Improving BGP Convergence Through Consistency Assertions

Dan Pei, Xiaoliang Zhao, Lan Wang, Dan Massey, Allison Mankin, S. Felix Wu, Lixia Zhang

Abstract—

This paper presents a new mechanism for improving the convergence properties of path vector routing algorithms, such as BGP. Using a route's path information, we develop two consistency assertions for path vector routing algorithms that are used to compare similar routes and identify infeasible routes. To apply these assertions in BGP, mechanisms to signal failure/policy withdrawal, and traffic engineering are provided. Our approach was implemented and deployed in a BGP testbed and evaluated using simulation. By identifying and ignoring the infeasible routes, we achieved substantial reduction in both BGP convergence time and the total number of intermediate route changes.

Keywords— **BGP, Convergence, Assertions, Path Vector Routing Protocol,**

I. Introduction

THIS paper presents an approach for improving the convergence time of Internet routing. The Internet is composed of thousands of Autonomous Systems (ASes), loosely defined as networks and routers under the same administrative control. BGP[1] is the de facto inter-AS routing protocol and BGP adapts to both changes in network topology and changes in AS routing policies. Ideally BGP would quickly adapt to changes and converge on a new set of stable routes. However, it has been observed that in many cases, BGP routers explore a large number of possible routes before converging on a new stable route. The route changes that occur during a convergence period can result in lost packets or delayed delivery as well as added overhead to BGP routers. Lengthy BGP convergence is a problem for the Internet today and threatens to become a larger problem as the Internet continues to grow in size. The objective of this work is to reduce the BGP route convergence time and minimize the number of route changes that occur during the convergence period.

Labovitz et al.[2] found that the delay in Internet inter-domain path fail-over now averages 3 minutes, and some non-trivial percentage of fail-overs trigger routing table oscillations lasting up to 15 minutes. Such a delay in route convergence will cause packet drops, loss of connectivity, and long end-to-end delay in the Internet.

TIME	BGP Message/Event
10:40:30	Route Fails/Withdrawn by AS2129
10:41:08	2117 announce 5696 2129
10:41:32	2117 announce 1 5696 2129
10:41:50	2117 announce 2041 3508 3508 4540 7037 1239 5696 2129
10:42:17	2117 announce 1 2041 3508 3508 4540 7037 1239 5696 2129
10:43:05	2117 announce 2041 3508 3508 4540 7037 1239 6113 5696 2129
10:43:35	2117 announce 1 2041 3508 3508 4540 7037 1239 6113 5696 2129
10:43:59	2117 sends withdraw

Fig. 1. Slow Convergence in the Internet

Figure 1 shows an example of BGP slow convergence. This example, taken from [3], actually occurred in the Internet and shows one router's view of the convergence problem. Figure 1 shows the BGP updates sent from AS 2117 for route to a destination within AS 2129. A single BGP withdrawal from AS 2129 triggers AS 2117's six unnecessary announcements and one withdrawal.

Figure 1 illustrates the delayed convergence problem that occurs after a route failure, but similar problems can occur when an AS switches to an alternate route. [3] also showed that multi-homed fail-over is equivalent to route failure, with respect to both convergence latency and the number of update messages triggered.

Note that in the Figure 1, AS 2129 reports that it has lost its route to the destination. AS 2117 finds and announces 6 different routes to the destination, but all the 6 these route end in AS 2129. Since AS 2129 has lost its route to the destination, all these 6 routes are invalid and are eventually discarded.

Section III will show how events, such as the withdrawal event by AS 2129, can be used to detect invalid routes and allow BGP, which is a path-vector routing protocol, to converge more quickly. The general assertions for path vector routing protocols are developed. The assertions are en-

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No DABT63-00-C-1027. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the DARPA.

hanced for BGP to handle traffic engineering using a logical AS solution. Although these assertions still cannot eliminate *all* the transient route changes, as explained in a later section, both our testbed experiments and the simulations results show dramatic reductions in BGP convergence time after our assertions were implemented. In our network testbed (described in Section VI), convergence time for a failure withdraw decreased from 30.3 seconds to 0.3 seconds and the convergence time after a route change decreased from 64.9 seconds to 0.1 seconds. In simulation tests with a network topology made of 61 ASes, the convergence time decreased from 823.81 seconds to 1.335 seconds and the number of route changes decreased from 17096 to 4193. Section VI presents these results in detail.

The remainder of the paper is organized as follows. Section II reviews the previous work. Section III presents the assertions for improving the performance of a simple path vector protocol and Section IV presents the enhanced assertions for BGP. Section V describes how these assertions are implemented in BGP. Section VI presents the results of testbed deployment and simulation of the assertions. Finally, Section VII summarizes the paper.

II. Previous Work

The BGP routing protocol has been studied from both a practical and a theoretical perspective. The results in this paper are motivated by these previous studies of BGP performance in the Internet.

