3 prefers all paths via AS 2, and so forth.

we show that while systems with next-hop rankings are generally stable, there exist examples that are unsafe, as well as systems that are unstable under filtering.

In the following proposition, we routing systems with next-hop rankings, provided that no filtering is employed. The proof is straightforward, using a construction due to Feigenbaum *et al.* [6].

Proposition 1 *Suppose $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ is a routing system such that $\prec_i$ is a next-hop ranking for each i, and $\mathcal{F}_i = \mathcal{P}_i$ for all i. Then there exists a stable path assignment $\boldsymbol{P}$ for this routing system.*

We now show that there may exist $\hat{\mathcal{F}}_1 \ldots \hat{\mathcal{F}}_N$, where $\hat{\mathcal{F}}_i \subseteq \mathcal{F}_i$ for all i, such that even though the system $(N, \prec_1 \ldots \prec_N, \mathcal{F}_1 \ldots \mathcal{F}_N)$ is stable, the filtered system $(N, \prec_1 \ldots \prec_N, \hat{\mathcal{F}}_1 \ldots \hat{\mathcal{F}}_N)$ is unstable. That is, there exist routing systems with next-hop rankings for which a stable path assignment exists, but introducing filtering can yield a system where *no* stable path assignment exists.

Observation 1 *A routing system where each node has only a next-hop ranking may not be safe.*

Example 3 Consider Figure 4. In this example, each AS ranks every one of its neighboring ASes. For example, AS 1 prefers all paths that traverse AS 3 as the immediate next hop over all paths that traverse AS 0 as the immediate next hop, regardless of the number of ASes each path traverses; similarly, AS 1 prefers paths that traverse AS 0 as the immediate next hop over paths that traverse AS 2. Each AS readvertises its best path to the destination to all of its neighbors (*i.e.,* the system has no filtering). Now consider the activation sequence in Figure 4(b); if infinitely repeated, this activation sequence would be fair, and the routing system would oscillate forever. Thus, the routing system is not safe.

Note that this system is not *safe*, but it is *stable*: for example, the path assignment $(10, 210, 3210)$ is stable. Nodes 2 and 3 are using paths through their most preferred nodes. Node 1's most preferred node, node 3, is using a path that already goes through node 1, so node 1 is also using its most preferred consistent path. As every node is using its most preferred consistent path, no node will change paths when activated, so the path assignment is stable. ■

A routing system where each node has a next-hop ranking may not be safe, but Feigenbaum *et al.* showed that there is always guaranteed to be at least one stable path assignment for such routing systems [6]. However, allowing nodes to filter paths from each other can create routing systems for which there is *no* stable path assignment.

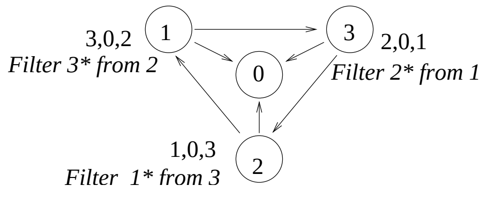


Figure 5: This routing system is stable without filtering but unstable under filtering. The figure shows a routing system with next-hop rankings and filtering that is equivalent to the unstable routing system with the rankings over paths shown in Figure 1.

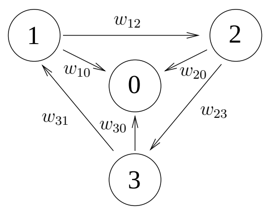


Figure 6: Routing system with edge weight-based rankings.

Observation 2 *There exist routing systems with next-hop rankings for which a stable path assignment exists, but introducing filtering can yield a system where* no *stable path assignment exists.*

Example 4 Consider Figure 5. As before, each AS ranks every one of its neighboring ASes. Additionally, each AS may also declare arbitrary filtering policies. In this example, each AS (other than the destination) does *not* readvertise any indirect path to the destination. For example, AS 1 does not advertise the path 130 to AS 2, and thus the path 2130 is not available to AS 2. Formally, we define $\mathcal{F}_1 = \{130, 10\}$, $\mathcal{F}_2 = \{210, 20\}$, and $\mathcal{F}_3 = \{320, 30\}$.

The resulting routing system is equivalent to the system in Figure 1, once the filtered paths are removed from each node's ranking. Thus, the filtered routing system is unstable by the same reasoning as in Example 1: for any path assignment in this routing system, at least one AS will have a higher ranked consistent path (and, hence, has an incentive to deviate from the path assignment). ■

Using a construction similar to that in Example 2, it is possible to show how this example could arise in practice. The example demonstrates the complex interaction between filtering and rankings—a class of rankings that guarantees stability without filtering can be unstable under certain filtering conditions.

4.2 Edge Weight-Based Rankings

There exists at least one routing system that preserves autonomy and yet ensures safety under filtering: if each provider is allowed to choose edge weights for its outgoing links, and each provider ranks paths based on the sum of edge weights, the resulting "shortest paths" routing system is guaranteed to be safe [12]. Since this result holds for any $\mathcal{F}_1, \ldots, \mathcal{F}_N$, any routing system built in this way is guaranteed to be safe under filtering. In this section, we will formulate a generalized model of such *edge weight-based rankings*, with both next-hop rankings and shortest path routing as special cases. Such rankings do not allow providers to directly specify their ranking; rather, the rankings of each provider are *derived* from the strategic choices made by all providers, namely, the choices of outgoing link weights that each provider sets. This notion of "derived" rankings is a potentially useful method for ensuring autonomy in interdomain routing protocols.

