Routers of the AS, Unite! Guaranteeing a Network Realizes Its Routing Policy

Jennifer Rexford*, Gordon Wilfong†, Rui Zhang-Shen*
*Princeton University †Bell Laboratories

ABSTRACT

We study how the many routers in an AS should cooperate to provably realize a single routing policy, a question that has remained unanswered despite years of experience with policy-based interdomain routing. The simplest solution is to distribute all interdomain routing information to every router, but this is not scalable. Instead, intra-AS route dissemination should ensure that every router learns a *sufficient* set of routes to make decisions that comply with the policy. Unfortunately, mismatches between today's routing policies and route-dissemination protocols can easily lead to protocol oscillations, traffic blackholes, and violations of business contracts. This paper presents a systematic study of the role of route dissemination in realizing an AS's policy. We begin by defining a policy as an *AS-wide* route preference plus a *router-specific* preference, and show that minimizing intra-AS route dissemination while provably satisfying the policy is NP-complete in general. Fortunately, polynomial-time algorithms exist for today's typical policies and other policies of similar structure. Our analysis shows that each router advertising a single best route, as in today's internal BGP (iBGP) protocol, is not sufficient to realize some common policies. Our proposed changes *guarantee* that policies will be realized correctly, can be implemented by router features available in the near future, and simplify the router configuration process.

1. INTRODUCTION

The Internet consists of many individually administered networks, known as autonomous systems (ASes),[1] that connect to each other via interdomain routing. Despite many years of implementing and managing policy-based interdomain routing, the precise definition of a routing policy remains elusive. On the one hand, an AS's routing policy states how the network as a whole should attract and forward traffic, and is based on an AS's business objectives, such as using more profitable routes, conserving its own resources, balancing load, and adhering to contracts with other ASes. On the other hand, policies are often expressed in terms of router configurations, and routing is regarded a router-level operation. This raises the question of how a distributed collection of routers can realize a policy. The mismatch of concept and practice today can easily lead to undesirable behaviors, such as those presented in Section 2.2. That is, a seemingly reasonable policy cannot be realized even when the routers in an AS are configured in the best possible way. Policy violations can cause routing oscillations [1], traffic black holes, and violation of business contracts [2]; worse yet, they are often subtle and can go undetected for a long time.

The main reason for policy violations today is the *limited route visibility* within an AS, caused by each router selecting a single best route from the ones it has learned and propagating only that best route, even to other routers within the same AS. In the interdomain setting, ASes are assumed to be *competitive*, making it reasonable that each router discloses just enough information (i.e., one route) to maintain connectivity. In contrast, within an AS the routers are *cooperative*, and if realizing their joint goal (i.e., the AS's policy) requires the routers to share more information, they should do so. Today, the protocol designed for interdomain route exchange (i.e., BGP) is also used for route dissemination within an AS (in a slightly modified form, iBGP), even though this is not sufficient to realize an AS's policy.

In this paper we formally study the properties that a protocol, and in particular route dissemination, must satisfy in order to guarantee that the routers in an AS collectively realize the AS's policy. We find that the policy has a profound influence on the number of routes that must be disseminated to ensure that the chosen routes ultimately comply with the stated policy. For example, if the policy is as simple as strictly preferring shortest routes, then each router disseminating the single best route is sufficient; but today's common policies, which are slightly more complex, require that routers disseminate multiple routes. We also show that in some cases disseminating more routes than strictly necessary can lead to more rapid routing convergence.

[1]Individually administered networks do not necessarily have a one-to-one correspondence with AS numbers or business entities, e.g., a stub AS may not have an AS number, and a single company can manage multiple ASes.

A protocol can be viewed as a distributed algorithm that solves a central problem, either by design or by hindsight. For example, TCP maximizes a certain form of network utility [3], OSPF finds shortest paths on a graph [4], and BGP solves the stable paths problem [5]. For the problem we study (realizing an AS's routing policy), the protocol has two relevant aspects: information *dissemination* among routers within the AS and route *selection* by each router. Since we express the policy by preference relationships between routes, our goal, then, is to define protocols (i.e., define which routes to disseminate and, of the learned routes, which to select) that guarantee all routers select the most preferred routes. We call this the *Route Preference Realization* (RPR) problem. Because a router can only select routes based on the information it has learned, dissemination needs to ensure that every router in the AS learns sufficient information to make a correct selection decision. That is, a router should disseminate more than one route if necessary. The idea of disseminating additional routes within an AS to eliminate policy violations has been suggested before [1, 6, 7], but here we systematically examine the precise tradeoff between policy flexibility and dissemination overhead.

When every router disseminates all the routes it has learned, often even very complex polices can be realized, but the communication overhead may be too high. On the other hand, if the policy is extremely simple, such as strictly preferring shortest routes, then each router disseminating a single shortest route (as the protocol today would do) can be sufficient. Thus, there is a fundamental trade-off between policy flexibility (realize complex business objectives) and protocol scalability (minimize communication overhead). We focus on the space between the two extreme design points mentioned above, where policies can be sufficiently flexible but do not require extremely high communication overhead. This includes common routing policies implemented today.

To analyze the tradeoff between policy flexibility and protocol scalability, we need to speak precisely about them. Protocol scalability can be measured by the communication overhead, e.g., in terms of how many routes are propagated among the routers, while policy flexibility is much harder to quantify. Rather than investigate arbitrary policies that require every router to learn every route, we look for families of policies that can be realized in a distributed fashion and are commonly used today. Thus, we restrict policies as follows:[2]

- **Policies that make pairwise comparison of *individual* routes:** This precludes policies that make decisions based on a combination of routes, e.g., "use a route only if there are at least two other routes sharing the same next-hop AS." Note that it is possible that the decision of a router depends on the *order* of comparisons, as is the case today when the multi-exit discriminator [9] is used. As will be seen later, we can model such policies with *multiple* stages of pairwise comparisons where the routes tied at the top in the previous stage are entered into the next stage, so that within each stage only simple pairwise eliminations are needed.

- **Policies that can be realized if every router, given all the routes learned by the AS, could make a decision on its own:** This precludes policies that require explicit coordination of the routers' decisions, e.g., "while each router selects only one route, the whole AS should use as many different routes as possible."

We have narrowed down policies to those where each router makes selection and dissemination decisions by comparing the routes it has learned. If all routers in an AS agree on some of the comparison criteria (e.g., preferring shorter ones), communication can be reduced because certain routes (e.g., longer ones) need not be disseminated. The common ranking of routes is the *AS-wide route preference* (e.g., the first four steps of BGP route selection in Table 1), which can be null if the routers are not required to agree at all. Among the routes that are ranked highest according to the AS-wide route preference, each router may then apply *router-specific tie-breaking* (e.g., the last two steps in Table 1) to select its own favorite route. Thus, we view policies as consisting of an AS-wide route preference and a router-specific preference. Then, a policy is said to be realized if the route preferences are realized, i.e., each router ultimately selects a route (or routes) such that no other route known anywhere in the AS is more preferred according to the AS-wide policy, and of all those equally preferred in terms of the AS-wide policy, there are no known routes that are more preferred by the router-specific policy. We will show that both parts of the policy influence which routes need to be disseminated in order to satisfy the policy.

Hence, route dissemination needs to ensure that every router learns sufficient routes so that the AS solves the Route Preference Realization problem. We show that in general, it is NP-complete to compute the smallest set of routes that, if disseminated, are sufficient to guarantee that the route preferences will be realized. But we then show that for some commonly used policies, the smallest dissemination set can be easily computed in polynomial time. In particular, this is the case for policies expressed by BGP attributes, even when the multi-exit discriminator attribute is used.

Our analysis shows that there are two reasons to disseminate a route: (i) to *inform* other routers of a

[2]Centralized solutions, where route servers learn all routes and select routes on behalf of the routers [8], can implement more flexible policies but are not the focus of this paper.

route that they may want to select and use, and (ii) to *prevent* other routers from selecting some other less-preferred route. Sometimes these two missions overlap; for example, when the policy is simply preferring shortest routes, disseminating a shortest route achieves both goals. However, for some of today's routing policies, for instance those based on multi-exit discriminators, each router disseminating a single route is no longer sufficient [10]. Such policies require a slightly higher communication overhead than today's protocol. Thus, we propose an enhancement to the protocol (i.e., disseminating extra routes as specified by our analysis) to eliminate today's policy violations. The enhancement can be supported by add-path [11], a mechanism available in the near future, and requires no hardware change on the routers or coordination with other ASes[3].

