Making an AS Act Like a Single Node: Toward Intrinsically Correct Interdomain Routing

Rui Zhang-Shen, Yi Wang, and Jennifer Rexford
Princeton University

ABSTRACT

Although interconnecting independently-administered networks is one of the main purposes of the Internet, the roles and responsibilities of an Autonomous System (AS) remain vaguely defined. An AS is often viewed as an *atomic* entity that behaves like a single node with a clearly-defined routing policy, although these notions are also ill-defined. In this paper, we describe precisely what it means for an AS to be atomic by introducing three necessary conditions that capture an AS's responsibilities to its neighbors while achieving its own policy goals. For example, an AS should export "consistent" routes at all connections to the same peer. We show that BGP violates the necessary conditions, making it difficult for an AS to adhere to its routing policy without resorting to unwieldy constraints on the network configuration. We then propose modest enhancements to how BGP routes are selected, disseminated, and used within an AS to make the network "intrinsically atomic." These changes are easily implementable in practice, introduce minimal overhead, and, since they are local to an AS, can be deployed incrementally. We show that making ASes intrinsically atomic enables the designers of interdomain routing protocols to ignore intra-AS details *without sacrificing correctness*.

1. INTRODUCTION

The separation of intradomain and interdomain routing is one of the founding principles of the Internet [1]. This decoupling is important because it enables the *distributed management of the Internet* by separate administrative entities in a scalable fashion. Within their own networks, the operators are free to select any routing protocols and define any local policy objectives. They can hide sensitive information, such as their network topology and business relationships, from their competitors. However, despite the importance of the two-tiered routing architecture, the roles and responsibilities of each Autonomous System (AS) are not well defined. In addition, each AS is itself a distributed network, raising the difficult challenge of making a collection of routers behave as a single entity.

1.1 Case for "Atomic" Autonomous Systems

An AS is often loosely defined as a collection of routers and links managed by a single administrative entity. Historically, an AS is defined as "a connected group of one or more IP prefixes run by one or more network operators which has a SINGLE and CLEARLY DEFINED routing policy" [2]. As such, the many routers in an AS should, collectively, *act like a single node with a clearly defined way of selecting and exporting routes*. To differentiate from the common usage of "AS," we refer to a network that "acts like a single node" as an *atomic AS*. However, over the years, the BGP protocol and operational practices have deviated far from this goal, for legitimate reasons such as improved flexibility and scalability. Ensuring that ASes adhere to a clear notion of atomicity, while remaining flexible and scalable, would enable both practitioners and researchers to ignore intra-AS details when configuring routers, analyzing measurement data, and designing protocols, *without sacrificing correctness*.

Suppose for a moment that an AS consists of a single BGP-speaking router. That router would select a single "best" route (for each destination prefix) from the set of routes learned from neighboring ASes, and export that route (or not) to each neighbor. Both the choice of the best route and the decision to export it are dictated by the AS's policy. The landscape changes in several interesting ways when we move to a more complex setting with multiple routers:

- In a distributed setting, it is not reasonable to require every router to select the same route. For example, an AS may want to employ "hot potato" routing to have traffic exit its network as early possible.
- The routers may not be able to learn every externally-learned route, due to scalability concerns.
- Multiple routers may connect to the same neighboring AS, begging the question of what expectations the neighbor should have (if any) about the routes exported at different locations.
- Data packets may traverse multiple routers in the AS, and some of these routers may have selected different "best" routes.

These challenges make it difficult for a distributed collection of routers to realize even some of the most basic policies

correctly, even though BGP gives network operators considerable flexibility in specifying how *individual routers* select and export routes.

For example, peering agreements typically require an AS to export "consistent routes" (e.g., with the same AS-path length) across different connections to the same peer [3, 4, 5]. This prevents the AS from biasing how the peer chooses to direct traffic out of its own network. However, an AS may easily violate consistent export *despite configuring each router to select and export routes the same way*. Consider an AS that always selects routes with the shortest AS paths, and only exports routes learned from customers to its peers. Because of hot-potato routing, some routers in the AS may select a customer-learned route while other routers select a peer-learned route. If there is a peer that connects to the AS in several locations, some routers will export the (customer-learned) route, and others will decline to export the (peer-learned) route, causing the peer to receive a route in some locations but not in others (see Figure 2 in Section 3), violating the consistent export requirement [6].

As the example illustrates, the ASes in today's routing system are *not* necessarily atomic. Atomicity is easily violated because of the way BGP routes are selected, disseminated, and exported by the routers in an AS. Within large ASes, BGP routing is quite fragile. Different policy objectives can have subtle interactions that lead to policy violations and even routing oscillations. In addition, techniques for making route dissemination more scalable can lead to oscillation, blackholes, and forwarding loops. Network operators can sometimes prevent these problems by avoiding problematic network configurations, but reasoning about the "correctness" of a given configuration is surprisingly difficult. In addition, network operators are understandably loathe to sacrifice important goals like flexibility and scalability, even to ensure correctness.

Making ASes "intrinsically atomic" would offer significant practical benefits, such as preventing subtle violations of routing policies and simplifying the task of configuring the routers. In addition, researchers and practitioners alike could ignore intra-AS details when modeling BGP, analyzing BGP measurements, designing protocol extensions, and proposing new interdomain routing architectures. In fact, many researchers already model the Internet topology at the AS level when designing and analyzing interdomain routing, but this assumption is unfortunately inaccurate today. We believe that protocol designers *should* be able to ignore intra-AS details, and we show how the distributed collection of routers within an AS can support that abstraction.

1.2 Making ASes Behave Atomically

In this paper, we formally define what it means for an AS to be *atomic*. We argue that the model of an atomic AS should include the multiple edge links between ASes, but not internal details like whether two links terminate at the same border router. Ignoring the network's internal structure, an AS imports at most one route for a given destination prefix at each edge link; in turn, the AS assigns a (possibly null) route to each edge link and exports this route to the incident neighbor. We consider the AS to be atomic if three conditions hold, stated informally as follows:

1. A router should never select (or export) a route that is "worse" than some other route that the AS has learned, where the notion of "better" or "worse" depends on the AS-wide policy objectives.
2. While selecting routes to achieve its internal objectives, an AS should export "consistent" routes at all edge links connecting to the same neighboring AS, based on the neighbor's expectation (if any) of how similar the routes should be.
3. An AS should forward data packets (arriving on an edge link) along the route it announces in the opposite direction on the same link.

In Section 4, we state these three conditions more precisely, based on an AS-wide ranking of routes. We also show how a distributed collection of routers can select, disseminate, and export routes without violating atomicity.