Definition 14 (Edge weight-based rankings) $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ *is a routing system with* edge weight-based rankings *if there exists an assignment of edge weights w_{ij} to each ordered pair of ASes i, j, together with a parameter $\alpha \in [0, 1]$, such that for each AS i and paths $P_i, \hat{P}_i \in \mathcal{P}_i^N$ with $P_i = ii_1 \ldots i_n 0$ and $\hat{P}_i = ij_1 \ldots j_m 0$, there holds:*

$$P_i \prec_i \hat{P}_i \text{ if and only if } w_{ii_1} + \alpha\left(\sum_{k=1}^{n-1} w_{i_k i_{k+1}} + w_{i_n 0}\right) > w_{ij_1} + \alpha\left(\sum_{\ell=1}^{m-1} w_{j_\ell j_{\ell+1}} + w_{j_m 0}\right).$$

The interpretation of this definition is as follows. Each node chooses edge weights for all possible outgoing links; *i.e.*, node i chooses a weight w_{ij} for each node j. Next, node i determines its rankings by ordering all paths $P_i = ii_1 \ldots i_n 0$ in increasing order according to their weight $w_{ii_1} + \alpha(\sum_{k=1}^{n-1} w_{i_k i_{k+1}} + w_{i_n 0})$, where α is a global parameter used to weight the tail of the path. The parameter α allows us to compare two extreme points: $\alpha = 1$, corresponds to shortest path routing based on the matrix of edge weights $\boldsymbol{w}$, while $\alpha = 0$ corresponds to next-hop rankings. A natural question to ask is whether a routing system using edge weight-based rankings can be safe for intermediate values of α. It turns out that the *only* edge weight-based ranking class that can guarantee safety (and safety under filtering), regardless of the weights chosen by each provider, is the scheme defined by $\alpha = 1$; *i.e.*, shortest path routing.

Observation 3 *A routing system with edge weight-based rankings may be unstable for any α where $0 < \alpha < 1$.*

Example 5 Consider the routing system shown in Figure 6. If the system is such that each node prefers the two-hop path to the destination, followed by the one-hop (*i.e.*, direct) path, followed by the three-hop path, then the system will be unstable because its behavior will match Example 1. The routing system will be unstable if the following conditions are satisfied, for all $i = 1, 2, 3$: $w_{i,i+1} + \alpha w_{i+1,0} < w_{i,0} < w_{i,i+1} + \alpha(w_{i+1,i+2} + w_{i+2,0})$ (for addition modulo 3). If $\alpha = 1$, these inequalities cannot be simultaneously satisfied for any nonnegative choice of the edge weight vector $\boldsymbol{w}$, which is expected, since $\alpha = 1$ corresponds to shortest path routing. On the other hand, if $0 < \alpha < 1$, then there are many vectors $\boldsymbol{w}$ that satisfy the inequalities above. For example, we can choose $w_{10} = w_{20} = w_{30} = 1$, and let $w_{12} = w_{23} = w_{31} = x$, for any x such that $(1 - \alpha)/(1 + \alpha) < x < 1 - \alpha$. For this definition of $\boldsymbol{w}$, all three inequalities above will be satisfied, and thus the rankings of each node will lead to the same oscillation described in Example 1. ■

5. Dispute Wheels and Dispute Rings

Our goal is to study the classes of rankings for which the routing system is guaranteed to be safe under filtering. Griffin *et al.* have shown that checking whether a particular routing system is safe is NP-hard [12]. To simplify our study of safety, we introduce a useful concept developed by Griffin *et al.* [12], known as a *dispute wheel*. Informally, a dispute wheel gives a listing of nodes, and two path choices per node, such that one path is always preferred to the other. If a routing system oscillates, then it is possible to construct a dispute wheel whereby each node in the wheel selects its more preferred path (via the node in the clockwise direction) over its less

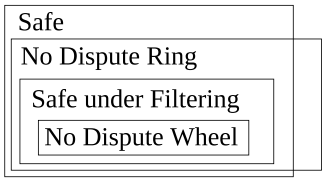


Figure 7: Relationships between safety and dispute rings and wheels. Previous work showed that a routing system with no dispute wheel is safe [12]. Section 5 presents all other relationships shown in this figure.

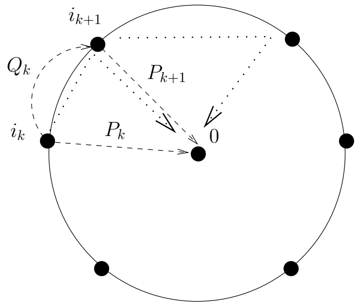


Figure 8: Illustration of a dispute wheel. Dotted lines show preferred (indirect) paths to the destination. The nodes $i_1, \ldots, i_m$ are pivots.

preferred path. Griffin *et al.* showed that if a routing system with no filtering does not have a dispute wheel, then it is safe.

The dispute wheel is a useful concept because it allows us to analyze dynamic properties such as safety by simply looking at the rankings of each node in the routing system. In this section, we formally define a dispute wheel and show the relationship of Griffin's routing model, which simulates messages being passed between nodes, to the model we use in this paper, which uses fair activation sequences. This relationship allows us to study safety in terms of the routing model in this paper. We then introduce a special type of dispute wheel called a *dispute ring* and show that, if any routing system has a dispute ring, then it is not safe under filtering. Finally, we relate dispute wheels to dispute rings and show that, although the presence of a dispute ring guarantees that a routing system is not safe under filtering, it does not necessarily imply that a routing system is not safe without filtering. Figure 7 summarizes the results of this section and how they relate to results from previous work [12].

5.1 Dispute Wheels and Safety

Defi nition 15 (Dispute wheel) *Given a routing system* $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$, *a* dispute wheel *is a collection of distinct nodes* $i_1, \ldots, i_m$, *called* pivots, *together with two sets of paths* $P_1, \ldots, P_m$ *and* $Q_1, \ldots, Q_m$, *such that the following conditions all hold (where we define* $i_{m+1} = i_1$ *for notational convenience):*

1. $P_k \in \mathcal{F}_{i_k}$ *for all* $k = 1, \ldots, m$;
2. Q_k *is a path from* i_k *to* i_{k+1} *for all* $k = 1, \ldots, m$;
3. *The path* $\hat{P}_k = i_k Q_k i_{k+1} P_{k+1} 0$ *is feasible,* i.e., $\hat{P}_k \in \mathcal{F}_{i_k}$,
4. $\hat{P}_k \succ_{i_k} P_k$.

Thus, each node i_k prefers the path $i_k Q_k i_{k+1} P_{k+1} 0$ to the path $i_k P_k 0$, as shown in Figure 8.

We now show that safety in the Simple Path Vector Protocol (SPVP) defined by Griffin *et al.* [12] implies safety in our model, which allows us to use dispute wheels to analyze safety.

Proposition 2 *Given a routing system, a fair activation sequence, and an initial path assignment* $\boldsymbol{P}_0$, *let* $\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots$ *be the resulting sequence of path assignments according to the dynamics described in Figure 3. Then there exists a sequence of messages in the Simple Path Vector Protocol (SPVP) such that the same sequence of path assignments is observed.*

Thus, in particular, if a routing system is safe under SPVP, then it is safe according to Definition 10.