In 1997, Labovitz et al [4] observed millions of BGP route changes and update messages per day. Most of these changes were pathological updates that did not correspond to legitimate topology changes. This degraded the network performance, wasted the network bandwidth and sometimes even caused BGP routers to crash. These problems were primarily caused by improper software implementations and router mis-configurations.

Following studies on the Internet stability and wide-area network failures [5], [2], [6] found that route changes and route failures resulted in a significant delay before BGP converged on a new set of stable routes. This delay averaged three minutes and some changes took up to 15 minutes. Current BGP implementations explore a potentially large number of backup routes when a failure occurs and many of the backup routes may be already invalid. Analysis in [2] showed that in the theoretical worst case a completely connected n-AS system might explore all (n!) possible paths during the convergence period. Later work demonstrated through experiments that the convergence time is proportional to the longest possible backup autonomous system path between the source and destination node[6]. [2] also showed through simulations that if loop detection is performed on both the sender and receiver side, the convergence time of 7-node completely connected network could be reduced from 120 seconds to 30 seconds. This paper presents a new technique for further improving BGP convergence time.

It should be noted that BGP might never converge to a stable route. Discussions on this BGP route divergence problem are beyond the scope of this paper.

III. Assertions for Improving Routing Convergence in Simple Path Vector Protocols

We define a **routing convergence period** as the period that starts when a previously stable route to some destination D becomes invalid and ends when the network has obtained a new stable route for D (or when D has been correctly declared unreachable). We evaluate routing convergence based on the length of the convergence period and the number of intermediate route changes that occur during the convergence period. Due to factors such as processing and propagation delay, obtaining the new route will always require some time and there will always be at least one route change since the previous stable (and now invalid) route must be removed. We say a slow convergence problem occurs if the convergence time greatly exceeds the time that would have been required to propagate the new stable route if no invalid intermediate routes had been tried.

In terms of route convergence, path vector protocols such as BGP[1] were considered to be an improvement over earlier distance vector routing protocols such as RIP[7]. In a distance vector protocol, routers advertise their distance to each destination. The distance conveys little or no information about the actual path used to reach the destination and distance vector protocols suffer from slow convergence problems caused by transient routing loops and “counting to infinity”[8]. A path vector protocol improves upon distance vector protocols by advertising the path to each destination. For example, BGP route updates include the path of ASes used to reach the destination. It was believed that the addition of path information would prevent routing loops, thus eliminate slow convergence problems, but the previous work in Section II has shown otherwise.

In [9], [10], [11], [12], the convergence properties of distance vector algorithms were improved by exploiting the relationships between routes and using this information to detect invalid routes. Similarly, we look for relationships between path vector routes and use these relationships to detect and ignore invalid routes. The resulting approach allows a router to disregard (invalid) intermediate routes that might occur during the route convergence period. This reduces both the convergence time and the total number of route changes.

The BGP Protocol is a practical path vector protocol being used in the Internet and has a number of important implementation details that are considered in the paper. But in order to clearly present the main concept, we will also introduce and make use of the Simple Path Vector Protocol described below.

Definition 1: **Simple Path Vector Protocol** A Simple Path Vector Protocol is the path vector protocol in which each node can select and use only one of its available routes to one destination and can only advertise to its neighbors the route it is using. That is, in Simple Path Vector Protocol, a node will only advertise one single route to one destination to its neighbors.

The Simple Path Vector Protocol differs from BGP in some respects. The Path Vector is roughly equivalent to the AS Path and an AS might be viewed as a node in the simple path vector protocol. However, an AS doesn't match the Simple Path Vector definition because of traffic engineering, withdrawals due to policy changes, AS partitions, and so forth. A more detailed discussion of these problems will be presented in Section IV.

In this section, we will first present the assertion results for the Simple Path Vector Protocol. The BGP specific results in Section IV are obtained by extending the results in this section.

A. Consistency Theorem for Simple Path Vector Protocols

To illustrate the relationship between different routes in Simple Path Vector Protocols, suppose node R has learned two potential routes to D. Neighbor N_1 is advertising the route (N_1, A, M, C, D) and neighbor N_2 is advertising $(N_2, U, M, W, X, Y, Z, D)$. By examining the relationship between these two routes, one can conclude that at least one of these routes must be invalid. If one believes N_1, then M's route to D is (M, C, D). If one believes N_2, then M's route to D is (M, W, X, Y, Z, D). Since M can only advertise one route to D, either N_1 or N_2 (or both) must be advertising an invalid route to D.