In Section 5, we explore how these results apply specifically to BGP. First, we consider how to make an AS behave atomically without any changes to the way today's routers operate. We prove that network operators can make their ASes atomic today by adhering to a set of configuration guidelines. Despite being useful in practice, these configuration guidelines impose undesirable constraints on how network operators run their networks. Operators might reasonably choose to violate the guidelines in the interest of greater flexibility and efficiency, or lower cost. As such, we explore protocol extensions that make a BGP-speaking AS *intrinsically atomic*. At a high level:

- The first condition can be satisfied by increasing *route visibility* by disseminating extra routing information within the AS.
- The second condition can be satisfied by giving a border router the flexibility to select (and export) a different best route for each edge link.
- The third condition can be satisfied through a simple tunneling mechanism that directs data packets from the ingress link to the intended egress link.

We argue that these mechanisms can be easily supported, with just a modest increase in overhead. In fact, some routers already have similar features today for a different reason—to enable ISPs to offer Virtual Private Network (VPN) services. Since these protocol enhancements only affect the selection, dissemination, and export of routes within a single AS, the changes can be deployed incrementally.

Finally, Section 6 discusses related work and Section 7 concludes the paper.

Stage	BGP route selection step	Routes compared	Result applies to
i	1. Highest local_pref 2. Lowest AS path length 3. Lowest origin type	all routes	entire AS
ii	4. Lowest MED value	routes from the same neighbor	entire AS
iii	5. eBGP-learned over iBGP-learned 6. Lowest IGP path cost 7. Lowest Router ID	all routes	single router

Table 1: The seven-step BGP route-selection process consists of three main stages.

2. BGP ROUTING WITHIN AN AS

The routers inside an AS disseminate the externally-learned routes and select routes to collectively realize the AS's policy goals. When an AS has a simple routing policy (e.g., based only on AS-path length) and no scalability concerns (e.g., a full-mesh iBGP configuration), atomicity is easy to ensure. Over the years, BGP has become more complex in order to enhance flexibility and scalability. More route attributes were added to BGP announcement messages, and more steps to the BGP route-selection process, to give operators more flexibility in expressing policies. Mechanisms (such as route reflectors and confederations) were introduced to alleviate the scaling problems of an iBGP full mesh. The added complexity makes it extremely difficult to ensure atomicity. In this background section, we describe how route selection and dissemination work today.

2.1 Flexible Route Selection

A BGP-speaking router learns one or more routes for a destination prefix from neighboring ASes and other routers in the local AS. The router may apply an import policy to filter the externally learned routes or modify their attributes, with the goal of influencing the route-selection process. The router then proceeds through a sequence of steps that compares the routes based on their attributes, ultimately selecting a single "best" route for each destination prefix. In practice, the router receives and sends BGP update messages over time, leading to changes in the choice of the best route. If starting at some time there are no more update messages, then once the system converges, the router has a stable set of candidate routes and has selected the best route from that set. Since we are primarily interested in defining and analyzing "correctness" in steady state, we do not consider the transient behavior and instead focus only on how the routers select the best route from a set of candidate routes.

The BGP route-selection process consists of seven main steps [7], as shown in Table 1. In the first three steps, the router identifies the routes with the highest local preference, breaking ties based on AS-path length and then origin type. The fourth step is peculiar because the Multi-Exit Discriminator (MED) attribute is compared *only* amongst routes learned from the same neighboring 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) that announce routes with the smallest MED value). The MED-comparison step makes it impossible to define a total ordering on the routes, because the relative ranking of two routes may depend on the presence or absence of a third route. Despite the appealing simplicity of having a total ordering, the MED attribute was added to BGP to address an important policy goal, and as such a useful model of atomicity must not assume that the route-selection process has a total ordering of routes.

In the first four steps all routers in the AS would select the same best routes when presented with the same set of routes. The final three steps allow different routers to make different decisions. Steps five and six, in particular, allow each router to direct traffic to the "closest" egress point—realizing *hot-potato* routing. For example, suppose an AS connects to a neighbor in two locations, e.g., Seattle and Boston. Hot-potato routing allows the traffic entering the network in the west coast to leave via Seattle, and the traffic entering in the east coast to leave via Boston. Proximity is represented in terms of the cost of the intradomain path from the ingress router to the egress router, as computed by an Interior Gateway Protocol (IGP) like OSPF or IS-IS. If multiple routes have the same IGP path cost, the router selects the route learned from the router with the smaller router-id in a final tie-breaking step, so that a router selects at most one route. Similar to the addition of the MED attribute, hot-potato routing realizes an important policy goal and, as such, any model of atomicity should allow different routers in the same AS to select a different best route.

In summary, the BGP route-selection process consists of three main stages: (i) AS-wide path pruning, where all routes are compared and all routers would make a decision in the same way, (ii) the MED comparison that only compares routes learned from the same neighboring AS, and (iii) the router-specific tie-breaking steps where different routers may make different decisions.

2.2 Scalable Route Dissemination

Upon selecting a best route, a router uses internal BGP (iBGP) to inform other routers in the AS about the route. 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

- Disallow the use of "prepend" or "no export" communities.
- Strictly prefer customer-learned over peer-learned routes.
- Do export filtering based only on business relationship.
- Terminate each external link with the MED attribute on a separate router.
- Use full-mesh iBGP.
- Use hot-potato routing.

Table 2: Solutions to solve BGP's violations of atomicity.

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 [8] and confederations [9]—were introduced to allow an AS to disseminate routes in a hierarchical fashion. Since route reflectors are more commonly used, we briefly explain them here. A router configured as a route reflector selects a single best route and propagates the route to its clients. Yet, a route reflector only propagates an iBGP-learned route to other route reflectors if the route was learned from one of its clients.

Using route reflectors reduces the number of iBGP sessions, as well as the number of routes the clients need to receive and store. Because each route reflector propagates only its best route to its iBGP neighbors, the routes a router learns depend on the routes selected at other routers. That is, the use of route reflectors reduces the *visibility* of the routes. In fact, in some iBGP configurations, a router may not learn *any* route. A route learned at one router propagates to another router over a sequence of iBGP sessions, also called a *signaling path*. A *connected signaling graph*, then, is an iBGP configuration that ensures that every border router has a signaling path to every other router in the AS. A connected signaling graph guarantees that, as long as any router in the AS learns a valid route to reach a destination prefix, all the routers in the AS would learn at least one route.

In summary, the iBGP route-dissemination process allows routers to learn about routes first learned by other routers (via external BGP). However, existing techniques for scalable route dissemination reduce the visibility the routers have into the available routes. Since scalability is crucial for large ASes, a practical definition of atomicity should not require every router to learn every route.