Proof Sketch. The key difference between SPVP and the dynamics we have defined is that SPVP is *asynchronous* (*i.e.*, at any time step, messages may be in flight), so different nodes may have different assumptions about the global path assignment at any time. SPVP is *nondeterministic* with respect to the timing of messages; the delay between a routing update at node j and the receipt of the new route advertisement from node j at node i can be arbitrary. We use this fact to construct, inductively, a sequence of messages such that at time t, the current set of paths available to node i_t in SPVP corresponds exactly to $\boldsymbol{P}_{t-1}$. Furthermore, we time the delivery of routing updates to node i_t in SPVP so that any updates that occurred since the last time i_t was activated arrive exactly at the start of time step t. In SPVP, this will initiate a routing update at node i_t, which corresponds exactly to the activation of i_t in our model (see Figure 3).

Thus, the sequence of path assignments seen in this realization of SPVP matches the sequence of path assignments seen in our dynamics. We conclude that if SPVP is guaranteed to be safe for the given routing system (*i.e.*, if eventually no further routing updates occur, regardless of the initial path assignment), then the routing system is safe according to Definition 10 as well. ■

Corollary 1 *If a routing system* $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ *has no dispute wheel, then it is safe under filtering (and hence safe).*

Proof. Choose subsets $\hat{\mathcal{F}}_i \subseteq \mathcal{F}_i$. Then, any dispute wheel for the routing system $\hat{S} = (N, \prec_1, \ldots, \prec_N, \hat{\mathcal{F}}_1, \ldots, \hat{\mathcal{F}}_N)$ is also a dispute wheel for the original routing system $S = (N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$. Thus, the result follows from Proposition 2 and the results of [12]. ■

If no dispute wheel exists, the routing system is safe under filtering, but, unfortunately, this condition is not a necessary condition for safety, and thus not much can be said about a system that does have a dispute wheel. Furthermore, there exist routing systems that have a dispute wheel but which are safe under filtering.

Observation 4 *The existence of a dispute wheel does* not *imply that the routing system is unsafe, nor that the routing system is not safe under filtering.*

Example 6 See Figure 9. The first two most preferred paths in each node's ranking form a dispute wheel, but the system is safe: the system converges to $\boldsymbol{P} = (10, 20, 30)$. Furthermore, *no combination of filters can create an oscillation.* The two-hop paths are not part of the stable path assignment, so filtering those paths has no effect on the protocol dynamics. Filtering a three-hop path would simply result in a node selecting the direct path to the destination, and the node would never deviate from that path. If one direct path

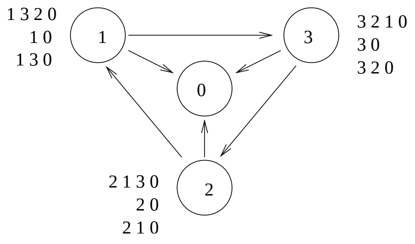


Figure 9: A routing system that is safe for any choice of filters.

is filtered, then the other two nodes will take direct paths to the destination and the node whose direct path is filtered will take its most preferred three-hop path. If two direct paths are filtered, then $\boldsymbol{P}$ is simply a chain to the destination: the node that has the direct path takes it, and the other two nodes will take two and three-hop paths. ■

5.2 Dispute Rings and Safety

In this section, we extend the dispute wheel notion to understand the relationship between ranking expressiveness and safety under filtering. We define a relationship between rankings called a *dispute ring*, a special case of a dispute wheel where each node appears at most once. The dispute ring is a useful concept because it allows us to prove a *necessary* condition for safety under filtering.

Definition 16 (Dispute ring) *A* dispute ring *is a dispute wheel—a collection of nodes $i_1, \ldots, i_m$ and paths $P_1, \ldots, P_m$, $Q_1, \ldots, Q_m$ satisfying Definition 15—such that $m \geq 3$, and no node in the routing system appears more than once in the wheel.*

Proposition 3 *If a routing system has a dispute ring, then it is not safe under filtering.*

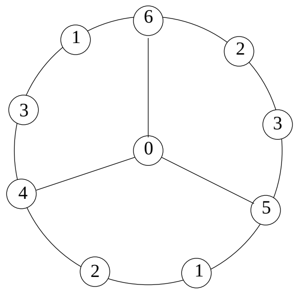
Proof. Assume that a routing system has a dispute ring, defined by $i_1, \ldots, i_m$, and paths $Q_1, \ldots, Q_m$, $P_1, \ldots, P_m$. Then, construct filters such that $\mathcal{F}_i$ contains *only* the paths in that dispute ring. Specifically, $\mathcal{F}_i$ contains the following paths from $\mathcal{P}_i^N$ (where we define $i_{m+1} = i_1$). (1) If i is not in the dispute ring, then $\mathcal{F}_i = \emptyset$. (2) If i is a pivot node on the dispute ring, say $i = i_k$, then $\mathcal{F}_i$ contains exactly two paths: P_k, and $i_k Q_k i_{k+1} P_{k+1} 0$. (3) If i is not a pivot node, but $i \in Q_k$ for some k, then we can write $Q_k = i_k Q_k^1 i Q_k^2 i_{k+1}$. In this case $\mathcal{F}_i$ consists of the single path $i Q_k^2 i_{k+1} P_{k+1} 0$. (4) If i is not a pivot node, but $i \in P_k$ for some k, then we can write $P_k = i_k P_k^1 i P_k^2 0$. In this case, $\mathcal{F}_i$ consists of the single path $i P_k^2 0$. Since each node appears at most once on the dispute ring, the preceding definition uniquely defines $\mathcal{F}_i$ for all nodes i.

There exists at least one consistent path assignment $\boldsymbol{P}_t$ such that some pivot node i_{k-1} uses its most preferred path, $i_{k-1} Q_{k-1} i_k P_k 0$, every other pivot node i_j uses path $i_j P_j 0$, and every other non-pivot node i uses its only available path consistent with this assignment. Then, the following activation sequence will result in an oscillation:

1. *Activate node i_k.* Node i_k then switches to its more preferred path, $i_k Q_k i_{k+1} P_{k+1} 0$.
2. *Activate nodes along Q_{k-1} in reverse order, from the node immediately preceding i_k, to i_{k-1}.* All nodes along Q_{k-1} switch to the empty path, ε.
3. *Activate node i_{k-1}.* The path $i_{k-1} Q_{k-1} i_k P_k 0$ is now inconsistent, so i_{k-1} must switch to the path $i_{k-1} P_{k-1} 0$.