The relationship between two paths that share a common node can be formalized as follows:

Theorem 1: Let $path(N_1, D) = (N_1, P_1, P_2, ..., P_n, D)$ be the path from N_1 to D and let $path(N_2, D) = (N_2, Q_1, Q_2, ..., Q_m, D)$ be the path from N_2 to D in a Simple Path Vector Protocol. If $Q_i = P_j$ for some i and j, then either $n - i = m - j$ and $Q_{i+k} = P_{j+k}$ for all $1 \leq k \leq n - j$, or at least one of the paths is invalid.

- **Proof:**

Suppose $Q_i = P_j = M$. In other words, suppose the path from N_1 to D and the path from N_2 to D intersect at node M. In Simple Path Vector Protocol, node M can only have one valid route to D and let $path(M, D)$ denote this route. The path from N_1 to D can be written as $path(N_1, D) = path(N_1, M) + path_{N_1}(M, D)$, where $path_{N_1}(M, D) = (P_j, P_{j+1}, ...P_n, D)$. If $path_{N_1}(M, D) \neq path(M, D)$, then N_1 has incorrect view of how packets reach D, the route advertised by N_1 is invalid, and the Theorem holds. Otherwise it must be the case that $path_{N_1}(M, D) = path(M, D)$.
Similarly, the path from N_2 to D can be written as $path(N_2, D) = path(N_2, M) + path_{N_2}(M, D)$, where $path_{N_2}(M, D) = (Q_i, Q_{i+1}, ..., Q_m, D)$. If $path_{N_2}(M, D) \neq path(M, D)$, then N_2 has incorrect view of how packets reach D, the route advertised by N_2 is invalid, and the Theorem holds. Otherwise, $path_{N_2}(M, D) = path(M, D) = path_{N_1}(M, D)$. This implies that $Q_{i+k} = P_{j+k}$ for all $1 \leq k \leq n - j$ and the Theorem holds.

This theorem is theoretically true, but is difficult to apply in practice. Assume the theorem was checked and a conflict was found. In other words, N_1 and N_2 both advertise a route to D and these routes intersect at a common node, M. The $path_{N_1}(M, D) \neq path_{N_2}(M, D)$ so according to Theorem 1, at least one of these routes must be invalid. However, Theorem 1 offers no indication of whether the route from N_1 or the route from N_2 (or both) is invalid. In practice, path vector protocols don't check Theorem 1 since even if a conflict was detected, there is no clear way to determine which of the conflicting routes is invalid.

However, in the particular case of $M = N_1$, one could claim that information received directly from N_1 should take precedence over information about N_1 that was received indirectly via N_2. In this case, one should mark the route from N_2 as **infeasible**. An infeasible route can not be selected as the best route to D, but the infeasible will be retained until either N_2 changes the infeasible route or until a route change from N_1 removes the conflict.

For example, suppose neighbor N_1 is advertising the route (N_1, X, Y, Z, D) and neighbor N_2 is advertising the route (N_2, N_1, D). These routes intersect each other at N_1 and $(N_1, X, Y, Z, D) = path_{N_1}(N_1, D) \neq path_{N_2}(N_1, D) = (N_1, D)$. The intuition behind our approach says that the route (N_1, X, Y, Z, D) that was learned directly from N_1 should take precedence over the route (N_1, D) that was learned indirectly from N_2. Because the route (N_1, X, Y, Z, D) is believed to correct, the route (N_2, N_1, D) is marked as infeasible and can not be selected as the best route to D. The route (N_2, N_1, D) remains infeasible until N_2 changes this route or until N_1

announces that its new route is (N_1, D).

Note that even the information from N_1 is usually preferred, it is still possible that the information from N_2 is correct. This is why the route from N_2 is marked infeasible instead of being removed.

On one hand, if the route advertised by N_2 was invalid, then ignoring this route avoids an incorrect route change and prevents an invalid route from being advertised to downstream neighbors. On the other hand, If the route advertised by N_2 was valid, then a valid route is being marked as infeasible and is being ignored. However, this can only occur if there is also pending update from N_1 that would correct the conflict with N_2. The maximum amount of time that N_2's route will be marked infeasible and ignored is bounded by the time required for the pending update to arrive from N_1. Furthermore, if paths are selected based on the shortest path length rule for selecting routes, then the temporarily ignored route from N_2's is not the best route to D. The $path(N_1, D)$ is shorter than $path(N_2, N_1)+path(N_1, D)$. By ignoring N_2's route, the router is ignoring a route that would change as soon as the pending update from N_1 arrives.

We applied Theorem 1 to the restricted case where $N_1 = M$ and obtained the following two assertions for processing path vector route updates and determining the feasibility of a route.

B. The Route Withdrawal Assertion

In Simple Path Vector Protocols, assume node R has neighbors $N_1, N_2, ..., N_n$, $path(N_i, D)$ is the last path to D reported by neighbor N_i, and assume that N_{lost} withdraws its route to D. The $path(N_{lost}, D)$ is set to $NULL$ and R checks whether any existing route to D will be invalidated by this withdrawal.