When BGP was first invented, the goal was simply to connect multiple networks so that end hosts in different networks could communicate with each other. Over the years, more and more functionality has been added, in the form of new BGP attributes, new steps in the BGP decision process, and new configuration commands for expressing ever more complex routing policies. However, these extensions to BGP have been done in an ad hoc fashion, without careful study of the effects of new functionality or the interaction with existing features. Our work attempts to change this practice. We provide a theoretical framework for studying the protocol requirements that guarantee the realization of a routing policy, expressed in terms of route preferences. The policies we study are more general than those allowed by today's BGP. Thus, in the future when more flexibility is needed in routing policies, and new features are again added to BGP, our work can provide guidance for the best way to enhance flexibility without causing policy violations or large increases in dissemination overhead.

The remainder of the paper is structured as follows. Section 2 briefly introduces the BGP protocol, and gives examples of policy violations. Section 3 describes the formal model and definitions of a policy, protocol, and what it means for a protocol to realize a policy. Then in Section 4 we present our results on sufficient conditions for a protocol to realize a policy, minimizing the number of routes to disseminate while still satisfying the conditions, and the effects of additional route dissemination on convergence time. Section 5 discusses some simple extensions to our model and practical issues for changing the iBGP protocol. Finally, Section 6 presents related work, and Section 7 concludes with a discussion of future research directions.

[3]Disseminating more than one route may cause confusion about which route incoming data traffic should follow; a simple and practical solution is to encapsulate incoming packets, as discussed in more detail in Section 5.3.

Stage	Step	Attribute comparison
i	1	Highest local_pref
	2	Lowest AS path length
	3	Lowest origin type
ii	4	Lowest MED (with *same* next-hop AS)
iii	5	Closest exit point
	6	Lowest Router ID

Table 1: The six-step BGP route-selection process divided into three stages.

2. PROBLEMS WITH TODAY'S IBGP

This section will briefly describe how policy routing works today. In particular, we first overview BGP and then give examples of policy violations.

2.1 BGP Viewed from a Single AS

In order to understand how routes are disseminated and selected today, we briefly describe the BGP protocol from the viewpoint of a single AS, based on how most routers are implemented and used today. Routes are imported and exported at the edge routers. A route announcement identifies the downstream path that data traffic would traverse, and some other attributes to describe the properties of the path. An edge router may perform *import actions* to modify some route attributes when routes are first learned from other ASes, with the goal of influencing the route-selection process. For each destination prefix, a router selects at most one route by comparing its learned routes in the *route-selection* process, and may advertise its selected route to other routers. We say it *exports* the route if the receiving routers are *outside* the AS; and it *disseminates* the route if the receiving routers are *inside* the AS. An edge router also performs *export actions*, which may modify or filter the selected route before exporting it to an external neighbor. Thus, viewed from the outside, an AS learns some routes from its neighbors, and exports some routes. Given the set of routes learned by the AS, what routes are exported and where they are exported should comply with the AS's policy. BGP import actions are straightforward, and we next describe route selection in detail.

We consider a single destination prefix throughout this paper. A router learns a set of routes to reach the destination from other routers either inside the AS or in neighboring ASes, and then proceeds through a sequence of six *route-selection* steps that compare the routes based on their attributes (Table 1), ultimately selecting a single "best" route for each destination prefix [9]. None of the attributes are directly manipulated by the AS except the local_pref, which is set upon import based on (among other things) the AS's business relationships with its neighbors, traffic-engineering goals,

and security objectives [12].

The route-selection process, and differences in perspective across the routers in an AS, has a significant influence on how much routing information the routers need to share. Thus, we divide the six steps in the selection process into the following three stages (Table 1):

i **All routes are compared in the same way by all routers.** In the first three steps, a router first identifies the routes with the highest local preference; then among routes with the highest local_pref, it identifies those with the shortest AS path; and among those, it identifies the routes with the lowest origin type.

ii **Routes with the same next-hop AS are compared in the same way by all routers.** The fourth step is peculiar because the Multi-Exit Discriminator (MED) attribute is compared only among routes with the *same next-hop AS*. A neighboring AS uses the MED attribute to indicate its preference for where it wants to receive traffic, i.e., at the location(s) announcing the smallest MED value.

iii **Router-specific tie-breaking.** After the first four steps, many routes can still be equally preferred. Step five allows different routers to make different decisions, and each router will direct traffic to the "closest" exit point to implement *hot-potato* routing [13]. If there is still a tie, the router uses router ID as the arbitrary tie-breaker, so that a router selects at most one route.[4]

If a router learns any route to reach the destination, it will select one best route after the six steps.

Route dissemination is important because a router can only select among the routes it has learned, and this is the focus of our paper. The way that routing information travels within an AS can be described by the *signaling graph* [14]. Throughout this paper, by *dissemination* we refer to the propagation of routes on the signaling graph, and assume that *the signaling graph is connected* in the sense that if any node learns any route to a destination, all nodes in the AS eventually learn some route to the destination.

There are multiple ways to have a connected signaling graph. In the simplest case, each border router has an iBGP session with every other router in the AS, i.e., a *full mesh* of iBGP sessions. Unfortunately, a full-mesh configuration does not scale because of the overhead on the routers to maintain the iBGP sessions and store the routes learned from other routers. To address the scaling problem, two mechanisms—route reflectors [15] and confederations [16]—were introduced to allow an AS to disseminate routes in a hierarchical fashion. Route reflectors are more commonly used. A router configured as a route reflector selects a single best route and propagates the route to its clients. A route reflector only propagates an iBGP-learned route to other route reflectors if the route was learned from one of its clients. The signaling graph is connected if the top-level route reflectors are connected in a full mesh and all other routers are their direct or indirect clients [14].

[4]Some router vendors have introduced other tie-breaking options.

2.2 Example Violations of Routing Policies

An AS's routing policy can be violated in several ways. In the following, we give three examples of policy violations. The first example shows that route selection may not converge; the second shows that a router may mistakenly select a route less preferred than another route in the AS-wide preference; the third shows that, among the most preferred routes according to AS-wide preference, a router may not select the route most preferred by its router-specific preference.

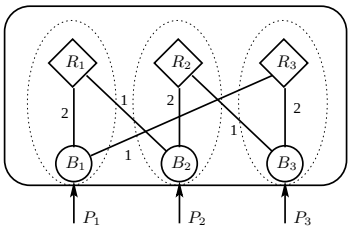

Figure 1: Failure to converge because each router reflector prefers routes not learned by its own client.

In Figure 1 [14], R_i is a route reflector and B_i is a client router of R_i, for $i = 1, 2, 3$. We have omitted the iBGP sessions in the figure but the route reflectors connect in a full mesh and each route reflector connects to its client. Router B_i learns of route P_i externally and each P_i is equally preferred at the AS-wide level (i.e., the P_i have the same local_pref, AS path length, and origin type). The solid lines represent physical links and the number next to a link is the intradomain (Interior Gateway Protocol (IGP)) distance for determining the closest exit point. Thus we see that each route reflector is closer to the client of another route reflector, and R_1 prefers P_2, if R_2 decides to announce it, etc. This configuration will never converge. For example, when R_1 learns of P_2 from R_2 it will choose it and then not disseminate P_1, which in turn causes R_3 to not be able to choose P_1 and settle for P_3. But then R_2 will choose P_3 and no longer disseminate P_2 and so on. Of course, if each route reflector R_i continued to disseminate P_i, despite having selected a different route, then this configuration would converge. Alternatively, the configuration can be changed so that a route reflector is closer to its client than to the other routers.

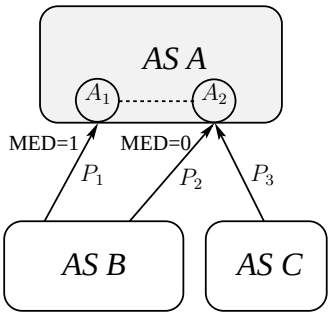

Figure 2: MED semantics can be violated even with full-mesh iBGP. Router A_2 prefers P_3 so router A_1 never learns route P_2, which has the lowest MED value. Router A_1 then selects P_1, violating the MED semantics.