Node	*Ranking*
1	$160 \succ 1240$
2	$240 \succ 2350$
3	$350 \succ 3160$
4	$43160 \succ 40$
5	$51240 \succ 50$
6	$62350 \succ 60$

(a) Routing system



(b) Dispute wheel

Figure 10: System that (1) has no dispute ring and (2) is not safe.

	Path Assignment					
Act.	1	2	3	4	5	6
—	(1 2 4 0)	(2 4 0)	(3 5 0)	(4 0)	(5 0)	(6 0)
5	(1 2 4 0)	(2 4 0)	(3 5 0)	(4 0)	**(5 1 2 4 0)**	(6 0)
1	**(1 6 0)**	(2 4 0)	(3 5 0)	(4 0)	(5 1 2 4 0)	(6 0)
3	(1 6 0)	(2 4 0)	**(3 1 6 0)**	(4 0)	(5 1 2 4 0)	(6 0)
4	(1 6 0)	(2 4 0)	(3 1 6 0)	**(4 3 1 6 0)**	(5 1 2 4 0)	(6 0)
5	(1 6 0)	(2 4 0)	(3 1 6 0)	(4 3 1 6 0)	**(5 0)**	(6 0)
3	(1 6 0)	(2 4 0)	**(3 5 0)**	(4 3 1 6 0)	(5 0)	(6 0)
2	(1 6 0)	**(2 3 5 0)**	(3 5 0)	(4 3 1 6 0)	(5 0)	(6 0)
6	(1 6 0)	(2 3 5 0)	(3 5 0)	(4 3 1 6 0)	(5 0)	**(6 2 3 5 0)**
4	(1 6 0)	(2 3 5 0)	(3 5 0)	**(4 0)**	(5 0)	(6 2 3 5 0)
2	(1 6 0)	**(2 4 0)**	(3 5 0)	(4 0)	(5 0)	(6 2 3 5 0)
1	**(1 2 4 0)**	(2 4 0)	(3 5 0)	(4 0)	(5 0)	(6 2 3 5 0)
5	(1 2 4 0)	(2 4 0)	(3 5 0)	(4 0)	**(5 1 2 4 0)**	(6 2 3 5 0)
6	(1 2 4 0)	(2 4 0)	(3 5 0)	(4 0)	(5 1 2 4 0)	**(6 0)**

Figure 11: Activation sequence for unsafe system from Figure 10.

4. *Return to Step 1 with k replaced by $k+1$, and iterate again.*

By the fourth step of the iteration above, the new path assignment is "isomorphic" to the initial configuration: now node i_k is using the path $i_k Q_k i_{k+1} P_{k+1} 0$, and every other pivot node i_j is using path $i_j P_j 0$. Thus, as this iteration repeats, the dynamics will ultimately reach node i_k once again with the original path assignment. Note that all paths in this activation sequence are guaranteed to be available and consistent, by the definition of $\mathcal{F}_i$. To make this activation sequence fair, we must also activate the nodes that are not in $P_i \cup Q_i$ for any i in the dispute ring; and the non-pivot nodes in P_i for all i in the dispute ring. The nodes that are not in $P_i \cup Q_i$ for any i have only the path ε available, and each non-pivot node in P_i (for all i) has only one path to the destination available. Therefore, these nodes will never change paths, and do not affect the oscillation. ■

We emphasize that, for simplicity, we reduced the set of filters, $\mathcal{F}_i$, to include only the set of paths that are involved in an oscillation. We note that there will typically be more permissive sets $\mathcal{F}_i$ that will also result in oscillation, because the dispute ring is present in the underlying set of rankings. Our intent is to highlight the most basic case of filtering that can cause an oscillation, given the existence of a dispute ring.

Despite the fact that systems that are safe under filtering are guaranteed not to have a dispute ring, testing for a dispute ring is not sufficient to guarantee that the routing system is safe, because of the following observation

Observation 5 *Routing systems that have a dispute wheel but do not have a dispute ring may not be safe.*

Example 7 Consider the routing system described by Figure 10(a) and the corresponding dispute wheel in Figure 10(b). Suppose that nodes 1, 2, and 3 prefer two-hop paths over three-hop paths, and the only paths available to nodes are those depicted in the figure. This system is not safe; for example, suppose $\boldsymbol{P}_0 = (1240, 240, 350, 40, 50, 60)$. The system then oscillates as shown in Figure 11. However, the system has no dispute ring; in particular, the dispute wheel depicted in Figure 10(b) cannot be reduced to a dispute ring. ■

6. Autonomy and Safety

In this section, we determine necessary and sufficient constraints on the allowable classes of rankings, such that if each AS autonomously sets its ranking while filtering is unrestricted, the protocol is guaranteed to be safe. We do so by characterizing whether a routing system where rankings are independently specified by each AS can induce either a dispute ring or a dispute wheel.

Any protocol's configurable parameters implicitly constrain the rankings ASes can express. For example, in BGP, the set of protocol parameters is rich enough to allow providers to express essentially any possible ranking over paths. In Section 6.1, we axiomatically formulate two properties that should be satisfied by any protocol that respects autonomy: *permutation invariance* and *scale invariance*. The first requires the rankings allowed by the protocol to be independent of node labeling, while the second requires the allowed rankings to scale gracefully as nodes are added to the system. We abstract protocols satisfying these two conditions using the notion of an *autonomous ranking constraint* (ARC) function; such a function takes the ranking of a single AS as input, and accepts it if that ranking is allowed by the protocol. Observe that *any* protocol that respects the ability of ASes to autonomously choose rankings can be represented by a corresponding ARC function.

In Section 6.2, we give two examples of such functions: the shortest hop count ARC function (which only accepts rankings where shorter paths are preferred to longer paths), and the next-hop ARC function (which only accepts next hop rankings). We then determine the class of ARC functions such that, as long as each node independently chooses an acceptable ranking, the resulting *global* routing system will be safe under filtering. In Section 6.3, we show that the only ARC functions that are safe under filtering are nearly equivalent to the shortest hop count ARC function.