If N_{lost} appears in $path(N_i, D)$, then N_i depended on the lost route $path(N_{lost}, D)$ to reach D. According to Theorem 1, the invalidated $path(N_i, D)$ is not modified or removed, but it is marked as infeasible and it can not be used as R's route to D.

C. The Route Change Assertion

The Route Change Assertion is similar to the Route Withdrawal Assertion, only now two different feasibility checks are applied. First, the new route is used to check the feasibility of existing routes. Second, the existing routes are used to check the feasibility of the new route.

Assume node R has neighbors $N_1, N_2, ..., N_n$, $path(N_i, D)$ is the last path to D reported by neighbor N_i, and assume that R receives a new route, $path(N_{change}, D)$, from neighbor N_{change}.

First, R checks to see whether $path(N_{change}, D)$ invalidates any existing routes to D. If N_{change} appears in $path(N_i, D)$ and $path(N_i, D) \neq path(N_i, N_{change}) + path(N_{change}, D)$, then the new route from N_{change} invalidates the existing route N_i. The invalidated $path(N_i, D)$ is not modified or removed, but $path(N_i, D)$ marked as infeasible and N_i can not be used as R's route to D.

Second, R checks the feasibility of $path(N_{change}, D)$ to see if it is invalidated by the existing routes. If N_i appears in the $path(N_{change}, D)$ and $path(N_{change}, D) \neq path(N_{change}, N_i) + path(N_i, D)$, then the existing route from N_i invalidates the new route from N_{change}. The invalidated route $path(N_{change}, D)$ is retained, but marked as infeasible and N_{change} can not be used as R's route to D.

IV. ENHANCED ASSERTIONS FOR BGP

Theorem 1 and Route Withdrawal/Change Assertions in Section III provide mechanisms for improving the convergence for Simple Path Vector Protocols. One may consider an AS as a node in Simple Path Vector Protocols. However, because there may be more than one BGP routers in one AS(neighbors in the same AS are called **iBGP peers**, and neighbors in neighbor ASes are called **eBGP peers**), BGP does not fit completely into Simple Path Vector definition. The reasons are:

- Due to traffic engineering, different routers within one AS may advertise different routes to different neighbors.
- One AS may become partitioned into several parts due to failure of internal links, resulting different routes advertised.
- Due to policy reasons, one AS may choose not to advertise to some neighbors the route that it is using, while advertising the route to others.
- One AS number may consecutively appear multiple times in one AS path.

For the consecutive (and the same)AS numbers, simple processing should be conducted to make sure that such consecutive AS numbers is viewed as one single AS number. Detailed discussion on how to address the other 3 problems will be presented in following subsections.

A. Logical AS Solution for Traffic Engineering

With traffic engineering, different routers within one AS may advertise different routes to their eBGP peers.Our analysis of the BGP routing table from Oregon Route Views Server [13] on 06/08/2001 showing that AS701 (one Backbone ISP) is advertising multiple routes to each of the 3596 destinations among totally 102,677 destinations. Two routes to prefix 169.131.0.0/16 (in Figure 2) shows

Neighbor AS	Route to prefix 169.131.0.0 / 16
AS1	1 701 6079 4527
AS1740	1740 701 6347 4527

Fig. 2. Traffic Engineering Example in the Internet

that AS701 advertises (701 6079 4527) to AS1, and advertises (701 6347 4527) to AS1740.

Further analysis shown that among 121,602 prefixes in the Internet, about 56,081 prefixes are involved with traffic engineering by one or more ASes. Among 11,514 ASes, about 125 ASes are doing traffic engineering. (these numbers are the average number based on the data from 07/10/2001 to 07/18/2001 on Oregon Route Views Server, and does not necessarily reflect latest status of the Internet.)

A.1 Logical AS Solution

One AS, through multiple BGP routers, may advertise multiple routes to one single destination in the Internet due to traffic engineering and AS partition. However, we must stress the fact that a single BGP router can only advertise the route to the destination that it itself is currently using, and the fact that at one particular time, one single BGP router can use only one route to one destination[1]. However, given that the Internet is changing and the current facts may change later, we would like to present this fact as an assumption.

Assumption 1: One single BGP router can only advertise to its peers(iBGP or eBGP peer) one single(same) route to one destination.

Assumption 1 is true in the Internet to the best of our knowledge [1], and is the basis of the enhanced consistency assertions for BGP.

The enhanced assertions for BGP require an AS that are doing traffic engineering on the routes to one destination to attach additional information to the route when advertising it to neighbor ASes. This additional information contains the ID of Entry Router, the router who receives the route from eBGP peers or originates the route itself.

In the case that routes to the same destination are received from multiple entry routers, the AS could be divided into multiple **logical ASes**, each of which could be uniquely identified in the Internet by the tuple (ASN, $EntryRouterID$).