2.3 Flexible Route Export

After selecting a single best route, a border router must decide whether to export the route (perhaps after some modification) to each eBGP neighbor. In the simplest case, the border router simply exports the unmodified route to every eBGP neighbor. However, over the years, export policies have become more complicated to support important policy goals. For example, export policies often reflect the business relationship between a pair of ASes. An AS typically classifies a route based on its relationship with the neighboring AS who announced the route. Common business relationships include customer-provider (where the customer pays the provider for transit service to reach the rest of the Internet) and peer-peer (where two ASes agree to carry traffic between their respective customers) [10]. A router typically exports any best route to its customers, but does not export routes learned from one peer or provider to another.

A provider may also give its customers some control over how it selects and exports routes. In particular, the BGP community attribute [11] is often used to give customers "remote control" over the handling of a route. For example, a customer may tag the route with a "no-export" community to instruct the provider not to export the route to other ASes. As another example, a customer may tag the route with a community that instructs the provide to *prepend* the AS-path attribute with additional hops as part of exporting the route. By making the AS path artificially longer, AS prepending reduces the likelihood that other ASes would select the route, allowing the customer to reduce the volume of incoming traffic.

In summary, BGP offers significant flexibility in how a router manipulates and exports the best route, as these examples illustrate. In fact, commercial routers allow network operators to filter and modify routes based on regular expressions on the AS-path attribute as well. Network operators make use of these features to achieve their policy goals. As such, any practical definition of atomicity must permit the flexible expression of export rules.

3. BGP VIOLATES ATOMICITY

In this section we show by example three types of problems that the current BGP protocol has, each due to the measures to make BGP distributed, flexible, and scalable, respectively. We draw examples from previous work, where the problems have been studied in isolation from one another and mainly in the context of protocol convergence, and even-

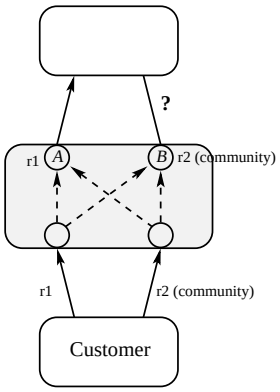

Figure 1: Communities specifying "no export" or "export prepending" can cause inconsistent export. Solid lines are eBGP sessions and dashed lines are iBGP sessions. Arrows represent route propagation. The two routes r_1 and r_2 have the same AS path length and origin type, and are selected by the two routers based on hot-potato routing. If only one of the two routes r_1 and r_2 has the community attribute then they will be exported differently.

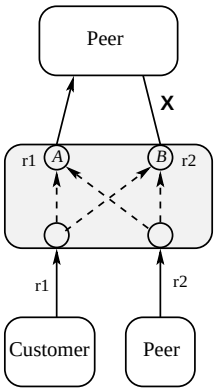

Figure 2: Equally preferring customer-learned routes and peer-learned routes can cause inconsistent export. The two routes r_1 and r_2 have equal AS path length. Routers choose routes based on hot-potato routing. Only r_1 is exported because peer-learned routes are not exported to peers while customer-learned routes are.

tually deduce from them the principles to guarantee atomicity. We provide solutions to these problems that are feasible today, i.e., by putting constraints on network and routing configurations but without changing the protocol itself. These solutions are summarized in Table 2.

3.1 Different Routers, Different Decisions

When two networks *peer*, peering contracts often require them to export consistently—exporting routes with the same AS path lengths—at all peering points so as not to influence the routing decisions of the other network [3, 4, 5]).[1] In this section we present two examples of inconsistent export, the first due to the community attribute which is commonly used by customers to dictate the provider's export actions, and the second due to the export filtering based on business-relationships. Similar examples have been presented in [6].

In Figure 1 the AS learns two routes r_1 and r_2 from the customer, which have the same AS path length and origin type, and r_2 has a community attribute specifying an export action, such as "no export" or "AS path prepending" [11]. The router that chooses r_1 will export it to a neighbor but the router that chooses r_2 will not export it if r_2 has the "no export" community or will export a prepended version (which appears to have a longer AS path length) if r_2 has the "prepending" community. In either case, the AS exports differently on the two links to the same neighbor.

By using the community attributes the customer expresses its preference that traffic from the provider can use all the routes but traffic from the neighbors of the provider can use only the routes without communities. The correct action in Figure 1 would be for the provider to export r_1 at both links. This is not possible with today's BGP protocol because a router can choose only one route. One way to solve the problem is to disallow the use of communities which modify or filter routes on export. This is probably not acceptable because network operators need them in order to express their (or their customers') policy objectives. An alternative solution is to take care of communities upon import by setting local_pref values so that routes with no-export or prepend communities are less preferred than those without.

In Figure 2 the AS learns two routes of equal AS path length: r_1 from a customer and r_2 from a peer. The AS bases its path selection decisions first on AS path length, which can be achieved by assigning the same local preference to all routes. For example, it would like to use shorter AS paths and to split traffic over more downstream paths [10]. So both r_1 and r_2 are selected by the routers. Yes, an AS does not want to export peer-learned routes to peers, so as not to act as transit for the two peers. So the peer-learned route will not be exported to the peer while the customer-learned route will, causing inconsistent export.

Since not exporting peer-learned routes to peers is an important policy that almost every AS implements, the level i) solution is for every AS to prefer customer-learned routes strictly over peer-learned routes. This way in Figure 2 only r_1 will be selected by the routers and exported. If r_1 fails, r_2 will be selected but will not be exported. This solution may affect network performance if the customer has only very limited resources.

3.2 Absence of a Total Ordering

The Multi-Exit Discriminator (MED) attribute, despite its

[1] Even when consist export is not required, there may be other expectations of how routes are exported to a neighbor.

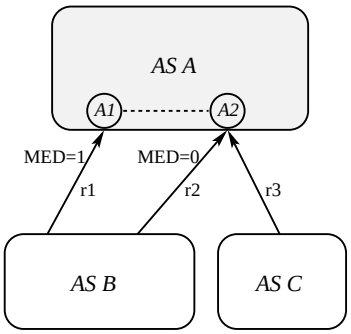

Figure 3: An example where MED semantics are violated. Router A_2 prefers r_3 so router A_1 never learns route r_2, which has the lowest MED value, causing A_1 to use route r_1.