6.1 ARC Functions

In this section, we define an *autonomous ranking constraint* (ARC) function, which serves as an abstraction of the protocol's constraints on allowed rankings over routes. We start by defining a local ranking constraint (RC) function, which takes as input the ranking of a single AS i, $\prec_i^N$ and determines whether that ranking is allowable.

Definition 17 (Local RC function) *Given N nodes, a* local ranking constraint (RC) function $\pi(\prec_i)$ *takes as input the ranking of a single AS i over all paths in $\mathcal{P}_i^N$, and returns "accept" if $\prec_i$ is allowed by π, and returns "reject" otherwise. If $\pi(\prec_i) =$ "accept", we will say that $\prec_i$ is π-accepted. If we are given a routing system $(N, \prec_1, \ldots, \prec_N, \mathcal{F}_1, \ldots, \mathcal{F}_N)$ where each $\prec_i$ is π-accepted, we will say the routing system is π-accepted.*

Because we are restricting attention to protocols that respect the ability of ASes to choose rankings autonomously, a first condition that must be satisfied is that constraints on rankings should be "local": that is, an AS should not face constraints on allowable rankings due to the rankings chosen by other ASes. For this reason, *local* RC functions act only on the ranking of a single AS. More generally, protocols might place system-wide constraints on the vector of rankings chosen by all ASes; such protocols should be represented by "global" RC functions. Of course, such protocols do not respect autonomy, and so we do not consider them here.

We now define two natural conditions any local RC function that preserves autonomy should satisfy. First, the local RC function's conditions on rankings should provide consistent answers to different ASes, regardless of the *labeling* of the ASes. That is, for the local RC function to preserve uniformity, each AS should be subject to the same constraints on routing policies, and those constraints should not depend on the particular assignment of AS numbers to ASes. For example, suppose the routing system consists of three ASes, and AS 1 has an accepted ranking where it prefers 1230 over 120, and 120 over 10. Then we expect the same ranking should be accepted, even if the labels of nodes are *permuted*. For example, suppose we permute the node labels that $1 \to 2$, $2 \to 3$, and $3 \to 1$. Then node 2 should also have an accepted ranking where it prefers 2310 over 230, and 230 over 20 (because 2310, 230, and 20 are the new paths that result after applying the permutation to 1230, 120, and 10, respectively). If this property were not satisfied, then the set of accepted rankings determined by a local RC function would depend on the global assignment of AS numbers to nodes, not on the characteristics of the individual rankings themselves. We call this notion *permutation invariance*; to define it precisely, we must proceed through a sequence of definitions, starting with *path permutation*.

Definition 18 (Path permutation) *Given N nodes, let σ be a permutation of the nodes $1, \ldots, N$. Then σ induces a* path permutation *on any path $P = ii_1i_2 \ldots i_mj$ from i to j, yielding a new path $\sigma(P) = \sigma(i)\sigma(i_1)\sigma(i_2)\ldots\sigma(i_m)\sigma(j)$ from $\sigma(i)$ to $\sigma(j)$. We always define $\sigma(0) = 0$.*

Definition 19 (Ranking permutation) *Given N nodes, let σ be a permutation of the nodes $1, \ldots, N$. Then σ induces a* ranking permutation *on a ranking $\prec_i$ for node i over the paths in $\mathcal{P}_i^N$, yielding a new ranking $\sigma(\prec_i)$ over the paths in $\mathcal{P}_{\sigma(i)}^N$, as follows: If $P_1, P_2 \in \mathcal{P}_i^N$, and $P_1 \prec_i P_2$, then $\sigma(P_1)\sigma(\prec_i)\sigma(P_2)$ (where $\sigma(P_i)$ is the path permutation of path P_i under σ).*

Note that a permutation does not modify the routing system any substantive way, except to *relabel* the nodes, and to relabel the paths and rankings and in a way that is consistent with the relabeling of nodes.

Definition 20 (Permutation invariance) *A local RC function π is* permutation invariant *if, given N and a ranking $\prec_i$ for an AS i over all paths in $\mathcal{P}_i^N$, the relation $\prec_i$ is π-accepted if and only if $\sigma(\prec_i)$ is π-accepted, for any permutation σ of $1, \ldots, N$.*

Second, a local RC function should specify conditions for acceptance or rejection of rankings that "scale" appropriately with the number of nodes in the system; we call this property *scale invariance*. Suppose, for example, that a local RC function accepts a ranking $\prec_i$ over $\mathcal{P}_i^N$, when N nodes are in the system. Now suppose that we add nodes to the system, so the total number of nodes is $\hat{N} > N$. As additional nodes are added to the system, additional paths become available as well, and each node i must specify its rankings over the larger set $\mathcal{P}_i^{\hat{N}}$. Informally, scale invariance of the local RC function requires that i should be able to "extend" the ranking $\prec_i$ to an accepted ranking over $\mathcal{P}_i^{\hat{N}}$, without having to modify its existing ranking over $\mathcal{P}_i^N$; otherwise, the properties of allowed rankings would depend on the number of nodes in the global system.

To formalize this concept, we first define a subranking.

Definition 21 (Subranking) *Given N nodes, let $\prec_i$ be a ranking for AS i over all paths in $\mathcal{P}_i^N$. Given $\hat{N} > N$, let $\hat{\prec}_i$ be a ranking for AS i over all paths in $\mathcal{P}_i^{\hat{N}}$. Note that $\mathcal{P}_i^N \subset \mathcal{P}_i^{\hat{N}}$. We say that $\prec_i$ is a* subranking *of $\hat{\prec}_i$ if $P_1 \prec_i P_2$ implies $P_1 \hat{\prec}_i P_2$, for all $P_1, P_2 \in \mathcal{P}_i^N$.*

We now define scale invariance.

Definition 22 (Scale invariance) *A local RC function π is* scale invariant *if the following condition holds: given any π-accepted ranking $\prec_i$ for AS i over $\mathcal{P}_i^N$, and given any $\hat{N} > N$, there exists at least one π-accepted ranking $\hat{\prec}_i$ over $\mathcal{P}_i^{\hat{N}}$ that has $\prec_i$ as a subranking.*

Permutation invariance guarantees that relabeling nodes does not affect allowed rankings; scale invariance ensures that even as the set of nodes in the network increases, the rankings over previously existing paths should remain valid. Local RC functions that satisfy both permutation invariance and scale invariance correspond to protocols that respect the ability of ASes to autonomously choose rankings; we call such functions *autonomous ranking constraint functions*.