All the BGP routers in AS ASN that choose to select as the best the route whose Entry RouterID is RID belong to the logical AS <ASN,RID>. In the case that there are no traffic engineering on the route, no Entry RouterID is attached and the logical AS is the same as the real AS, and should be viewed as a different logical AS when compared to logical AS that has the same AS number but has Entry RouterID attached. The logical AS is defined in the context of one particular destination, therefore there are no corresponding physical divisions of the logical ASes and for different destinations there may be different divisions of the logical ASes.

Since one BGP router can only advertise to its iBGP peers one route to a destination according to Assumption 1, all the routers within a logical AS can only use and advertise to their peers one single route to one destination. Here we can find that the logical AS concept fits well into the Simple Path Vector Protocol. Therefore, the way to enhance the theorem and assertions in Section III, is naturally to replace the real AS with the logical AS in the theorem and assertions.

Theorem 2: Assume Assumption 1 is true and every AS attach the Entry Router-ID to the route if doing traffic engineering on the route considered. Let $path(N_1, d) = (N_1, P_1, P_2, ..., P_n)$ be the path from N_1 to destination d and let $path(N_2, d) = (N_2, Q_1, Q_2, ..., Q_m)$ be the path from N_2 to d. $N_1, N_2, P_i(i = 1, ..., n)$, and $Q_i(j = 1, ...m)$ are all logical ASes. If $Q_i = P_j$ for some i and j, then either $n - i = m - j$ and $Q_{i+k} = P_{j+k}$ for all $1 \leq k \leq n - j$, or at least one of the paths is invalid.

- **Proof:**

Replace the real AS in Theorem 1 Proof with logical AS, this Theorem holds.

The enhanced version of the Route Change Assertion can be obtained by simply replacing the real AS in Route Change Assertion in Section III-C. The enhanced version of the Route Withdrawal Assertion, however, will be discussed in Section IV-B after we address the policy withdrawal problem.

One alternative of Entry RouterID in the the logical AS solution is that a Exit Router, who advertises to its eBGP peers the route to one destination, attach a Exit RouterID to the route. This approach acutually need an assumption even weaker than Assumption 1.

Assumption 2: One single BGP router can only advertise to its eBGP peers one single(same) route to one destination.

Assumption 2 is true in the current Internet [1], and based on it we could develop similiar enhanced theorems and assertions for BGP. Both of these two alternatives work in the sense that no valid route is marked infeasible, but due to strict checking of logical AS, they may both miss some chances to invalidate some invalid backup routes in some scenarios. We are currently investigating the advantages of these two altanatives, thus using Entry RouterID is just a tentative solution and we may change to Exit RouterID solution if necessary.

A.2 Implementation of Logical AS Solution

In order to implement the Logical AS Solution in a backwards compatible way, we extend the BGP protocol by defining and using new community attributes [14]. The community attribute is 32 bit value, normally associated with route advertisements and used to convey routing policy information. For example, including a community value of 0xFFFFFF02 with a route advertisement indicates that the route should not be advertised to other peers.

To implement logical AS solution, each Entry router in the AS that is doing traffic engineering should attach an Entry RouterID community attribute to the route, whose format is defined as the following.

ASN	F	E=0	RID

where ASN is the 2-Byte AS number of local AS, F is the 1-Byte flag which will take a specific value to indicate that this community attribute will include the Entry RouterID Information. If the 1-bit extension flag E=0, the left 7-bit RID field will be the RouterID of the router who is creating this community attribute. When 7-bit RID field is not enough to contain the Router-ID, two consecutive Entry Router-ID community attributes in the following format.

ASN	F	E=1	H-RID
ASN	F	E=0	L-RID

The first community attribute with Flag E=1 gives the higher 7 bits of the RouteID, and the second one with E=0 gives the lower 7 bits of the RouterID.Therefore, at most 16,256 Router IDs could be assigned by an AS, which we believe is enough.

When a BGP router receives an route with Entry RouterID community attribute and selects the route, it should not modify this attribute and should propagate it when advertising the route to eBGP peers.

B. Failure Withdrawals and Policy Withdrawals

There are two distinct causes for a BGP withdrawal message. A **failure withdrawal** occurs if an AS has lost it route to the destination. Failure withdrawals can occur due to the failure of a route imported from IGP, the close of the peering session with the upstream peer advertising the route, or a withdrawal received from the upstream peer. In all of these of cases, the existing route to the destination is no longer valid and the failure withdrawal conveys topology information that can be used to invalidate other routes.

A **policy withdrawal** occurs if a change in route policy causes an AS to stop advertising a route to some of its neighbors. In this case, the upstream router still has its existing route to the destination but the upstream router no longer make this route available to some peer(s). To determine whether a backup route is feasible, one must distinguish between failure withdrawals, which convey new topology information, and policy withdrawals, which must not be used to invalidate backup routes.