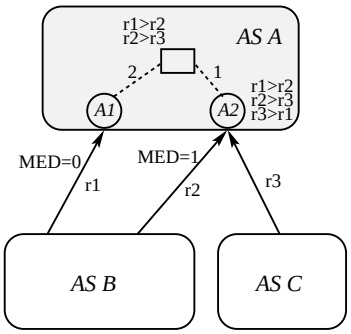

Figure 4: An example where route reflectors and the MED attribute cause oscillations in egress point selection. The route preference of routers A_2 and A_3 are shown. The dashed lines are iBGP sessions and the numbers on them represent IGP path weight.

usefulness, destroys the total ordering of routes, causing many problems [12, 13]. In this section we show two examples of MED-related issues, one with full-mesh iBGP and one involving route reflectors.

Figure 3 gives an example where the MED semantics are violated even with full-mesh iBGP. All three routes are the same in step I comparison in Table 1, but r_2 has a lower MED value than r_1. Between routes r_2 and r_3, router A_2 picks r_3 in the final tie-breaking step. So router A_1 never learns route r_2 and picks r_1 because it is eBGP-learned, even though the correct behavior of AS A is to not use r_1 unless r_2 is withdrawn, say, due to a failure.

Figure 4 gives an example where the interaction between the MED attribute and route reflectors causes oscillation in route selection. Again, all three routes are the same in step I comparison in Table 1. Router A_1 always chooses r_1 because it has a lower MED value and is eBGP-learned. Router A_2 initially learns routes r_2 and r_3, and chooses $r2$ based on router ID tie break. The route reflector learns r_1 and r_2,

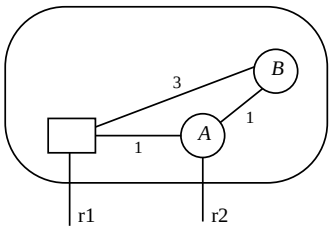

Figure 5: A simple example of packet deflection. The two routes are equal in step I and II comparisons in Table 1. The route reflector chooses r_1 and router A chooses r_2. Router B only learns r_1 so it chooses it. But when packets from B try to exit on r_1, they follow the shortest path and arrive at A, which will sent the packets out on r_2.

and picks r_1 due to MED, so router A_2 now learns all three routes and cannot use r_2 any more because it has a higher MED value, but it prefers r_3 over r_1 because r_3 is eBGP-learned. The route reflector now learns r_1 and r_3, and picks r_3 for a lower IGP path weight. Router A_2 no longer learns r_1 so it switches back to choosing r_2, causing the route reflector to switch back to r_1. The egress path selections of router A_2 and the route reflector oscillate on like this.

The previous examples shows that even with full-mesh iBGP, MED semantics can be violated because some routes are prematurely eliminated (like r_2). Fixing this problem is important because the AS may be disobeying a service contract with a customer. Router A_1 needs to learn all three routes in order to pick r_3, the correct route. Therefore a level i) solution is to terminate every edge link with MED on a separate router. But in order for all the routers to learn all the MED routes, full-mesh iBGP is also required because with route reflectors we can create the same problems by changing the routers into route reflectors (in Figure 4 we would have two levels of route reflectors) and placing a router to terminate each link.

3.3 Limited Route Visibility

With full mesh iBGP sessions, all the routers choose routes from the same set, which includes one best eBGP-learned route from each border router. This ensures that if a router picks the closest exit point then all routers along its shortest path to the exit point also pick the same exit. When route reflectors are used and not all routers learn all the best eBGP-learned routes, a router may choose a different route from what it would choose under full mesh iBGP sessions. If a router on a packet's shortest path to the chosen egress point happens to choose a different egress, the packet will be deflected off its course and may not exit at its chosen egress point. Packet deflection is studied in detail in [14].

Figure 5 is a simple deflection example. The route reflector has two clients, A and B. The two routes r_1 and r_2 are equally good after steps I and II in Table 1, so the route reflector and A choose their eBGP-learned routes r_1 and r_2,

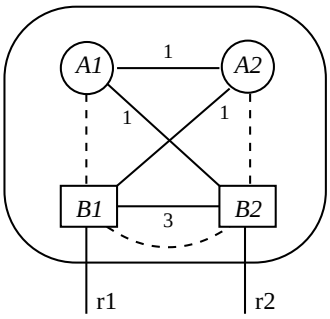

Figure 6: A simple example of data-plane forwarding loop. Solid lines are physical links and dashed lines are iBGP sessions. The two route reflectors learn two equally good routes, respectively. Router A_1 chooses r_1 because it is the only route it learns; similarly router A_2 chooses r_2. A_1's shortest path to its exit point is through A_2 and vice versa, so packets from A_1 and A_2 are stuck in the forwarding loop between them.

respectively. Router B only learns route r_1 and chooses it. When B forwards traffic toward r_1, it sends it to router A, who has chosen r_2. So the packets will exit the AS at A and take route r_2.

When multiple deflections interact there is a chance of creating forwarding loops, which is a much more serious problem. Figure 6 shows an example. If router A_1 is on the shortest path from router A_2 to its exit point and vice versa, there is a forwarding between the two routers. In order to avoid forwarding loops, operators make sure that a route reflector is closer to its clients than to other routers. This is not only an expensive solution but it also does not avoid deflections.

In order to avoid packet deflection in a arbitrary setup, one can use tunnels to forward traffic directly to the chosen exit point. Tunneling was a solution proposed in the early stages of adopting BGP [15], but routers did not support it efficiently until a few years ago. Now several backbones employ full mesh MPLS tunnels where traffic is forwarded from ingress router to egress router in one hop [16].

4. ATOMIC AUTONOMOUS SYSTEMS

In this section we first introduce simple models to describe atomic autonomous systems and routing policies. We then give the formal definition of an atomic AS and discuss how to provide intrinsic atomicity in a few different scenarios. The discussion on intrinsically-atomic distributed routing will guide the next section where we propose measures to restore atomicity to BGP. The analysis in this section applies to policy-based interdomain routing protocols where each AS selects routes from choices offered by its neighbors. BGP is such a protocol and we use it as an example.

4.1 Interdomain Routing with Atomic ASes

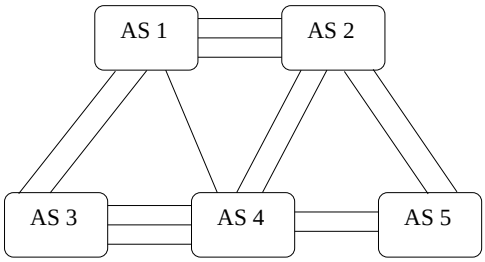

Figure 7: A network of autonomous systems. Each node represents an AS and each link represents a routing adjacency.