Definition 23 (ARC function) *A local RC function is an* autonomous ranking constraint (ARC) function *if it is both permutation invariant and scale invariant.*

We want to derive the conditions under which protocols are guaranteed to be safe under filtering. Given that we use an ARC function as an abstraction of the constraints placed by a protocol on rankings, we would thus like to characterize ARC functions that can ensure safety under filtering of the entire routing system (a global property). For this reason, we extend the definition of "safety under filtering" to cover local RC functions.

Definition 24 *Let π be a local RC function. We say that π is* safe under filtering *if all π-accepted routing systems are safe under filtering.*

6.2 Examples of ARC Functions

We now present two simple examples of ARC functions: the shortest hop count ARC function, which is guaranteed to be safe, but is not expressive; and the next hop ARC function, which is expressive, but not safe.

Example 8 Our first example is the *shortest hop count RC function*, π^{shc}. Given the number of nodes N, the RC function π^{shc} accepts a ranking $\prec_i$ for node i if and only if the relation $\prec_i$ strictly prefers shorter paths (in terms of hop count) over longer ones. Formally, it accepts $\prec_i$, if, for any two paths $P_i, \hat{P}_i \in \mathcal{P}_i^N$ such that $\mathrm{length}(P_i) < \mathrm{length}(\hat{P}_i)$, $P_i \succ_i \hat{P}_i$. Ties may be broken arbitrarily.

It is not hard to verify that π^{shc} is an ARC function. To check permutation invariance, note that if $\prec_i$ is allowed for node i, then of course for any permutation σ, the ranking $\sigma(\prec_i)$ will also be allowed for node $\sigma(i)$, as $\sigma(\prec_i)$ will also prefer shorter paths to longer paths. Scale invariance is natural: given any shortest hop count ranking $\prec_i$ over $\mathcal{P}_i^N$, and given $\hat{N} > N$, there obviously exists at least one shortest hop count ranking over $\mathcal{P}_i^{\hat{N}}$ that has $\prec_i$ as a subranking. ■

π^{shc} forces all providers to use shortest AS path length, effectively precluding each AS from having any policy expressiveness in choosing rankings (other than when breaking ties). A more flexible set of rankings is allowed by the *next hop RC function* of the next example.

Example 9 The *next hop RC function*, π^{nh}, accepts a ranking $\prec_i$ for node i if and only if $\prec_i$ satisfies Equation (1) in Section 4.1; that is, if $\prec_i$ is a next hop ranking.

π^{nh} is clearly permutation invariant: if $\prec_i$ is a next hop ranking for node i, then clearly $\sigma(\prec_i)$ is a next hop ranking for node $\sigma(i)$. Furthermore, note that any next hop ranking $\prec_i$ is determined entirely by the rankings of node i over each possible next hop, together with tiebreaking choices among routes with the same next hop. Thus, for $\hat{N} > N$, $\prec_i$ can be extended to a next hop ranking over $\mathcal{P}_i^{\hat{N}}$, by extending node i's rankings over each possible next hop, and determining tiebreaking rules for any routes with next hop $N+1, \ldots, \hat{N}$. We conclude that π^{nh} is scale invariant as well, and thus it is an ARC function.

π^{nh} grants greater flexibility in choosing routing policies than under the shortest hop count RC function, π^{shc}, albeit at some cost. With π^{nh}, each AS i will choose a next hop ranking $\prec_i$ without any *global* constraints on the composite vector of next hop rankings $(\prec_1, \ldots, \prec_N)$ chosen by the nodes. We have shown earlier in Section 4.1 that there exist configurations of next hop rankings that may not be stable or safe under filtering; thus, the ARC function π^{nh} can lead to a lack of safety. ■

Next, we use dispute rings and dispute wheels to characterize the class of ARC functions that are safe under filtering. We will prove that this class is closely related to the ARC function π^{shc}.

6.3 Impossibility Results

We prove two main results in this section. Informally, the first result can be stated as follows: suppose we are given an ARC function and an accepted ranking such that some n hop path is *less preferred* (*i.e.*, ranked lower) than another path of length at least $n+2$ hops. Then, we can construct an accepted routing system with a dispute ring; *i.e.*, one that is not safe under filtering. The second result states that if some n-hop path is *less preferred* than another path of length at least $n+1$ hops, then there exists a routing system with a dispute wheel (though not necessarily a dispute ring). Note that this result is weaker than our first result, because a dispute wheel does not necessarily imply that the system is not safe under filtering.

We interpret these results as follows: if we are searching for ARC functions that are safe under filtering, we are very nearly restricted to considering only the shortest hop count ARC function, because all paths of n hops *must be more preferred* than paths of at least $n+2$ hops to guarantee safety under filtering, and all paths of n hops must be more preferred than paths of at least $n+1$ hops to prevent dispute wheels.

Our first lemma, which is crucial to proving both of our results, uses permutation invariance to construct a dispute wheel from a single node's rankings. We use a permutation to "replicate" pieces of the dispute wheel until the entire wheel is completed.

To state the lemma, we will require the definition of *period* of a node with respect to a permutation, as well as the period of a permutation. Given a permutation σ on the nodes $1, \ldots, N$, let σ^k denote the permutation that results when σ is applied k times; *e.g.*, $\sigma^2(j) = \sigma(\sigma(j))$, where σ^0 is defined to be σ.

Definition 25 (Period) *Given a permutation σ on the nodes $1,\dots,N$, we define the* period of i *under σ as* $\mathrm{period}_i(\sigma) = \min\{k \geq 1 : \sigma^k(i) = i\}$.

Thus, the period of i is the minimum number of applications of σ required to return to i.

Definition 26 (Permutation period) *Given a permutation σ on the nodes $1,\dots,N$, we define the* period of the permutation σ *as* $\mathrm{period}(\sigma) = \min\{k \geq 1 : \sigma^k(i) = i \text{ for all } i\}$.

Thus, $\mathrm{period}(\sigma)$ *is the minimum number of applications of σ required to recover the original node labeling, and can be computed as the least common multiple of* $\mathrm{period}_i(\sigma)$, *for $1 \leq i \leq N$.*

The following result establishes the conditions under which we can apply a permutation to a π-accepted ranking to obtain a dispute wheel. We use this lemma as a building block for both of the theorems in this section.