The BGP specification does not differentiate a failure withdrawal from a policy withdrawal and the BGP UPDATE message format must be modified to indicate the withdrawal type. The modified UPDATE message must also remain compatible with the standard BGP UPDATE message so that our approach can be incrementally deployed. A simple 1-bit withdrawal type flag would have achieved this, but there are no reserved bits left in the BGP UPDATE message. Instead, the BGP community attribute[14] is used in a novel way. In our approach, a router signals an a failure withdrawal by including a **failure withdrawal community attribute** in the BGP UPDATE message.

Our approach reserves the community value 0x88888888 and associates this value with withdrawn routes. To indicate a failure withdrawal, an UPDATE message contains the route to be withdrawn and a community attribute with value 0x88888888. If the failure withdrawal community attribute is not present, then any withdrawn routes are assumed to be policy withdrawals.

This approach is compatible with existing implementations. If a router does not implement our approach, it will not include the failure community attribute and its withdrawals will be always be considered policy withdrawals and will processed according to traditional BGP rules. Routers which do not implement our approach are also protected from receiving the failure withdrawal community attribute. At the start of a BGP connection, the BGP capability negotiation process[15] is used to signal the use of our new approach. The failure withdrawal community attribute is only sent to those routers who negotiate to receive it.

To further insure backwards compatibility, route advertisements are never included in failure withdrawal UPDATES. In standard BGP, a single UPDATE message may contain withdrawals for some destinations and route advertisements for other destinations. The community attribute is normally applied to all the route advertisements in an update. A router that has not implemented our approach **and has incorrectly** negotiated BGP capabilities would apply the failure withdrawal community value to any advertised routes in the UPDATE. The failure withdrawal community value is not be meaningful to the advertised routes, but any unknown community attribute is simply associated with the advertised routes and would propagate along with the future advertisements for these routes. To avoid this scenario, failure withdrawals are always be sent in an UP-

DATE message that contains only the failure withdrawn routes.

In summary, the failure withdrawal community attribute is sent only to peers that have negotiated this capability and failure withdrawal UPDATES consist of only an withdrawn routes part and the failure withdrawal community attribute. This format is not a standard one since attributes are typically associated with advertised routes, but according to the latest BGP draft[22], an UPDATE with only withdrawn routes and valid path attributes shall be viewed as a valid UPDATE.

The enhanced version of the Route Withdrawal Assertion can be obtained by limiting assertions only apply to failure withdrawal, and replacing the real AS in the Route Withdrawal Assertion in Section III-B.

C. Addressing the AS Partitions

In some scenarios, an AS may become partitioned into several parts due to failure of internal links. As a result, routers in different partitions could choose different routes to one destination, or some routers lose the route to the destination while others still have the route. This also makes BGP not fit into the Simple Path Vector Protocol, although in Internet AS partitions should be rare and be fixed quickly.

Basically, we want to make sure that our assertions should not apply to any withdrawals or new route changes that resulted from AS Partition. In the case when an AS is doing traffic engineering on the route to one destination, and one of its routers loses a route due to the loss of the connectivity (AS Partition)to the entry router of the route, the withdrawal it sends should also be set as a policy withdrawal, since other routers in the same logical AS may still have the route. When a router advertises a new route due to the loss of the connectivity to the entry router of the previous route, the new route should have been already attached Entry Router-ID of the new route. On the other hand, in the case when an AS is not doing traffic engineering on the route to one destination, and one of its routers loses a route due to AS partition,a policy withdrawal is sent. When a router advertises a new route due to the loss of the connectivity to the entry router of the previous route, the router should attach the Entry RouterID of the new route. The local logical AS in the new route will be treated as a different one from the logical AS (no Entry RouterID attached) in former route, thus will not invalidate the former routes sent by other routers in the same AS.

V. Implementation in BGP

This section shows how the assertions from Section III and Section IV can be be implemented in BGP. The basic BGP Routing process is discussed below and then the implementation of each assertion is discussed. An example of how the Route Withdrawal Assertion works is given at the end of this section.

A. The BGP Routing Process

Neighboring BGP routers (BGP peers) exchange messages using long lasting TCP[16] connections. The use of TCP insures the reliable delivery of messages and periodic BGP KEEPALIVE messages verify that the TCP connection is functioning properly. To announce a route to some destination, a BGP router sends an UPDATE message. A route advertisement UPDATE message includes the destination network (NLRI) and a number of **path attributes**, most notably the AS Path attribute that lists the path of Autonomous Systems used to reach the destination. Additional UPDATE messages for this destination are sent only if the routes attributes change[1] or if the route is withdrawn. BGP withdrawal messages are sent by listing the destination in the withdrawn routes section of an UPDATE message.