In Figure 2 we have a situation where route selection converges, but the solution violates the semantics of MED. That is, a route with a higher MED value is ultimately chosen even though the AS learns a route from the same AS with a lower MED value. The routes P_1, P_2, and P_3 all have the same local_pref, AS path length, and origin type, but P_2 has a lower MED value than P_1 and they have the same next-hop AS. This reflects the policy goal that AS B wants AS A to use P_2 as a primary route and P_1 as a backup. Assume that router A_2 prefers P_3 over P_2 in the final tie-breaking step, and so P_3 is the only route that A_2 disseminates to A_1. So A_1 does not learn route P_2 and will choose P_1 over P_3 because the exit point of P_1 is closer. Therefore, even though the routers are configured in a full-mesh iBGP, they fail to realize the MED-based policy. If router A_2 were permitted to disseminate the route P_2 despite the fact that it has chosen another route, then the MED-based policy would be realized correctly.

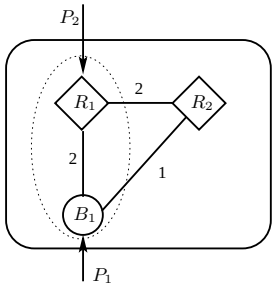

Figure 3: Violation of router-based preference– the hot-potato policy.

In Figure 3 we have two route reflectors R_1 and R_2, and R_1 has a client B_1. Router B_1 learns of route P_1 and R_1 learns of route P_2 externally. The two routes have equal local_pref, AS path length, and origin type, and they are either learned from different ASes or from the same AS with the same MED value. Route selection is then based on the distance to a route's exit point. The distances between routers is indicated by the values on the edges connecting them. In the situation illustrated in the figure, R_1 will choose P_2 and hence will not disseminate P_1 to R_2. Thus, even though R_2 would prefer P_1, which has the closest exit point, R_2 does not learn P_1 and, as such, cannot choose it. Such policy violations can be avoided if more routes are disseminated (e.g., if R_1 disseminates both routes to R_2) or if the route reflector clusters are configured differently (e.g., if B_1 is a client of R_2, the route reflector closer to it).

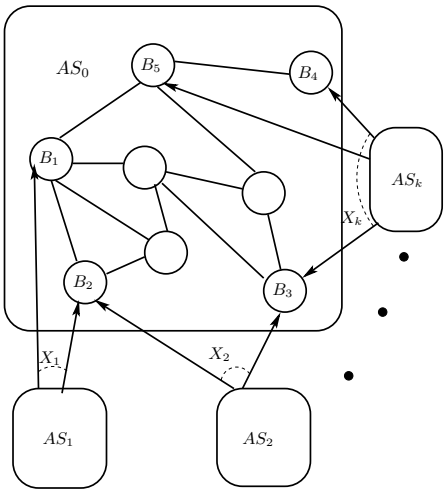

Figure 4: An AS and its neighbors.

3. FORMAL DEFINITIONS

In this section, we formalize the definitions that will allow us to discuss in a mathematical way what it means for a protocol to realize a given policy. The details of the underlying model of the network can be found in the Appendix.

3.1 Routers, Neighbors, and Routes

In Figure 4 we show an AS, namely AS_0, and its neighborhood. The signaling graph of the AS is represented as an undirected connected graph $G = (V, E)$. The circles labeled B_i indicate border routers and the unlabeled ones indicate internal routers, and the undirected lines indicate the edges in E. The AS has neighboring autonomous systems shown as AS_1, AS_2, ..., AS_k. The directed edges from a neighboring AS_i to a border router B_j mean that AS_i announces routes to B_j and hence B_j will be the exit point for such routes.

Let U be the set of all external routes that could possibly be learned. Let AS_i, $1 \leq i \leq k$, be the neighboring ASes of AS_0, and let X_i be the set of external routes learned from AS_i (see Figure 4). If node v learns of a route p directly from AS_i then we say that v is the *exit*

point of p and we write $e(p) = v$. Let $X = \cup_{i=1}^{k} X_i$.

Each node v maintains three sets of routes. The first set is κ_v, the set of all routes in X that have been learned at node v up to the current time. We call the routes in κ_v the *known routes at* v. There is a set $\theta_v \subseteq \kappa_v$ of routes along which packets can be forwarded. The routes in θ_v are called *chosen routes at* v. Finally, there is a set of routes $\sigma_v \subseteq \kappa_v$ containing the routes that v has disseminated to its neighbors inside the AS.

3.2 Protocol Specification

In this section, we discuss the actions that routers in AS_0 can take. In particular, the way a protocol modifies the set of chosen routes and how it chooses which routes to disseminate are the features that differentiate one protocol from another. Thus, specifying a particular routing protocol RP is achieved by defining

- a *selection function* that determines how the set of chosen routes changes when new routes are learned and
- a *dissemination function* that determines what routes are to be disseminated to neighbors within the AS after the selection function has executed.

During each time interval, some subset of the nodes are "activated," where an activated node v applies the selection and dissemination functions to the set κ_v to compute θ_v and σ_v, respectively. In a *fair activation sequence*, each node v is activated infinitely many times.

Note that iBGP essentially fits into our general model if we allow route withdrawals in the messages. A node running iBGP maintains a set of currently known routes plus the node's chosen route from amongst these. When the node receives route update messages from some of its neighbors, iBGP modifies these sets accordingly. In the case of iBGP, if only the set of known routes changes but not the chosen route, then iBGP does not generate any update information for its neighbors.

3.3 Routing Policies

To formalize what we mean by a policy, we first use today's iBGP as an example. In iBGP, some attributes or measures of "goodness" are consistent AS-wide (e.g., local_pref, AS_path_length, and MED). On the other hand, some attributes such as the IGP distance from a node to a route's exit point are not consistent throughout the AS, i.e., in general, nodes will not all have the same IGP distance to the exit point of a given route. More generally, we think of a policy as a combination of an *AS-wide* measure and a *router-specific* measure, and try to design protocols that respect these policies.

More formally, an AS-wide preference P_{AS} determines for each pair of routes, whether or not the routes are comparable and if so, which of the two is more preferred or if they are equally preferred, by all routers in the AS. If route p is *preferred* over route q according to P_{AS} then we write $p \rightarrow_{AS} q$. If they are *equally preferred*, then we write $p =_{AS} q$. Also, for each node v there can be a *router-specific preference* P_v that specifies for each pair of routes whether or not they are comparable and if so, which is more preferred by node v or if they are equally preferred by v. If route p is preferred over route q according to P_v then we write $p \rightarrow_v q$. If the routes are equally preferred by v then we write $p =_v q$.

Define τ_{AS} to be the subset of routes of X where $p \in \tau_{AS}$ if and only if there is no route $q \in X$ where $q \rightarrow_{AS} p$. Similarly, define $\tau_v \subseteq \tau_{AS}$ so that $p \in \tau_v$ if and only if there is no route $q \in \tau_{AS}$ where $q \rightarrow_v p$.

A *policy* then is defined to consist of an AS-wide preference and a router-specific preference for each node v.

3.4 Protocols that Realize a Policy

We are now prepared to define what we mean for a protocol to realize a policy. We define three increasingly tight constraints on the behavior of a protocol that ultimately lead to such a definition.

Convergent protocol: A basic property that we would like a protocol to have is that it eventually halts. For a given network $G = (V, E)$, we say that a routing protocol is *convergent*, if for all possible sets of externally learned routes X and all fair activation sequences α, there exists a time t such that for all time after t, $v \in V$ has received the routes disseminated by its neighbors. In particular, the set of chosen routes at each node remain fixed at every node after time t. Of course, assuming all other aspects being equal, a protocol that is convergent for all possible networks G is better than one that requires conditions on the network for it to be convergent. However, a protocol that is only convergent on a restricted set of graphs but is superior to a more generally convergent protocol in terms of message overhead, computation time, or other issues may be preferred (especially if the constraint on the set of graphs is typically satisfied in reality).