Lemma 1 *Let π be an ARC function. Suppose there exists a node i with a ranking $\prec_i$ over $\mathcal{P}_i^N$, two paths $R_i, \hat{P}_i \in \mathcal{P}_i^N$, and a permutation σ on $1,\dots,N$ such that: (1) $\prec_i$ is π-accepted; (2) $R_i \succ_i \hat{P}_i$; (3) $\mathrm{period}_i(\sigma) = \mathrm{period}(\sigma)$; and (4) there exists a path $\hat{Q}_i$ from i to $\sigma(i)$ such that:*

$$R_i = i\hat{Q}_i\sigma(i)\sigma(\hat{P}_i)0. \tag{2}$$

Then there exists a π-accepted routing system with a dispute wheel.

This dispute wheel is defined by pivot nodes $i_1,\dots,i_m$, and paths $P_1,\dots,P_m$ and $Q_1,\dots,Q_m$, where $m = \mathrm{period}(\sigma)$, and where for $k = 1,\dots,m$, we have $i_k = \sigma^{k-1}(i)$, $P_k = \sigma^{k-1}(\hat{P}_i)$, and $Q_k = \sigma^{k-1}(\hat{Q}_i)$.

Proof Sketch. Details are in the technical report [5]. The key idea of the proof is that, because $\mathrm{period}_i(\sigma) = \mathrm{period}(\sigma)$, we can repeatedly apply σ to the paths $\hat{Q}_i$ and $\hat{P}_i$ and apply permutation invariance to construct a π-accepted routing system with a dispute wheel. ■

The preceding lemma reduces the search for a dispute wheel to a search for a permutation and a π-accepted ranking with the stated properties. Observe from Equation (2) that the permutation σ maps the path $\hat{P}_i$ into the "tail" of the path R_i; in applying Lemma 1, we will construct a partial permutation by mapping a path $\hat{P}_i$ into the "tail" of R_i as in (2), and then we will complete the permutation by adding nodes to the system if necessary so that $\mathrm{period}_i(\sigma) = \mathrm{period}(\sigma)$. We use this approach to prove two theorems; the first states that if an ARC function accepts at least one ranking that prefers an n-hop path less than a path of at least $n+2$ hops, then the ARC function is not safe under filtering.

Theorem 1 *Let π be an ARC function. Suppose there exists a node i with π-accepted ranking $\prec_i$, and two paths $R_i, \hat{P}_i \in \mathcal{P}_i^N$ such that* $\mathrm{length}(R_i) > \mathrm{length}(\hat{P}_i) + 1$ *and $R_i \succ_i \hat{P}_i$. Then, π is not safe under filtering.*

Proof Sketch. Details are in the technical report [5]. The proof relies on Lemma 1 to build a dispute wheel. First, using scale invariance of the ARC function, we show that the stated conditions of the theorem ensure that there exist two paths R_i', $\hat{P}_i'$ such that: $\mathrm{length}(R_i') \geq \mathrm{length}(\hat{P}_i') + 1$; R_i' is more preferred than $\hat{P}_i'$ for some π-accepted ranking; and R_i' and $\hat{P}_i'$ have no nodes in common, other than i and 0.

We then use scale invariance and Lemma 1. We add nodes to the system, and use them to build a permutation σ satisfying the conditions of Lemma 1 (in particular, so that P_i' is mapped into the tail of R_i'). Furthermore, we show that because P_i' and R_i' share no nodes in common, the dispute wheel of Lemma 1 is in fact a dispute ring. This completes the proof, since by Proposition 3 the resulting routing system is not safe under filtering. ■

The preceding theorem suggests that ARC functions that are safe under filtering are closely related to the shortest hop count ARC function, because no rankings can be accepted where n hop paths are less preferred than $(n+k)$-hop paths, for $k \geq 2$. The next theorem draws this relationship even closer, by proving that there exists a dispute wheel if an ARC function accepts any ranking where an n-hop path is less preferred than an $(n+1)$-hop path.

Theorem 2 *Let π be an ARC function. Suppose there exists a node i with π-accepted ranking $\prec_i$, and two paths $R_i, \hat{P}_i \in \mathcal{P}_i^N$ such that* $\mathrm{length}(R_i) = \mathrm{length}(\hat{P}_i) + 1$ *and $R_i \succ_i \hat{P}_i$. Then there exists a π-accepted routing system with a dispute wheel.*

Proof Sketch. Details are in the technical report [5]. Our approach is to map the path $\hat{P}_i$ into the "tail" of the path R_i, which partially defines a permutation σ. We then show how to add nodes to the system and complete the permutation σ so that $\mathrm{period}_i(\sigma) = \mathrm{period}(\sigma)$. We then apply Lemma 1 to conclude there exists a π-accepted routing system with a dispute wheel. ■

The preceding results should not be interpreted as suggesting that we cannot find a routing system that is safe under filtering, where nodes prefer $(n+k)$-hop paths over n-hop paths. Indeed, from Example 6, there are routing systems where nodes prefer 3-hop paths over 1-hop paths, and yet the system is safe under filtering. However, checking whether such systems are safe under filtering requires global verification; the theorems we have presented suggest that safety under filtering cannot be guaranteed through local verification alone, if some nodes are allowed to prefer $(n+k)$-hop paths over n-hop paths.

Furthermore, the two results in this section highlight the importance of dispute rings. Theorem 1 gives the strong result that an ARC function that allows some $(n+k)$-hop path to be more preferred than an n-hop path cannot guarantee safety under filtering, if $k \geq 2$. Theorem 2 only guarantees the existence of a dispute wheel if an ARC function that allows some $(n+1)$-hop path to be more preferred than an n-hop path—we cannot draw conclusions regarding the stability or safety of a routing system on the basis of the existence of a dispute wheel (see Example 6).

7. Conclusion

This paper explored the fundamental tradeoff between the expressiveness of rankings and routing safety, presuming that each AS retains complete autonomy and filtering expressiveness. We make three main contributions. First, we show that next-hop rankings are not safe; we also observe that although rankings based on a globally consistent weighting of paths are safe under filtering, even minor generalizations of the weighting function compromise safety. Second, we define a *dispute ring* and show that any routing system that has a dispute ring is not safe under filtering. Third, we show that, providing for complete autonomy and filtering expressiveness, the class of allowable rankings that guarantee safety is effectively ranking based on (weighted) shortest paths. We also presented the first study of the effects of filtering on stability.