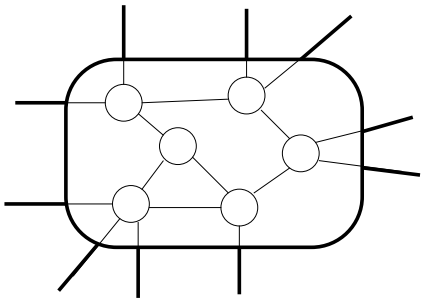

Figure 8: An autonomous system. The nodes represent routers and the lines represent links. The dark big circle is the AS boundary.

We define the boundary of an AS based on what information is available to neighboring ASes and what is not. We then model the functions of an AS by the route exchanges on the boundary.

Figure 7 is a network of ASes, where an AS is represented by a node and a routing adjacency between two ASes is represented by a bidirectional link. Routing information and corresponding traffic travel in opposite directions on these links. An AS is modeled as a node because typically one cannot assume to have information about an AS's internal structure. Since an AS learns routes only from neighbors, it is only concerned with the links connecting to its neighbors. We allow multiple links between two ASes because these edge links are visible to the neighboring AS and different routes may be exported on different links [17]. This differs from the common practice in interdomain routing research which uses one link to represent the connection between two ASes (e.g., [18, 19, 20, 21, 22]).

Figure 8 shows an autonomous system, where each router is represented by a node. Compare Figure 8 to the AS-level graph in Figure 7 and we can see that the dark circle surrounding the routers is the AS boundary, and what going on inside the boundary should not concern other ASes. Suppose the AS has N neighbors indexed by i and with neighbor i it has L_i edge links, indexed by ij, to exchange routing information. A *route* contains partial information about a path

that may span multiple ASes. It has the information an AS needs in order to make route selection and dissemination decisions, so exactly what information a route has depends on the protocol. The separation of intradomain and interdomain routing means that a route need not specify every router on the path. In BGP, for example, a route specifies the AS path, the next-hop router, and a few other optional attributes, to influence route selection and dissemination. To make future notations easier, we introduce the *empty route* ϵ, which gives no information about how to reach a destination.

For a destination prefix d, one route r_{ij} (possibly empty) is learned from edge link l_{ij}. Let $R = \{r_{ij}, i = 1, 2, \ldots, N, j = 1, 2, \ldots, L_i\}$ be the set of all externally learned routes. The AS selects a route $e_{ij} \in R \bigcup \{\epsilon\}$ via export to edge link l_{ij}. If $e_{ij} = \epsilon$ then no route is exported. *There is no distinction between route selection and export* in this model because route selection is per neighbor. This is a departure from the way BGP works today, but is important for atomicity because an atomic AS does not have the constraint that two links must always export the same route. However, we can add such a constraint to model BGP but it makes satisfying atomicity harder. Even though we assume only one route is imported and exported on each edge link in this paper, we can easily generalize the analysis to exchange multiple routes on an edge link.

4.2 AS-wide Policy Model

The policy of an AS reflects the AS's own objectives. It can also reflect the customers' objectives if it respects the preferences the customers communicate to it through route attributes. A policy can be expressed in its preference over routes. An AS normally would like to export differently to different neighbors, so it can have different preferences depending on which neighbor it is selecting routes for. We represent this by the *AS-wide route preference function* $B_i(\cdot)$ for neighbor i, which, given a set of routes R, returns a subset of $R \bigcup \{\epsilon\}$ that the AS most prefers to export to neighbor i. This model is general, because we do not require $B_i(\cdot)$ to impose a total ordering, i.e., we can have $r \in B_i(R)$ and $r \notin B_i(R \backslash \{t\})$ for some $t \in R$ and $t \neq r$. For example, if neighbor i is a customer, and the policy is to export the shortest routes to customers, then $B_i(R)$ will return routes in R that have the shortest AS path length. Note that since a border router can terminate multiple edge links, it needs to pick routes for the edge links independently, according to their corresponding rankings. This means a router may need to maintain several routing tables.

Although the behaviors of BGP can be modeled by AS-wide route preference functions, it is slightly unnatural because BGP does not have per-neighbor route selection. Each router selects only one route for each destination prefix, and then exports all routes to customers and filters peer- and provider-learned routes before exporting to peers and providers. So the BGP implementation described in Section 2 can be modeled as follows: if neighbor i is a customer, $B_i(R)$ returns the set of routes that have the highest local_pref, the shortest AS path length, the lowest origin type, and the lowest MED value among routes from the same neighboring AS; if neighbor i is a peer or customer, then peer- and customer-learned routes are filtered before being exported, so $B_i(R)$ returns the set of routes selected for a customer minus the filtered routes or the empty route.

Also as part of an AS's policy, the set of routes exported to neighbor i, $E_i = \{e_{ij}\}_j$, should satisfy some consistency check agreed upon between the two ASes. We represent this by the *neighbor preference function* $C_i(\cdot)$, which, given a set of routes, returns a subset of routes which are most preferred by neighbor i. A set of exported routes pass the consistency check if $C_i(E_i) = E_i$ Note that a neighbor is expected to check the consistency of all the exported routes E_i, but may only use a subset of them or none at all. Even though consistency check is determined through bilateral agreements between two ASes, the requirements may not be symmetric. For example, two peers are usually expected to export routes of the same AS path lengths to each other, while a customer may be able to export any way it wants to its provider and expects to receive routes from the provider in a certain way.

4.3 Formal Definition of Atomicity

We have informally defined an atomic AS as a network that acts like a single node with a clearly defined policy. Now we formalize this definition with the notations we have introduced in this section. The policy of an AS is represented by the set of AS-wide route preference functions $B_i(\cdot)$. The routers in the AS should select routes that are the most preferred in order to realize AS-wide policy goals. In the mean while, the AS should adhere to its neighbors' expectation on how routes are exported. Finally, data packets should traverse routes specified by the control plane because the ultimate goal of exchanging routing information is to carry traffic to their destinations.

Formally, an AS is *atomic* if it has the following three properties:

- **Policy consistency.** A router should never export a route to neighbor i that is worse than some other route that the AS has learned according to the AS-wide route preference functions $B_i(\cdot)$ for neighbor i, i.e.,

$$E_i \subset B_i(R), \quad \text{for} \quad i = 1, 2, \ldots, N \tag{1}$$

  where R is the set of routes learned by the AS.

- **Export consistency.** An AS must export routes to a neighbor that can pass the corresponding consistency check, i.e.,

$$C_i(E_i) = E_i, \quad \text{for} \quad i = 1, 2, \ldots, N \tag{2}$$

- **Data-plane consistency.** Data traffic follows the paths indicated by the exported routes, i.e., packets entering the AS on link l_{ij} will exit the AS on the edge link where route e_{ij} was learned.