AS-wide preference correct protocol: Let RP be a convergent protocol for G. For a set of external routes X and a fair activation sequence α, consider some time after RP has converged. Suppose τ_{AS} is nonempty. If for all $v \in V$, we have $\theta_v \neq \emptyset$ and $\theta_v \subseteq \tau_{AS}$ then we say that the routes in $\cup_{v \in V} \theta_v$ are *AS-wide preference correct*. If these final chosen routes after convergence are AS-wide preference correct for all X and all α, then RP is said to be an *AS-wide preference correct protocol for* G. If $\tau_{AS} = \emptyset$ then RP is (trivially) AS-wide preference correct if $\theta_v = \emptyset$ for all v.

Fully correct protocol: Suppose RP is an AS-wide preference correct protocol for G. Again suppose τ_{AS} is nonempty. For a given X and α, we say that the resulting set of chosen routes after convergence is *fully*

policy correct if for all $v \in V$, we have $\theta_v \neq \emptyset$ and $\theta_v \subseteq \tau_v$. Basically this means that any route $q \in X$ that router v would prefer over $p \in \theta_v$, according to its own selfish policy, has already been removed from consideration by the AS-wide preference. For a given policy, if the final chosen routes after convergence are fully policy correct for all X and α, then the protocol RP is said to be a *fully correct protocol for* G. Note that in the remainder of the paper, we will often say that a protocol is fully correct without adding that this is for a given policy and/or network when the policy and/or network in question are obvious from context.

In conclusion then, we say that a protocol *realizes* a policy in a network represented by G when for that policy, the protocol is a fully correct protocol for G. Therefore in Section 4, we use the definitions and model described here to study conditions under which a protocol will be fully correct for a given policy.

4. THEORETICAL RESULTS

In this section, we present the main result of the paper—two properties of a protocol that ensure the specified policy is realized correctly; that is, a protocol satisfying these properties will be a fully correct protocol for the given network. We then show that minimizing the number of routes that must be disseminated to satisfy these properties is NP-complete in general. Next, we show that, for common policies that can be expressed via BGP today, the minimum set of routes to disseminate can be computed in polynomial time. Finally, we observe that disseminating fewer routes can unfortunately cause the protocol to converge more slowly.

4.1 Guaranteeing a Fully Correct Protocol

In this section, we present certain properties about route dissemination and selection such that any protocol that satisfies these properties is fully correct. However, we note that the study of route dissemination is only interesting if there is some relationship between the router-specific preference at each node. That is, if there is no relationship between the router-specific preferences at different nodes then essentially the only way to ensure that a protocol is fully correct is to do as follows: a node v should disseminate each route p such that there is no route $q \in \kappa_v$ where $q \rightarrow_{AS} p$. Thus, we explore a fairly loose assumption on router-specific preferences that allows protocols to disseminate fewer routes while remaining fully correct.

In particular, we consider router-specific preferences satisfying the following properties. A useful assumption in [14] to ensure correct behavior was that some shortest path between two nodes would also be a signaling path. We would like a similar but more general property that would be helpful even if not considering shortest paths.

We begin with a definition motivated by an object known as a Voronoi cell in computational geometry [17]. Given a set of points S in the plane, the Voronoi cell of point $v \in S$ is the set of all points p in the plane where v is the closest point in S to p. We borrow this notion to define what we will call a Voronoi set. For all nodes $v \in V$ and every pair of routes p and q we assume $p \rightarrow_v q$, $q \rightarrow_v p$, or $q =_v p$. That is, no two routes are incomparable by any router-specific preference. Also, if $e(p) = v$ then for all routes $q \neq p$, $p \rightarrow_v q$. For a route p and a set of routes S, define the set of nodes $\mathrm{Vor}(p, S)$, called the *Voronoi set of* p *with respect to* S, so that $v \in \mathrm{Vor}(p, S)$ if and only if there is no route $q \in S$ where $q \rightarrow_v p$. Finally, we assume that for every p and S, $\mathrm{Vor}(p, S)$ induces a connected subgraph of G. Router-specific preferences satisfying these conditions will be said to be *consistent*.

As we saw in Figure 3, in order to have a fully correct protocol for which the router-specific component is the hot-potato preference, it may not be sufficient for a router to only disseminate what it considers to be the closest route. However, suppose G is such that for each pair of nodes u and v, some shortest path between u and v in the underlying physical network is also a path in G. Then we say that G *covers the shortest paths*. If G covers the shortest paths, then notice that the common router-specific preference of preferring routes whose exit point is nearer under the IGP metric is consistent. That is, consider the case where $p \rightarrow_v q$ if and only if $d_v(p) < d_v(q)$ where $d_v(p)$ and $d_v(q)$ are the IGP distances from v to the exit point of p and q respectively. Also, $p =_v q$ if and only if $d_v(p) = d_v(q)$. We call this router-specific preference the *hot-potato preference* and it is clearly consistent if G covers the shortest paths. For the hot-potato preference we will say that route $q \in S$ is a *closest route in* S *to* v if $v \in \mathrm{Vor}(q, S)$.

We now define two properties of a protocol that we will show are useful in proving that a protocol is fully correct. Let

$$Q_v = \{r \,:\, r \in \kappa_v,\ \nexists\, q \in \kappa_v \text{ where } q \rightarrow_{AS} r\}.$$

That is, Q_v is the set of all known routes at v for which there are no known routes at v that are more preferred according to the AS-wide preference. Let C_v be the set of routes in Q_v where $p \in C_v$ if and only if $v \in \mathrm{Vor}(p, Q_v)$. That is, C_v is the subset of routes in Q_v that are most preferred according to the router-specific preference at v. The first property is an invariant concerning the set θ_v that a protocol should maintain.

Property 1 (Selection): $\theta_v \subseteq C_v$ and θ_v is nonempty if Q_v (hence C_v) is nonempty.

In other words, v should select known routes that are most preferred according to v's router-specific preference from amongst those known routes that are most

preferred according to the AS-wide preference.

Let S be any set of routes and define $dom(S) \subseteq U$ so that $p \in dom(S)$ if and only if there is some route $r \in S$ where $r \rightarrow_{AS} p$. Then we say that route p is *dominated* by r and $dom(S)$ is the set of routes in U dominated by S. Set $S \subseteq Q$ is said to be *as dominating as* Q if either $Q = \emptyset$ (in which case $S = \emptyset$) or $Q \neq \emptyset$ in which case $S \neq \emptyset$ and $dom(S) = dom(Q)$. Thus if nonempty Q does not dominate any routes, we only consider nonempty subsets S as being as dominating as Q. We also say that p is an *undominated route* if there is no route $q \in \kappa_v$ where $q \rightarrow_{AS} p$.

We call a set T a *covering set* if for some $S \subseteq \kappa_v$ where S is as dominating as κ_v, either

- $T = S$ if $S \cap C_v \neq \emptyset$ or
- $T = S \cup \{p\}$ for some $p \in C_v$ if $S \cap C_v = \emptyset$.

The final property we wish a protocol to have concerns the routes contained in a message that has been sent by an activated node. That is, Property 2 is a condition on the set of routes that are disseminated.

Property 2 (Dissemination): The routes in μ_v are all the routes in some covering set that have not previously been disseminated.

In other words, v should disseminate sufficient routes to ensure that its neighbors do not select routes that are not the most preferred according to the AS-wide preference. Also, v should disseminate routes that are not only most preferred by the AS-wide preference but could conceivably be the most preferred by a neighbor according to its router-specific preference.

We see next that, for a given policy, a protocol that satisfies Properties 1 and 2 is guaranteed to be fully correct for G when the router-specific preference is consistent. Thus given a policy, i.e., an AS-wide preference and a consistent router-specific preference, a protocol's selection and dissemination functions should be defined to satisfy Properties 1 and 2 in order to guarantee that the protocol will be fully correct.

THEOREM 1. *A routing protocol RP satisfying Properties 1 and 2 is fully correct whenever the router-specific preference is consistent.*

PROOF. Let $I_i = [a_i, b_i]$, $b_i < a_{i+1}$, $i = 1, 2, 3, \ldots$ be nonoverlapping intervals of time steps such that during each I_i each node is activated at least once. Then if RP does not converge then during each I_i at least one node must disseminate at least one route. But by the definition of Property 2, each node can only disseminate each route once and so for $t > |V||X|$ there can be no route disseminated during I_t. That is, RP must converge.