A BGP router records the routes received from each of its peers in a table. The table for peer p is denoted $AdjRIB[p]$ and entry $AdjRIB[p][d]$ indicates the route peer p uses to reach destination d. After receiving a route advertisement for destination d (or a withdrawal for d), the corresponding entry in the $AdjRIB$ table is updated and the BGP Decision Process is run to determine the new route to d. For all peers p_i, the router calculates a preference for $AdjRIB[p_i][d]$. If no feasible routes to d are available, the router will declare the destination unreachable. Otherwise, the best route to d is installed in the router's routing table. If the BGP Decision Process resulted in a new route to d (or if d has become unreachable), the router applies its routing policies and sends the appropriate UPDATE messages listing the new route to d.

In order to constrain the amount of routing traffic, the BGP standard includes a mechanism to control the frequency of route advertisements [1]. The BGP standard requires a minimum amount of time must elapse between route advertisements for a particular destination. The minimum time period, denoted MinRouteAdver, is recommended to be 30 seconds with a random jitter. The assumption behind this approach is that a changed route is likely to change again in a brief interval [8]. Waiting for MinRouteAdver seconds allows BGP routers to ”pack” consecutive updates and diminishes the global load on the BGP infrastructure[8]. In order to avoid long-lived black

[1] A route refresh capability[17] has been added so a router may request the re-advertisement of a route.

holes, MinRouteAdver does not apply to withdrawals[1].

Although the MinRouteAdver should be implemented on a (destination, peer) basis, it is believed this may add unwarranted overhead[1]. Therefore the BGP standard also states that a per peer basis implementation is acceptable, provided that the transmission of two consecutive updates for the same destination will always be at least MinRouteAdver seconds apart and will also be upper bounded by some constant value, denoted MaxRouteAdver. In most implementations, a 30-second timer is applied to each peer, and the updates (except for explicit withdrawals) will be sent out only after the timer expires. Thus if a route changes multiple times during the 30-second period, only the last change should be announced.

B. Implementing the Route Change Assertion

In order to implement the route change assertion, the UPDATE processing and route selection algorithms must be changed. A new route advertisement must be consistent with the routes stored in the AdjRIB tables. When a route advertisement arrives, the AdjRIB tables are checked for consistency and routes that fail the Route Change Assertion are marked as infeasible. These infeasible routes can not be selected as the preferred route to the destination.

After receiving a route advertisement for destination d from peer p, the $AdjRIB[p][d]$ table entry is updated and a new table entry, denoted $conflicts[p][d]$ is initially set to $NULL$. The $conflicts$ entry will indicate whether the Route Change Assertion holds for this route. If $conflicts[p][d] = NULL$, the this route does not conflict with the routes from any other peer. Otherwise, $conflicts[p][d]$ lists the AS numbers of the conflicting peers.

In traditional BGP, each $AdjRIB[p_i][d]$ entry is assigned a preference. In our modified version, each $AdjRIB[p_i][d]$ is also checked for feasibility and $conflicts[p_i][d]$ is updated accordingly. In the discussion below, the **logical** AS(as defined in Section IV) of peer p (p_i) is denoted as $AS(p_{change})$ ($AS(p_i)$) and the **logical** AS path associated with the route from p (p_i) to d is denoted as $aspath(p_{change}, d)$ ($aspath(p_i, d)$) (respectively).

First, the new route is used to check the feasibility of existing $AdjRIB[p_i][d]$ entries. If $AS(p_{change}) \in aspath(p_i, d)$ and $aspath(p_i, d)$ does not end with $aspath(p_{change}, d)$, then the $AS(p_{change})$ is added to the set $conflicts[p_i][d]$. If $aspath(p_i, d)$ does end with $aspath(p_{change}, d)$ and $AS(p_{change} \in conflicts[p_i][d]$, then $AS(p_{change})$ is removed from $conflicts[p_i][d]$.

Second, the existing $AdjRIB[p_i][d]$ entries are used to check the feasibility of the new route. If $AS(p_i) \in aspath(p_{change}, d)$ and $aspath(p_{change}, d)$ does not end with $aspath(p_i, d)$, then $AS(p_{change})$ is added to the set $conflicts[p][d]$.

peer	AS Path	Conflicts
p2129	2129	NULL
p5696	5696, 2129	NULL
p1	1, 5696, 2129	NULL

Fig. 3. Initial $AdjRIB$ Values

Finally, for the peers with $conflicts[p_i][d] = NULL$, the route with the best preference is selected as the route to d and the BGP process continues in the normal way.

C. Implementing the Route Withdrawal Assertion

After receiving a withdrawal for destination d from peer p_{lost}, the $AdjRIB[p_{lost}][d]$ table entry is cleared. In the discussion below, the **logical** AS number of peer p_{lost} is denoted as $AS(p_{lost})$ and the **logical** AS path associated with the route from (p_i) to d is denoted as $aspath(p_i, d)$.