The three atomic properties are stated in terms of necessary conditions and, as such, are not concerned with how these properties are satisfied. They are loose enough to give an AS flexibility in deciding exactly which routes to export. For example, the policy consistency requirement does not require that a router picks the same route as it would pick if the network has full visibility. And data-plane consistency requires that the AS as a node does not deflect traffic, and does not require that the AS is deflection-free inside. So an atomic AS behaves like a single node in the sense that the decisions of all routers are consistent, but it does not necessarily behave like a single router or an AS with a full-mesh iBGP configuration.

4.4 Intrinsically Atomic Routing Protocols

In this subsection we discuss how to design distributed routing protocols to *guarantee* that the first two atomic properties are satisfied in an AS. We focus on distributed protocols because satisfying atomicity in a centralized environment is relatively straightforward. To achieve intrinsic data-plane consistency, using tunnels is the most straightforward solution. Since a route is picked for each edge link independently, having just router-to-router tunnels is not sufficient, because a border router with multiple edge links still needs to know which link the packet should go out to. So a full-mesh of tunnels connecting every edge link to all the other edge links is sufficient. These tunnels are used only for delivery of packets and need not provide quality of service guarantees, so only encapsulation is needed. So they need not be as complicated as MPLS tunnels has packet signaling.

4.4.1 Centralized Control

In a centralized route selection and dissemination architecture like [23], the central server learns all the routes (R), e.g., by having eBGP sessions with neighboring ASes' border routers, selects a route for each edge link from $R \bigcup \{\epsilon\}$, and sends each route to the corresponding border router. The central server has *full visibility* because With full visibility of the routes, the server can select for neighbor i out of best set $B_i(R)$ that can pass the consistency check by neighbor i. For example, it can select the set $C_i(B_i(R))$.

A trivial extension of centralized control is to disseminate all routes to all routers so that each router can perform the route-selection process exactly as the central server would do. This may not be desirable because of scalability concerns. We say a network has *full visibility* if every router learns *all* the routes. We say a network has *equivalent full visibility* if every router learns enough routes to guarantee that it selects the same route as if the network had full visibility.

4.4.2 When Routes Have Total Ordering

Now we consider the simple case where the routes have total ordering, i.e., if $r \in B_i(R)$ then $r \in B_i(R \bigcup \{t\})$ for any $t \notin R$. We further assume that $B_i(\cdot)$ and $C_i(\cdot)$ intrinsically agree, i.e., $C_i(B_i(S)) = B_i(S)$ for any set of routes S. This means export consistency is automatically satisfied if policy consistency is.

In this case each router can pick a best route according to $B_i(\cdot)$ out of all the routes it learns and disseminates it. As long as the signaling graph is connected, every router will learn at least one best route, so atomicity is satisfied.

4.4.3 For General Preference Functions

We still assume that $B_i(\cdot)$ and $C_i(\cdot)$ intrinsically agree, i.e., $C_i(B_i(S)) = B_i(S)$ for any set of routes S, so we only need to satisfy policy consistency. If $B_i(\cdot)$ does not satisfy total ordering, then we need to be careful when deciding not to disseminate a route. Suppose a router (indexed by r) keeps a subset S_r of all learned routes which could influence route selection of any router. It only needs to disseminate the routes in S_r to the other routers. Due to the lack of total ordering, it is possible that removing a route t from the set S_r may cause a router to select a route not in $B_i(S)$. Then the route t must be disseminated. So a route can be taken out of set S_r if doing so will not cause policy consistency to be violated. Formally, for route t in the candidate set, if we have $r \in B_i(S \backslash \{t\})$ for any superset of routes $S \supseteq S_r$, for any $r \in B_i(S)$ and $r \neq t$, and for all i, then route t can be taken out of the S_r without causing violation of atomicity. Note that once we take out a route the set S_r change, so if we want to see if another route can be taken out we have to go through the exercise again. Taking out multiple routes at a time is dangerous because suppose S_r has two equally good routes and each of them can be taken out with out violating policy consistency, but taking out together could mean there are no more routes to choose from.

From the discussion above we can see that well designed policies and consistency checks can greatly simplify protocol design.

5. RESTORING ATOMICITY TO BGP

In this section we present two ways to make BGP atomic: by giving sufficient conditions which constraints how the network should be configured and how the protocol should be used, and by making modest changes to the current BGP protocol to make it intrinsically atomic.

5.1 By Constraining the Current BGP

We now re-examine the fixes summarized in Table 2 and argue that they are sufficient to restore BGP atomicity. We explain the motivations for these conditions and discuss alternative approaches.

5.1.1 Policy consistency

Let us assume the AS has a full-mesh iBGP configuration. The current BGP route-selection process (shown in Table 1) does not have total ordering because of the MED attribute, therefore disseminating single best route per router does not

provide policy consistency. We observe that the routes considered inferior in stages (i) and (ii) (in Table 1) do not affect the final outcome and the routes considered "the best" after stage (ii) are $B_i(R)$ for neighbor i who is a customer (if i is a peer then the set $B_i(R)$ is a subset of the best routes). Therefore disseminating all routes after stage (ii), i.e., all routes in $B_i(R)$, provides equivalent full visibility.

We can reduce the number of routes disseminated by letting each router disseminate at most one best route learned from each neighboring AS. This means, if a router learns two routes in $B_i(R)$ that are from the same neighbor AS, only one of them needs to be disseminated. This does not provide equivalent full visibility, in that a router may choose a different route than it would with full visibility, but can guarantee that routers only select routes that are in the set $B_i(R)$. We need to disseminate one best route from each neighboring AS because that the MED attribute is only compared among routes learned from the same AS.

Usually an AS only respects a neighbor's MED attribute when the business contract dictates so. For the neighbors whose MED attribute do not need to be respected, the AS typically reset the MED values to zero upon importing the routes. Routes learned from these neighbors are compared even if they are not from the same AS. Thus we can further reduce the number of routes disseminated by disseminating only one best route among the all the routes learned from the ASes whose MED is not respected, instead of one best route from each AS.

The solution in Table 2 makes an edge link connecting to a neighbor who uses MED terminate on a separate router, so all MED routes are disseminated. The routers that do not have routes using MED disseminate one best route. This is enough route dissemination to ensure that policy consistency is satisfied.

5.1.2 Export Consistency

The examples in Section 3.1 show that when export gives different treatments to two selected routes, which are considered equally good by the selection process, there is a danger of having inconsistent export. In order to solve this problem, it is sufficient to make sure export treats all routes that are considered equal in route selection in the same way, which means when exporting routes to a particular neighbor, routers can filter only based on information common to these selected routes, i.e., the next-hop AS, the local_pref, the AS-path length, and the origin type.