We now prove by way of contradiction that RP is AS-wide preference correct. Consider θ_v after convergence. Suppose there is some $p \in \theta_v$ where $p \notin \tau_{AS}$. That is, there is some $q \in X$ where $q \rightarrow_{AS} p$. In particular, choose such a q where $v \in \text{Vor}(q, X)$. Let π be a path from v to the exit point $u = e(q)$ of q. Let w be the node on π closest (in terms of hop count) to u such that κ_w does not contain a route r with $r \rightarrow_{AS} p$. By Property 1, κ_v does not contain a route r with $r \rightarrow_{AS} p$. Thus w is a well defined object. Let x be the neighbor of w on π nearer to u. Since node $q \in \kappa_u$, it must be that $u \neq w$, and so such an x must exist. Then by definition of x, κ_x contains some route r with $r \rightarrow_{AS} p$. Thus by Property 2, x must have disseminated to w some route r' that dominates p. That is, $r' \in \kappa_w$ where $r' \rightarrow_{AS} p$ contradicting the definition of w. Therefore RP is AS-wide preference correct.

Finally, we prove by way of contradiction that RP is fully correct. Suppose RP is not fully correct. Then there must be some node v and some route $p \in \theta_v$ where there is a route $q \in \tau_{AS}$ where $v \notin \text{Vor}(p, \tau_{AS})$ but $v \in \text{Vor}(q, \tau_{AS})$. By Property 1, it must be that there is no such q in κ_v. Since the router-specific preference is consistent, there is a path π in $\text{Vor}(q, \tau_{AS})$ from v to $e(q)$, the exit point of q, in But then Properties 1 and 2 and a simple induction on the hop count from $e(q)$ along π show that every node w along π will learn and disseminate a route r where $r \in \kappa_w \cap \tau_{AS}$ and $r =_w q$. In particular, v will learn of such a route r. But then Property 1 contradicts the assumption that $p \in \theta_v$. □

Notice that Property 2 captures the idea that there are two reasons for disseminating routes. First, a node v should inform its neighbors of one of its routes p from the known set R of routes that are dominated by no other, such that $v \in \text{Vor}(p, R)$ since p is potentially a route that its neighbor might like as a chosen route. This is typically the routing information disseminated in iBGP. Secondly, a node should inform its neighbors of any route it knows about that could dominate another route (i.e., prevent its neighbor from choosing a route that might not meet the AS-wide preference). As has been noticed, iBGP can have trouble converging because routes can be "hidden" from some routers. Disseminating routes that could dominate other routes resolves this problem.

It should also be noted that the above results hold in the case where the router-specific preferences are trivial in the sense that for all v, any two routes p and q are such that $p =_v q$. The results hold since we could set $d_v(p) = 0$ for all v and p. In such a case, potentially one fewer routes would have to be disseminated by an activated node since all routes are "closest".

Notice that a number of different protocols can be defined all of which satisfy Properties 1 and 2 but which differ in the number of routes they disseminate and in the number of chosen routes. For instance, we could define a protocol RP_1 where Property 1 is satisfied by

setting θ_v to be all of the routes in C_v and Property 2 is satisfied by disseminating all routes (not previously disseminated) in C_v and every route in Q_v, that is, the routes in κ_v that are not dominated by any other route in κ_v. This would be a good choice if a goal was to have as many backup routes available at each node v, all of which are in τ_{AS} and for which there are no other routes preferred by the router-specific preference of v. The protocol described in [1] is essentially this protocol (except that each node maintains only a single chosen route). Thus their protocol is fully correct.

4.2 Disseminating a Minimal Set of Routes

In this section we try to minimize the number of routes that must be disseminated to guarantee a fully correct protocol. Unfortunately, we show that in general this will be an NP-complete problem.

The large number of disseminated routes in RP_1 may be considered a practical liability. Consider the following protocol RP_2 that attempts to disseminate fewer routes by modifying the dissemination function of RP_1. This protocol satisfies Property 2 by disseminating the routes not previously disseminated from a smallest cardinality subset $S \subseteq \kappa_v$ such that S is as dominating as κ_v plus a route p where $v \in \text{Vor}(p, D)$ and D (where D is the set of all routes in κ_v that are not dominated by any route in κ_v) if no such route is contained in S. Unfortunately, the following result shows that this is not a good choice for a protocol in general.

LEMMA 1. *Given $k > 0$ and set of routes Q, it is NP-complete to determine if there is a subset $S \subseteq Q$ that is as dominating as Q and $|S| \leq k$. Call this the* MIN DISSEMINATION *problem.*

PROOF. The routes dominated by a given set of routes can be determined in polynomial time so MIN DISSEMINATION is in NP. The NP-hardness of MIN DISSEMINATION can be shown using the following straightforward reduction from SET COVER.

In SET COVER we are given a collection of subsets $S_1, S_2, \ldots, S_n$ and a positive integer k and asked if there is a subcollection I of S_i's such that $\cup_{i \in I} S_i = \cup_{j=1}^{n} S_j = S$ where $|I| \leq k$. Consider such an instance of SET COVER. We now show how to construct an equivalent instance of MIN DISSEMINATION. Suppose $S = \cup_{j=1}^{n} S_j = \{r_1, r_2, \ldots, r_m\}$. For each element r_i in S define a route r_i. For each S_j define a route s_j and define $s_j \rightarrow_{AS} r_i$ if and only if $r_i \in S_j$. Let $U = S \cup \{s_j : 1 \leq j \leq n\}$. Define $Q = \cup_{j=1}^{n}\{s_j\} \bigcup \cup_{i=1}^{m}\{r_i\}$. Then it is clear that $\cup_{i \in I} S_i = S$ if and only if $\cup_{i \in I}\{s_i\}$ is as dominating as Q. □

Thus Lemma 1 says that it is unlikely that there is an efficient procedure for computing a minimum cardinality set $S \subseteq \kappa_v$ so that S is as dominating as κ_v. However, one could implement various heuristic versions of RP_2 where a greedy procedure is used to determine a minimal cardinality subset of κ_v that is as dominating as κ_v. The resulting protocol would still be fully correct but the number of routes disseminated at each activated node may be more than is strictly required.

4.3 The MED AS-wide Preference

While in general it is a hard problem to minimize the number of routes disseminated, there are cases where such minimal sets can be easily computed. An example of an AS-wide preference is the one defined by BGP that says that all routers should prefer route p over q if the local_pref value of p is greater than that of q, or if they are equal and the AS_path_length of p is less than that of q, or if those values are also equal and p and q were both learned from the same neighboring AS_i where AS_i is using MEDs to influence the action of AS and the MED value of p is less than that of q.

Let M denote those neighboring ASes that use the MED attribute and define $X_M \subseteq X$ be the subset of routes that are learned from ASes in M. Similarly, let N denote the neighboring ASes that do not use the MED attribute and define X_N to be the routes learned from ASes in N. Then we can define the MED preference a little more formally as follows. Under the MED preference, $p \rightarrow_{AS} q$ if and only if

- the local_pref of p is greater than that of q or
- p and q have the same local_pref and the AS_path_length of p is less than that of q or
- p and q have the same local_pref and the same AS_path_length, p and q are both learned from the same $AS_i \in M$ and the MED value of p is less than that of q.

Also $p =_{AS} q$ if and only if

- $p, q \in X_N$ and they have the same local_pref and the same AS_path_length or
- $p, q \in X_M$, they are learned from the same $AS_i \in M$ and they have the same local_pref, the same AS_path_length and the same MED values.

When discussing the case where the MED preference is the AS-wide preference we use the notation $\rightarrow_{MED}$ and $=_{MED}$ in place of $\rightarrow_{AS}$ and $=_{AS}$ respectively.

While it is computationally difficult in general to find a smallest subset S of a set Q that is as dominating as Q, in some cases it is straightforward. In particular, we consider the case where the AS-wide preference is the MED preference and we show that finding a minimum cardinality subset $S \subseteq \kappa_v$ so that S is as dominating as κ_v is simple. We begin with the following lemma.