In light these results, a natural question to ask is whether they are positive or negative. In one sense, our results are grim, because they suggest that if BGP remains in its current form and each AS retains complete autonomy and filtering expressiveness (*i.e.*, the only realistic scenario for the foreseeable future), then the routing system cannot be guaranteed to be safe unless each AS constrains its rankings over available paths to those that are consistent with shortest hop count (or, alternatively, preferences that are based on consistent edge weights).

On the other hand, our results suggest several clear directions for designing a policy-based routing protocol that is guaranteed to be safe. Although we presented the ARC function as a proof technique, such a function could be implemented in practice to restrict the rankings that operators specify in operational networks. We foresee two possibilities: (1) the routing protocol remains as it is today, and the constraints derived in Theorems 1 and 2 are checked by a tool that statically detects configuration faults (*e.g.*, *rcc* [3]); or (2) the routing protocol is modified to prevent operators from expressing rankings that violate these constraints in the first place. Although we have not yet fully evaluated the merits of each approach in this work, we now briefly speculate on advantages and disadvantages of each approach.

Enforcing the ranking constraints in a static checker would require no changes to the existing routing protocols and configuration languages. Unfortunately, the results in Theorems 1 and 2 only provide global guarantees if *every* AS abides by the constraints: as a result, there may be little incentive for one AS to overly constrain its policies if other ASes do not abide by the same constraints (and creating the potential for unsafe routing in the process).

Implementing the ranking constraints by restricting the protocol "knobs" requires wholesale changes to both the configuration language and the routing protocol, but it would provide absolute guarantees for safety while still preserving the autonomy of each AS. Our results in Section 4.2 suggest one possible direction: there we show that routing using preferences derived from edge weights is guaranteed to be stable. Suppose each AS ranks paths based on the sum of edge weights to the destination and adjusts weights on its incident outgoing edges to neighboring ASes. Rankings would then be derived from the total path cost, but an AS might still retain enough flexibility to control the next-hop AS en route to the destination. Such an approach could ensure that the protocol is safe on short timescales, while allowing "policy disputes" to occur on longer timescales, out-of-band from the routing protocol. Of course, we must still explore whether this apparently more restrictive language could still implement the policies that operators want to express. Furthermore, a more restrictive policy language would guarantee safety, but would likely cause routing to oscillate on a slower timescale as operators observe the routing protocol converging to undesirable paths. It is our position that this design decision is exactly the right one: the routing protocol *should* converge on a fast timescale and accurately reflect network topology, while policy conflicts should be resolved on slower, "human" timescales.

Acknowledgments

This work was supported by the NSF under Cooperative Agreement ANI-0225660 and a Cisco URP grant. Nick Feamster is partially supported by an NSF Graduate Research Fellowship.

REFERENCES

[1] ALAETTINOGLU, C., ET AL. Routing policy specification language (RPSL). RFC 2622, June 1999.

[2] Private communication with Randy Bush, May 2004.

[3] FEAMSTER, N., AND BALAKRISHNAN, H. Detecting BGP Configuration Faults with Static Analysis. In *Proc. 2nd Symposium on Networked Systems Design and Implementation* (Boston, MA, May 2005), pp. 49–56.

[4] FEAMSTER, N., BORKENHAGEN, J., AND REXFORD, J. Guidelines for interdomain traffic engineering. *Computer Communications Review 33*, 5 (October 2003), 19–30.

[5] FEAMSTER, N., JOHARI, R., AND BALAKRISHNAN, H. Stable policy routing with provider independence. Tech. Rep. MIT-LCS-TR-981, Massachusetts Institute of Technology, February 2005.

[6] FEIGENBAUM, J., SAMI, R., AND SHENKER, S. Mechanism design for policy routing. In *ACM Symposium on Principles of Distributed Computing* (2004), pp. 11–20.

[7] GAO, L., GRIFFIN, T. G., AND REXFORD, J. Inherently safe backup routing with BGP. In *Proc. IEEE INFOCOM* (Anchorage, AK, April 2001), pp. 547–556.

[8] GAO, L., AND REXFORD, J. Stable Internet routing without global coordination. *IEEE/ACM Trans. Networking 9*, 6 (December 2001), 681–692.

[9] GOVINDAN, R., ALAETTINOGLU, C., EDDY, G., KESSENS, D., KUMAR, S., AND LEE, W. An architecture for stable, analyzable Internet routing. *IEEE Network Magazine 13*, 1 (January/February 1999), 29–35.

[10] GOVINDAN, R., ALAETTINOGLU, C., VARADHAN, K., AND ESTRIN, D. Route servers for inter-domain routing. *Computer Networks and ISDN Systems 30* (1998), 1157–1174.

[11] GRIFFIN, T., JAGGARD, A., AND RAMACHANDRAN, V. Design principles of policy languages for path vector protocols. In *Proc. ACM SIGCOMM* (Karlsruhe, Germany, August 2003), pp. 61–72.

[12] GRIFFIN, T., SHEPHERD, F. B., AND WILFONG, G. The stable paths problem and interdomain routing. *IEEE/ACM Trans. Networking 10*, 1 (2002), 232–243.

[13] GRIFFIN, T., AND WILFONG, G. A safe path vector protocol. In *Proc. IEEE INFOCOM* (March 2000), pp. 490–499.

[14] JAGGARD, A. D., AND RAMACHANDRAN, V. Robustness of class-based path vector systems. In *Proc. International Conference on Network Protocols* (November 2004), pp. 84–93.

[15] MACHIRAJU, S., AND KATZ, R. Verifying global invariants in multi-provider distributed systems. In *Proc. SIGCOMM Workshop on Hot Topics in Networking (HotNets)* (November 2004), pp. 149–154.

[16] REKHTER, Y., AND LI, T. A Border Gateway Protocol. RFC 1771, March 1995.

[17] SOBRINHO, J. L. Network routing with path vector protocols: Theory and applications. In *Proc. ACM SIGCOMM* (Karlsruhe, Germany, August 2003), pp. 49–60.

[18] VARADHAN, K., GOVINDAN, R., AND ESTRIN, D. Persistent route oscillations in inter-domain routing. Tech. Rep. 96-631, USC/ISI, February 1996.