The conditions in Table 2—not allowing the use of communities that modify or filter routes on export, strictly preferring customer-learned routes over peer-learned routes (i.e., assigning them different local_pref values), and doing export filtering only based on business relationships (i.e., filtering based on next-hop AS type and local_pref)—make sure route selection agree with export rules. We can relax the first condition to allow the use of communities but take them into consideration when setting local_pref, i.e., giving routes with the "no export" and "prepend" communities with a lower local_pref value than routes without communites. This way communities influence both route selection and export in the same way.

5.1.3 Data-plane Consistency

The conditions in Table 2, using full-mesh iBGP and hot-potato routing, provide consistent forwarding. If we allow the use of route reflectors, the two sufficient conditions to guarantee forwarding correctness given in [14] require that a route reflector is closer to its clients than to other routers and that all IGP shortest paths are also signaling paths. The first condition alone is enough to guarantee loop-free forwarding, and is often satisfied in practice by using many route reflectors, which is not only expensive but also unscalable.[2] The second condition, which would guarantee the network to be free of deflections, is much harder to satisfy and is rarely done in practice. Recent work [24] shows that this condition can be satisfied by carefully choosing the locations of route reflectors. Finally, using data-plane tunnels can always guarantee data-plane consistency, as discussed in Section 4.4. The fact that using tunnels can simplify the interaction between BGP and IGP was recognized when BGP was first used [15], but the technology (e.g., MPLS) has only matured in recent years.

Discussion: As we can see, these "sufficient conditions" approaches to make BGP atomic are highly constraining on how one can configure the network and use the protocol. They also require many extra routers, have high configuration overhead, are prone to errors, and may require frequent reconfigurations. For example, if a customer decides to start using MED, then the service provider may have to add routers to make sure each of the customer's edge links terminate on a separate router.

5.2 By Making BGP Intrinsically Atomic

Making today's BGP atomic is not easy as we have to put many constraints on how the protocol is used, some of which can be difficult to adhere to in practice. For example, if an ISP gets a new customer who would like to use MED, then the operators have to buy extra routers so terminate the customer's links. This can take too much time to make the scheme practical. Fortunately with a few modest changes to the way routes are selected, disseminated, and exported, we can make it correctly realize a clearly defined policy without special configurations. What is left, then, is to define the policy so that it complies with export consistency checks and to set up edge-link to edge-link tunnels for forwarding data packets.

Based on the guidelines for designing intrinsically atomic distributed routing protocol for general preference functions, given in Section 4.4.3, we propose modifications to BGP

[2]To make matters worse, operators need to ensure that even if one of the route reflectors fails, the condition is still satisfied, often doubling the number of route reflectors required.

(and iBGP) so that the result is an intrinsically atomic routing protocol. Some of these changes may seem like radical departures of the protocol today, but we argue that these changes can be realized within a few years' time.

Current BGP selects routes by sequentially executing the three stages in Table 1. After a router selects a route it disseminates it via iBGP sessions and may filter or manipulate the route before exporting it. The modified BGP we propose has two major changes: First, the route selection and dissemination process is changed so that route manipulation is done upon import and routes are disseminated before tie breaking; second, it has per neighbor routing table and route selection is done separately for each neighbor. This means that there is no need to filter routes because the routes to be filtered will simply not be selected. The selected route for a neighbor is exported to the neighbor.

The proposed procedure of a router's operation (for each neighbor) is:

1. Modify routes according to communities, etc., if needed;
2. Select routes with the highest local_pref, lowest AS path length, and lowest origin type;
3. Select routes with the lowest MED among those learned from the same neighbor;
4. Among the routes learned from the same neighbor who uses MED, only one route needs to be kept. Among all routes learned from the neighbors who do not use MED, only one route needs to be kept;
5. Disseminate all the routes that are left at this point;
6. Each router tie breaks according to eBGP vs. iBGP comparison, IGP cost, and router ID, and selects one route (possibly empty).

The above procedure ensures policy consistency, and we call this *atomic BGP*.

5.2.1 Feasibility of Atomic BGP

To provide routers the necessary visibility to the candidate routes, atomic BGP requires more routes to be propagated and stored than today's BGP does. This is mainly a problem for large ISPs that have many neighbors and high path diversity. In this section we explore the growth factor of the storage requirement, because the storage overhead and dissemination overhead are similar. We assume that tunnels are used to forward traffic, therefore eliminating the need for internal routers to speak BGP, so we only need to consider the border routers.

In atomic BGP, each border router selects routes for each neighbor independently and keeps a routing table for each neighbor. But many neighbors may share the same preference function so each router only needs to keep a routing table for each *class* of neighbors (neighbors correspond to the same preference function). In general, there are only two classes of neighbors for exporting routes: the first class is customers and the second class is peers and providers. For each neighbor class, a router disseminates at most one route learned from the same neighbor, so the number of routes a router disseminates is no more than the number of best AS paths. Since a router may not learn all paths, the number of best AS paths likely gives a loose upper bound on the number of routes disseminated. So the growth factor for storage requirement is upper-bounded by two times the average number of best AS paths to reach a prefix. Note that this upper bound does not depend on the iBGP configuration, whether it is router reflector based or a full mesh.

It bas been reported that AT&T has only one best AS-level path to about 50% of the prefixes, and no more than three best AS-level paths to about 85% of the prefixes [25]. Our estimation based on the data provided in [25] indicates that AT&T has on average about 2.2 best AS-level paths per prefix. This means the grow factor of the storage requirement and the grow factor of the number of route update messages are less than $2 \times 2.2 = 4.4$.

6. RELATED WORK

Most previous work on modeling BGP has focused on whether the routing protocol converges, considering either inter-AS policy conflicts [18, 10] or intra-AS issues raised by route reflectors and the Multiple Exit Discriminator (MED) attribute [13, 14, 12]. Although some of our examples in Section 3 draw on previous work [6, 13, 14, 12], we focus broadly on the "correctness" of a single AS in steady state. Earlier studies also explored how an individual AS can use local BGP measurements to detect anomalies [6, 26, 27] or perform traffic engineering [28], but they do not deal with the correctness issues either.

Feamster and Balakrishnan define three aspects of routing-protocol correctness—route validity, path visibility, and safety—that can be used to evaluate the properties of routing protocols [29]. However, their work does not consider the AS-level protocol correctness, i.e., how an AS can behave "correctly" to other ASes as a whole when speaking certain protocol (e.g., BGP). Our work focuses on the intra-AS aspects of participating in an interdomain routing protocol and presents conditions that can be used to both verify if an AS behaves "correctly" to other ASes (behave atomically) and to guide the design of protocols that make an AS "intrinsically atomic."