LEMMA 2. *If the AS-wide preference is the MED preference then for any set of routes Q, finding a minimum*

cardinality subset S of Q such that S is as dominating as Q can be done in polynomial time.

PROOF. Without loss of generality we assume that Q is nonempty.

Let lp be the highest local_pref of any route in Q and define $Q' \subseteq Q$ be the subset of routes whose local_pref is lp. Let apl be the smallest AS_path_length of any route in Q' and define $Q'' \subseteq Q'$ to be the set of routes whose AS_path_length is apl. For each $AS_i \in M$ if $X_i \cap Q'' = \emptyset$ define $Q''_i = \emptyset$ but if $X_i \cap Q'' \neq \emptyset$ then let m_i be the lowest MED value of all routes in $X_i \cap Q''$ and define Q''_i to be the set of routes in $X_i \cap Q''$ whose MED value is m_i. Finally, define $Q''_N = X_N \cap Q''$.

Notice that if $AS_i \in M$, then the relationship $\rightarrow_{MED}$ is a total ordering (with ties) over routes that could be learned from AS_i. Similarly, $\rightarrow_{MED}$ is a total ordering (with ties) over the routes that could be learned from any $AS_i \in N$. Thus $R = \cup_{AS_i \in M} Q''_i \cup Q''_N$ is as dominating as Q and so showing that a set S is as dominating as R is sufficient to show that S is as dominating as Q. This total ordering property also says that for any $AS_i \in M$, any two routes in Q''_i dominate the same set of routes and similarly, any two routes in Q''_N dominate the same set of routes.

Define S as follows. Let max be the maximum allowable MED value. If there is some nonempty Q''_i where $m_i < max$ then define S to be any set having exactly one route, say q_i, from each nonempty Q''_i where $m_i < max$. If there is no such Q''_i but $Q''_N \neq \emptyset$, define S to be the set containing any one route, say q_N, from Q''_N. If there is no such Q''_i and $Q''_N = \emptyset$, then define S to be any route in any nonempty Q''_i (which must exist since we are assuming that $Q \neq \emptyset$).

We claim that S is as dominating as R. Suppose $r \rightarrow_{MED} p$ for some $p \in U$ and $r \in R$. If $p \in X_N$ then the local_pref of p must be less than lp or if equal to lp then the AS_path_length of p must be greater than alp. In either case, any route in S will dominate p. Similarly, if p is a route that could be learned from $AS_i \in M$ but $X_i = \emptyset$ or $m_i = max$. The only other case is that p is a route that could be learned from $AS_i \in M$ and $Q''_i \neq \emptyset$ and $m_i < max$. If the local_pref value of p is less than lp or is equal to lp but the AS_path_length of p is more than apl then any route in R (and hence any in S) will dominate p. If the local_pref of p is lp and the AS_path_length of p is apl then it must be that the MED value of p is greater than m_i. But then since S contains some route $q_i \in Q''_i$, q_i will dominate p.

Thus S is as dominating as R and so S is as dominating as Q. Suppose there is some set S' such that S' is as dominating as Q and $|S'| < |S|$. Then it must be that $|S| > 1$ and so $S = \{q_{i_1}, q_{i_2}, \ldots, q_{i_k}\}$ for some k. But that means that $m_{i_j} < max$ and so there are routes in U that are dominated only by routes in Q that are learned from AS_{i_j}. Thus for each AS_{i_j} there must be some route learned from AS_{i_j} in any set that dominates the same routes as Q. That is, S is a minimum cardinality set that dominates the same routes as Q.

To compute such a set S, one could compute one route from each neighboring AS that has local_pref value of lp and AS_path_length of alp. From this small number of routes, it is straightforward how in polynomial time to compute a set S as defined above. □

Thus Lemma 2 says that RP_2 is an easily computable fully policy correct protocol when using the MED preference that in some sense, disseminates a minimum number of routes from each activated node.

If the MED preference is the AS-wide preference and the hot-potato preference is the router-specific preference, then an activated node disseminates at most one route learned from each AS_i using MEDs (plus possibly one closest undominated route) or in some cases it will disseminate a single route from X_N (plus possibly one closest undominated route). This results in a fully-correct protocol that disseminates fewer routes that the protocol proposed in [1] and discussed earlier in Section 4.3.

4.4 Speed of Intra-AS Routing Convergence

RP_2 might seem preferable to RP_1 when using the MED preference since it can often disseminate fewer routes on each activation. However, there are examples where, disseminating fewer routes actually causes RP_2 to take roughly twice as long to converge as RP_1. For example, consider the following instance. In order to compare convergence times, we assume that all nodes are activated at each time step since that would seem to make convergence as fast as possible for any protocol. Let G be the graph with nodes $v_1, v_2, \ldots, v_n$ and edges $e_i = v_i v_{i+1}$, $i = 1, 2, \ldots, n-1$. Let the IGP length of edge e_i be denoted by $L(e_i)$. Suppose $L(e_1) = n-1$ and $L(e_i) = 1$ for $i = 2, 3, \ldots, n-1$. Let $X_{v_1} = \{p\}$ and $X_{v_n} = \{q, r\}$ where p, q and r all have the same local_prefs and AS_path_lengths. Routes p and q both come from $AS_i \in M$ but $r \in X_N$. Suppose the MED value of p is 0 and that of q is 1.

Consider the affects of running RP_1 on this instance. Basically one can imagine at time step i, the route p gets passed from v_i to v_{i+1} while routes q and r get passed in the opposite direction from v_{n-i+1} to v_{n-i} until all three routes "meet" in the middle. Since p has a lower MED value that q, q will no longer be disseminated after they meet in the middle. After $n-1$ steps the protocol will converge with $\theta_{v_i} = \{p, r\}$ for $i = 1, 2, \ldots, n$.

Now consider the behavior of RP_2. The difference is that in this case, v_n will initially only disseminate q since $\{q\}$ is as dominating as $\{q, r\}$. Then it takes $n-1$ for v_n to learn of p at which point it will then begin disseminating r (since it is the closer to v_n than p and q has been "eliminated" by p). Since $d_{v_i}(r) = n - i$ and

$d_{v_i}(p) = n + i - 3$ for $i = 2, 3, \ldots, n$, in $n - 2$ additional time steps, v_2 learns of r and then the protocol halts. Thus, it took $2n - 3$ time steps as opposed to the $n - 1$ time steps it took for RP_2 to converge. Thus, there can be a trade-off between trying to minimize routes disseminated and the convergence time.

5. REALIZING A POLICY IN PRACTICE

Building on our theoretical results, this section addresses practical issues that arise when supporting flexible intra-AS route dissemination in BGP. We first discuss extensions to iBGP for flexible route dissemination, and next expand our treatment of route selection and dissemination to include export policies with neighboring ASes. Then, we discuss how an AS can ensure incoming data packets traverse the chosen BGP route.

5.1 Disseminating Multiple Routes in BGP

Our route-dissemination guidelines at the end of Section 4.3 rely on routers advertising multiple routes to their iBGP neighbors. The "add-path" feature [11] being standardized by the IETF provides precisely this capability—allowing a BGP-speaking router to announce and withdraw multiple routes for the same destination prefix to the same neighbor. The add-path feature is broadly useful, not only for ensuring an AS correctly realizes its routing policy, but also to enable fast convergence to a backup path when a primary path fails.

In addition to the add-path features, the routers must know *which* extra routes to disseminate. For example, our guideline in Section 4.3 calls for identifying the best routes based on the AS-wide preference and then disseminating a route for each next-hop AS that sets the MED attribute. This can be easily supported by extending the router-configuration interface so network operators can specify which ASes have a prior agreement for respecting MEDs. Moving beyond the specifics of today's iBGP, we envision that network operators would configure each router with the AS's routing policy, allowing the router to determine which extra routes (if any) to send to its neighbors. This ensures the routers always realize new policies correctly, while limiting the overhead of disseminating extra routes.

Another important issue is *how many* extra routes are disseminated, since this has a significant impact on scalability. Fortunately, extra routes are disseminated only when a router has multiple "best" routes (in terms of the AS-wide preference) with different next-hop ASes that use MED. In practice, ISPs typically do not accept MEDs from very many neighbors, and do not have many "equally good" routes from different neighbors. For example, an earlier study showed that the AT&T backbone learns routes with shortest AS paths from just *one* neighboring AS for about half of the prefixes, and from *two* next-hop ASes for about 20% of the prefixes [18].