Muhlbauer et. al perform a comprehensive measurement study and point out that, different routers in the same AS often pick different best routes [17]. They also propose heuristics to build a model of an AS that fits the observed AS-level paths. Our work proposes a precise definition of what it means for an AS to "behave like a single node"—while remaining faithful to the reality that different routers may select different best routes.

Finally, our work is motivated in part by BGP protocol extensions [30] and new interdomain routing architectures [20, 21, 22, 31] that assume an AS behaves like a single node.

The three atomic conditions proposed in this paper can help the designers of new routing protocols ignore intra-AS details, without sacrificing correctness.

7. CONCLUSION

An Autonomous System (AS) consists of a distributed collection of routers that should, collectively, act like a single node with a clearly-defined routing policy. Behaving "atomically" is relatively easy when ASes are small and routing policies are simple. For example, an AS is atomic if all routers select a route with the shortest AS path, disseminate routes using a full-mesh of iBGP sessions, and export the best route to all eBGP neighbors. However, many of today's ASes are large networks with complex routing policies Over the years, BGP has evolved to provide greater scalability (through route reflectors and confederations) and flexibility (through additional route attributes and steps in the BGP decision process). Today, an AS can easily violate atomicity, leading to subtle routing problems and policy violations, even when all routers are configured the same way.

In this paper, we presented a precise defintion of atomicity, in terms of three conditions—policy consistency, export consistency, and data-plane consistency—as well as distributed protocols that achieve these goals. Applying these conditions to BGP, we identified a set of guidelines an AS can follow in configuring how BGP selects, disemminates, and exports routes without violating atomicity. We also identified modest extensions to BGP that would allow an AS to be "intrinsically atomic." We argue that deploying these modifications would make interdomain routing simpler to design, measure, model, and analyze. In our ongoing work, we are extending our model to networks that consist of multiple ASes. Many large ISPs are conglomerates of multiple ASes, sometimes for historical reasons (e.g., mergers of several ISPs) and sometimes for scalability reasons. Our goal is to identify conditions that would allow a *group* of ASes "act like a single node."

8. REFERENCES

[1] D. Clark, "The design philosophy of the DARPA Internet protocols," in *Proc. ACM SIGCOMM*, pp. 106–114, 1988.

[2] J. Hawkinson and T. Bates, "Guidelines for creation, selection, and registration of an Autonomous System (AS)." RFC 1930, March 1996.

[3] "AOL Transit Data Network Settlement-Free Interconnection Policy." `http://www.atdn.net/settlement_free_int.shtml`.

[4] "AT&T Global IP Network Settlement-Free Peering Policy." `http://www.corp.att.com/peering/`.

[5] "Verizon Business Policy for Settlement-Free Interconnection with Internet Networks." `http://www.verizonbusiness.com/uunet/peering/`.

[6] N. Feamster, Z. M. Mao, and J. Rexford, "Borderguard: detecting cold potatoes from peers," in *IMC '04: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, pp. 213–218, 2004.

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

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

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

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

[11] R. Chandra, P. Traina, and T. Li, "BGP Communities Attribute." RFC 1997, August 1996.

[12] T. Griffin and G. T. Wilfong, "*Analysis of the MED Oscillation Problem in BGP*," in *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, (Washington, DC, USA), pp. 90–99, IEEE Computer Society, 2002.

[13] A. Basu, C.-H. L. Ong, A. Rasala, F. B. Shepherd, and G. Wilfong, "Route oscillations in I-BGP with route reflection," *Proc. ACM SIGCOMM*, vol. 32, no. 4, pp. 235–247, 2002.

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

[15] Y. Rekhter and P. Gross, "Application of the Border Gateway Protocol in the Internet." RFC 1655, July 1994.

[16] C. Metz, C. Barth, and C. Filsfils, "Beyond mpls ... less is more," *IEEE Internet Computing*, vol. 11, no. 5, pp. 72–76, Sept.-Oct. 2007.

[17] W. Mühlbauer, A. Feldmann, O. Maennel, M. Roughan, and S. Uhlig, "Building an AS-topology model that captures route diversity," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 4, pp. 195–206, 2006.

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

[19] W. Xu and J. Rexford, "MIRO: Multi-path interdomain routing," in *Proc. ACM SIGCOMM*, pp. 171–182, 2006.

[20] L. Subramanian, M. Caesar, C. T. Ee, M. Handley, M. Mao, S. Shenker, and I. Stoica, "HLP: A next generation inter-domain routing protocol," in *Proc. ACM SIGCOMM*, pp. 13–24, 2005.

[21] X. Yang, "NIRA: A new Internet routing architecture," in *FDNA '03: Proceedings of the ACM SIGCOMM workshop on Future directions in network*

architecture, pp. 301–312, 2003.

[22] D. Zhu, M. Gritter, and D. R. Cheriton, "Feedback based routing," *Proc. ACM SIGCOMM*, vol. 33, no. 1, pp. 71–76, 2003.

[23] Y. Wang, I. Avramopoulos, and J. Rexford, "Morpheus: making routing programmable," in *INM '07: Proceedings of the 2007 SIGCOMM workshop on Internet network management*, pp. 285–286, 2007.

[24] M. Vutukuru, P. Valiant, S. Kopparty, and H. Balakrishnan, "How to Construct a Correct and Scalable iBGP Configuration," in *IEEE INFOCOM*, (Barcelona, Spain), April 2006.

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

[26] J. Wu, Z. M. Mao, J. Rexford, and J. Wang, "Finding a needle in a haystack: Pinpointing significant BGP routing changes in an IP network," in *Proc. Networked Systems Design and Implementation*, May 2005.

[27] Y. Huang, N. Feamster, A. Lakhina, and J. Xu, "Detecting network disruptions with network-wide analysis," in *Proc. ACM SIGMETRICS*, June 2007.

[28] N. Feamster and J. Rexford, "Network-wide prediction of BGP routes," *IEEE/ACM Transactions on Networking*, vol. 15, no. 2, pp. 253–266, 2007.

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

[30] D. Pei, M. Azuma, D. Massey, and L. Zhang, "BGP-RCN: Improving BGP convergence through root cause notification," *Comput. Netw. ISDN Syst.*, vol. 48, no. 2, pp. 175–194, 2005.

[31] P. F. Tsuchiya, "Efficient and robust policy routing using multiple hierarchical addresses," in *Proc. ACM SIGCOMM*, pp. 53–65, 1991.