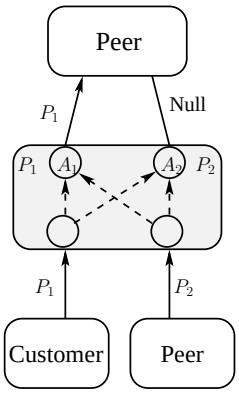

Figure 5: Example of inconsistent export.

Based on their probability distribution of the number of next-hop ASes, we estimate that the AT&T backbone learns routes with the shortest AS paths from an average of two next-hop ASes for each prefix. Even if AT&T respects MEDs for *all* of its neighbors (which is highly unlikely), the routers would only need to disseminate twice the number of routes they do today.

5.2 Satisfying Constraints on Export Policies

Our model in Section 3 focuses on route selection and dissemination within an AS, without regard for whether a node v exports the chosen routes θ_v to neighboring ASes. We can imagine that a router first selects a best route and only then applies an export policy to decide whether to export that route to each external BGP (eBGP) neighbor, as in today's BGP. However, neighboring ASes sometimes impose constraints on the advertised routes that may require an integrated consideration of route selection and route export. For example, when two ISPs peer, their peering contracts often require that they export "full customer routes" (i.e., a route for each destination prefix learned from one of their customers) so that they can reach each other's customers. If the ISPs peer at multiple locations, the contracts often require that the routes for the same destination prefix are *consistent* (i.e., have the same AS-path length) [19, 2] across the many peering points. This ensures that all learned routes are equally prefered at the AS-wide level, allowing each routers in each AS to freely direct traffic via "closest" exit point based on IGP distances.

These kinds of requirements are easily violated by the way route selection interacts with route export [19, 2], as shown by the example in Figure 5. Suppose the (shaded) AS prefers routes with the shortest AS path (i.e., by assigning the same local_pref to all routes). If the AS learns a route P_1 (from a customer) and P_2 (from a peer) that are equally good in the first four steps of the route-selection process, each router selects the

route with the closest exit point, causing A_1 to select P_1 and A_2 to select P_2. Now, an AS typically only exports customer-learned routes to its peers, to avoid providing transit service between its competitors. So, even though routers A_1 and A_2 are configured with the same route-selection and route-export policy, A_1 will export a route (namely, P_1) but A_2 will not (since P_2 is a peer-learned route). As such, the AS inadvertently violates the "consistent export" requirement.

These kinds of constraints on export policy can be handled by having a compound policy with a different AS-wide preference for each class of neighbors—one for customers and another for peers and providers. The AS-wide preference for customers would prefer routes with the shortest AS paths (i.e., assigning the same local preference to all routes), whereas the AS-wide preference for peers and providers would first prefer customer-learned routes before breaking ties based on AS path length. Each router would choose the routes to disseminate for each class, leading to the occasional dissemination of extra routes when the two classes have different requirements (as in Figure 5). Each router would then select a route for each eBGP neighbor, based on its class—essentially treating route export as "per edge link" (rather than per router) route selection. As with our intra-AS dissemination guidelines in Section 4.3, we believe the routers would only need to disseminate extra routes in unusual circumstances—precisely when the AS would otherwise violate its business contracts.

5.3 Ensuring Traffic Follows a Chosen Route

Our treatment of intra-AS route dissemination has focused, understandably, on the "control plane" protocol that selects and propagates routing information. Ultimately, the control plane only exists so the routers can decide how to forward packets in the data plane. As such, we must ensure that a packet entering the AS follows a route chosen by the ingress router (i.e., a route in θ_v if the packet enters the AS at router v). However, if a packet traverses multiple hops within the AS on its way to the chosen exit point, a router along the way might have chosen a different route, and inadvertently "deflect" data packets toward a different exit point [14]. Packet deflections can, in the extreme, cause persistent forwarding loops, where the routers continually forward the packet back and forth amongst themselves. These kinds of problems may be even more likely under our routing guidelines, since routers disseminate extra routes they themselves have not chosen to use.

Fortunately, this problem has a simple solution that is widely used in practice—not allowing intermediate routers to select exit points for packets. This can be achieved by existing packet encapsulation techniques like MultiProtocol Label Switching (MPLS) and IP-in-IP encapsulation. The simplest approach is router-to-router encapsulation, where the first router encapsulates an incoming packet with the identity of the chosen exit router. This prevents any intermediate routers from deflecting traffic to an alternate exit point, since these routers forward the packets based on the outer encapsulation header. However, this approach is not sufficient if a router may select a different best route for packets arriving via different ingress *links*, as in Section 5.2. To handle more general scenarios, the router could encapsulate incoming packets with the identity of the appropriate exit *link*. This would allow the routers to forward different incoming packets to different exit links that lead to different interdomain paths.

6. RELATED WORK

Most work on interdomain routing has focused on protocol convergence, giving conditions for inter-AS stability and to avoid intra-AS routing anomalies [5, 20, 21, 2, 14, 10]. Although we draw examples from previous work, our focus is not on what BGP does today, but on what an intra-AS routing protocol *should* do—guarantee that an AS correctly realizes its policy regardless of the topology. Applying our analysis to today's BGP-speaking networks, we suggest enhancements to today's protocols to eliminate policy violations.

The most related work to ours is Basu et. al. [1]. Interestingly, even though they started from modeling the current iBGP protocol, and we started from examining the problem iBGP should be solving, our solutions are similar: they propose to disseminate all best routes according to the AS-wide preference (as discussed at the end of Section 4.1), while we point out that only a subset of those routes need to be disseminated (as discussed in Section 4.3). Musunuri and Cobb [7] also propose a solution relying on dissemination of multiple routes. They suggest having route reflectors disseminate to one another the set of routes as described by [1]. However, each route reflector is assumed to have knowledge of the shortest path trees from each of its clients and hence the route reflector can use this knowledge to disseminate a single customized route to each of its clients. RFC 3345 [6] also discusses dissemination of multiple routes (essentially) one per neighboring AS. The proposals of both [7] and [6] are designed to prevent oscillations (although an example where [6] fails to achieve this is given in [1]) but neither is guaranteed to realize policy.

Metarouting [22, 23] shares our view that routers in an AS should select routes so that they collectively provide some policy guarantee. The two pieces of work are complementary as metarouting provides a formal way to *specify policies* by decomposing them into routing algebras, and gives sufficient conditions on policies to ensure convergence, while our work *provides the mechanism* that guarantees an AS realizes its policy.

As mentioned, we do not consider the issue of a higher level language to specify policies. The Routing Policy Specification Language (RPSL) [24] simplifies the specification and generation of per-router configurations; similarly, Boehm et. al. [25] created a system that translates a given policy objective, described in a high-level language, into vendor-specific router configurations. As we have shown, due to the interactions between policy objectives and dissemination protocols today, configuring all routers correctly does *not* ensure that the routers *collectively* realize the AS's policy. But our proposed changes to current protocols, combined with the automatic configuration tools, will guarantee that an AS realizes its policy—an ideal situation for operators.

7. CONCLUSION AND FUTURE WORK

Despite the importance of policy-based interdomain routing, an AS is not guaranteed to realize its policy today. In this paper, we present a formalism that allows us to precisely state what we mean by a policy and what it means for a protocol to realize that policy. We provide conditions on a distributed protocol that provably guarantees that it will correctly realize an AS's policy.

While we have provided a formal foundation for studying the correctness of a protocol in terms of the policy it is meant to realize, a number of important open issues remain. For instance, determining a covering set of routes given just a high-level policy description rather than from a low-level preference relationship as we have discussed may pose interesting challenges. Also, a more complete study of the trade-off between the number of routes disseminated versus the resulting speed of convergence would be useful and challenging.

APPENDIX

Underlying Operational Model

This appendix establishes the fundamental basis on which we will study routing protocols, expanding on the high-level overview presented earlier in Section 3.1. This includes a notion of discrete time progression, message passing assumptions, and a definition of the types of operations that a generic routing protocol will perform.

We begin by reviewing the model of an AS. The signaling graph of the AS under consideration is represented as an undirected connected graph $G = (V, E)$. The nodes represent routers and an edge represents a protocol communication link; that is, if $uv \in E$ then the protocol running at u can send information from u to v and vice versa for the protocol running at v. Thus the edges do not necessarily represent physical links along which packets would actually flow. For $v \in V$, a *neighbor* of v is a node u such that $uv \in E$.

A *protocol* is a procedure running at each node in G that proceeds in discrete time steps. During a particular time step, a node will consider its state and any new information available, modify its state, and if its state actually changes due to this operation, send messages to its neighbors alerting them of the change. As defined in [26], a *fair activation sequence* is a sequence $\alpha = V_1, V_2, \ldots$ of subsets of V such that every node is in infinitely many V_i. The idea is that if a node is in V_i then at step i, node v will run its instance of the protocol, which begins by updating its state based on the information it has at the start of the time step including any new information from its neighbors.

At the end of a time step, an activated node can send messages to its neighbors where a message is a set of routes. We say that the routes in such a message have been *disseminated*. While it could be that a node disseminates a different set of routes to each of its neighbors, for simplicity, in the protocols we consider here, a node disseminates the same routes to all neighbors. Thus for clarity and ease of notation we assume that all neighbors are sent the same message. This assumption is not essential and the model can easily be modified for the more general case. When a message is sent to a node v, it will arrive at the end of the queue b_v at v by the start of some later time step. If multiple messages arrive at b_v at the same time they enter the queue in arbitrary order. Let h_v be the message at the head (i.e., the front) of the queue b_v if $b_v \neq \emptyset$ and otherwise $h_v = \emptyset$. When a node v reads in a message, we say that v *learns* the routes in the message.

As in [1], [14] and others, we wish to study the behavior of the routers within the AS assuming that the routes available from neighboring ASes remain fixed. For ease of presentation, our analysis will assume that the routers within the AS begin knowing no routes. These assumptions imply that there is no need for a mechanism to withdraw routes that are no longer offered by neighboring ASes. Extending our model to allow for withdrawals is relatively straightforward but would just complicate the description unnecessarily.

Let U be the set of all possible external routes that could possibly be learned. Let AS_i, $1 \leq i \leq k$, be the neighboring ASes of AS, and let X_i be the set of external routes learned from AS_i (see Figure 4). If node v learns of a route p directly from AS_i then we say that v is the *exit point* of p and we write $e(p) = v$. Let $X = \cup_{i=1}^{k} X^i$. Since we are considering the case where the routes learned from external ASes are fixed, we will assume that for every node $v \in V$ there is a message in the queue b_v at the beginning of time step $t = 1$ consisting of the set of all routes for which v is the exit point. We assume that all nodes are activated at the first time step, i.e., $V_1 = V$. (These assumptions are just to get the system initiated in a manner that is simple to describe, however they are not essential.)

Each node v maintains three sets of routes. The first

set is κ_v, the set of all routes in X that have been learned at node v up to the current time. We call the routes in κ_v the *known routes at* v. There is a set $\theta_v \subseteq \kappa_v$ of routes along which packets can be forwarded. The routes in θ_v are called *chosen routes at* v. Finally, there is a set of routes $\sigma_v \subseteq \kappa_v$ containing the routes that v has disseminated to its neighbors.

Suppose v is activated in a time step and u is a neighbor of v. Then a message μ_v can be sent to the buffer b_u at the end of the time step. We assume that if the set κ_v remains unchanged when v is activated (i.e., v learns of no new routes) then no message is sent to v's neighbors. If activated in a time step, a node first reads in the routes in the message at the head of its buffer,[5] modifies its set of known routes and its set of chosen routes, sends messages to its neighbors, and modifies its set of disseminated routes. For all protocols we discuss, when a node v is activated it will modify the set of known routes to include those in h_v and it will modify the set of disseminated routes to include those in μ_v.

A. REFERENCES

[1] A. Basu, C.-H. L. Ong, A. Rasala, F. B. Shepherd, and G. Wilfong, "Route oscillations in I-BGP with route reflection," in *Proc. ACM SIGCOMM*, vol. 32, pp. 235–247, 2002.

[2] N. Feamster, Z. M. Mao, and J. Rexford, "BorderGuard: Detecting cold potatoes from peers," in *Proc. Internet Measurement Conference*, pp. 213–218, 2004.

[3] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness, and stability," in *Journal of the Operational Research Society*, vol. 49, 1998.

[4] J. Moy, "OSPF Version 2." RFC 2328, April 1998.

[5] T. Griffin, F. Shepherd, and G. Wilfong, "The stable paths problem and interdomain routing," *IEEE/ACM Trans. on Networking*, vol. 10, pp. 232–243, April 2002.

[6] D. McPherson, V. Gill, D. Walton, and A. Retana, "BGP Persistent Route Oscillation Solution." RFC 3345, August 2002.

[7] R. Musunuri and J. Cobb, "A complete solution for iBGP stability," in *Proc. International Conference on Communications*, 2004.

[8] Y. Wang, I. Avramopoulos, and J. Rexford, "Design for configurability: Rethinking interdomain routing policies from the ground up." to appear in *IEEE Journal on Selected Areas in Communications*, 2009.

[9] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)." RFC 4271, January 2006.

[10] T. Griffin and G. T. Wilfong, "Analysis of the MED Oscillation Problem in BGP," in *Proc. International Conference on Network Protocols*, pp. 90–99, 2002.

[11] D. Walton, A. Retana, E. Chen, and J. Scudder, "Advertisement of Multiple Paths in BGP." Internet Draft, July 2008. `http://tools.ietf.org/html/draft-walton-bgp-add-paths-06`.

[12] M. Caesar and J. Rexford, "BGP routing policies in ISP networks," *IEEE Network*, vol. 19, no. 6, pp. 5–11, Nov.-Dec. 2005.

[13] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford, "Dynamics of hot-potato routing in IP networks," *IEEE/ACM Trans. on Networking*, vol. 16, December 2008.

[14] T. G. Griffin and G. Wilfong, "On the correctness of IBGP configuration," in *Proc. ACM SIGCOMM*, pp. 17–29, 2002.

[15] T. Bates, R. Chandra, and E. Chen, "BGP Route Reflection - An Alternative to Full Mesh Internal BGP (IBGP)." RFC 4456, April 2006.

[16] P. Traina, D. McPherson, and J. G. Scudder, "Autonomous System Confederations for BGP." RFC 5065, August 2007.

[17] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*. Berlin: Springer-Verlag, 1987.

[18] N. Feamster, J. Borkenhagen, and J. Rexford, "Guidelines for interdomain traffic engineering," *ACM SIGCOMM Computer Communications Review*, October 2003.

[19] A. Flavel, A. Shaikh, T. Scholl, and M. Roughan, "Peer Dragnet: Analysis of BGP Peering Policies," *U. Adelaide Technical Report*, 2008.

[20] L. Gao and J. Rexford, "Stable Internet routing without global coordination," *IEEE/ACM Trans. on Networking*, vol. 9, no. 6, pp. 681–692, 2001.

[21] N. Feamster and H. Balakrishnan, "Correctness properties for Internet routing," in *Annual Allerton Conference on Communication, Control, and Computing*, September 2005.

[22] T. G. Griffin and J. L. Sobrinho, "Metarouting," in *Proc. ACM SIGCOMM*, pp. 1–12, 2005.

[23] A. J. T. Gurney and T. G. Griffin, "Rethinking BGP Metrics," *unpublished report*, March 2008.

[24] C. Alaettinoglu, T. Bates, E. Gerich, D. Karrenberg, D. Meyer, M. Terpstra, and C. Villamizar, "Routing Policy Specification Language (RPSL)." RFC 2622, June 1999.

[25] H. Boehm, A. Feldmann, O. Maennel, C. Reiser, and R. Volk, "AS-Wide Inter-Domain Routing Policies: Design and Realization," *NANOG 34 Presentation*, May 2005.

[26] T. Griffin and G. Wilfong, "A Safe Path Vector Protocol," in *Proc. IEEE INFOCOM*, (Tel Aviv, Israel), pp. 490–499, March 2000.

[5] A node can be activated repeatedly to empty its message buffer, if desired.