If the withdrawal is a policy withdrawal, then p_{lost} has stopped reporting its route to d and no information about p_{lost}'s route to d should be inferred. $AS(p_{lost})$ is removed from any $conflicts[p_i][d]$ that contains it.

If the withdrawal is a failure withdrawal, then $AS(p_{lost})$ has lost its route to d. If $AS(p_{lost})$ appears in any $aspath(p_i, d)$, then $AS(p_{lost})$ is added to $conflicts[p_i][d]$.

This implementation was deployed and tested in a network containing both modified routers and standard BGP routers. Substantial gains in route convergence were achieved and the results are discussed in Section VI.

D. Example

Figure 1 from Section I showed an example of slow BGP convergence. In that example, AS 2117 learned a route to d from AS 2129. When AS 2129 sent a withdrawal for d, AS 2117 first tried the route with AS path 5696, 2129. When that route was withdrawn, AS 2117 tried the route with AS path 1 5696, 2129 and so on. A slightly simplified version of this example is given below to illustrate how our modified BGP implementation would improve BGP route convergence. This example does not involve traffic engineering or AS partition, but does consider the policy withdrawal.

For simplicity, let r be a router in AS 2117 and assume that r has three peers: peer p2129 in AS 2129, peer p5696 in AS 5696, and peer p1 in AS 1. Initially, r uses p2129 to reach destination d and the corresponding $AdjRIB$ entries are shown in Figure 3.

Now suppose that p2129 sends a failure withdrawal for d. This failure withdrawal creates conflicts for the (invalid)

peer	AS Path	Conflicts
p2129	NULL	NULL
p5696	5696, 2129	2129
p1	1, 5696, 2129	2129

Fig. 4. $AdjRIB$ Values After a Failure Withdrawal By AS 2129

peer	AS Path	Conflicts
p2129	NULL	NULL
p5696	5696, 2129	NULL
p1	1, 5696, 2129	NULL

Fig. 5. $AdjRIB$ Values After a Policy Withdrawal By AS 2129

backup routes since both of the routes rely on AS 2129. The resulting $AdjRIB$ table is shown in Figure 4. Since the two backup routes contain conflicts, neither can be selected and router r declares d to be unreachable. Since AS 2129 route to d has failed, eventually AS 5696 and AS 1 will withdraw their routes to d and the conflicts will be removed. With only route change (current path to unreachable) and virtually no delay, router r has correctly determined that d is unreachable.

Now suppose that AS 2129 implements a policy change and no longer advertises the route d to AS 2117. In this case AS 2129 can still reach d, but the link between AS 2117 to AS 2129 can no longer be used to reach d. The policy withdrawal will not generate any conflicts and router r can switch to the (valid) backup route via peer p5696. The resulting $AdjRIB$ table is shown in Figure 5.

VI. Testbed Deployment and Simulation Results

To test the BGP convergence assertions, the assertions were implemented in MRTD[18] and deployed in the FNI-ISC project's BGP testbed. In addition, simulation results were used to explore large topologies that could not be created in the testbed. The results show a substantial reduction in both convergence time and number of updates exchanged.

A. Deployment in the Testbed

We built a testbed whose topology is shown Figure 6. In this topology, each router belongs to a different AS. Routers A, B, C and D use a modified MRTD that implements our approach. Routers H, I, and J run the original MRTD. Currently MRTD applies the MinRouteAdver timer to the withdrawals and use sender-based loop detection. Although sender-based loop detection is known to speed up convergence, most commercial routers only perform loop detection upon the receipt of a route update[8]. In order to clearly compare our approach with results achieved using commercial routers, we turned off the MRTD sender-based loop detection and adjusted the MinRouteAdver timer so that it no longer applies to withdrawal messages. These steps were done for all routers in the testbed.

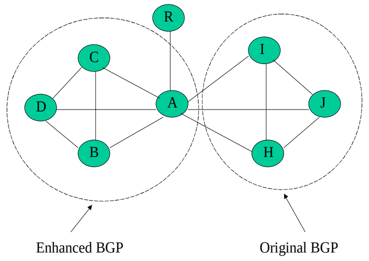

Fig. 6. Experiment Testbed Topology

	BGP	Enhanced BGP
Convergence time:	30.3s	0.3s
Update Messages:	24	12

Fig. 7. Testbed Results for Failure Withdrawals

In our experiments, R will advertise a route to a destination r and will generate the route failures and repairs at the a four-minute periodicity: 2 minutes after the route to r is announced, it will be withdrawn; 2 minutes later it will be announced again. A similar experiment is conducted on the route change: the route to r that R advertises will oscillate between a short route and a long backup route at a four-minute periodicity. The results of both experiments are summarized in the Figure 7 and Figure 8. Analysis similar to section 4.4 can show why our approach reduces the update messages. As soon as B, C and D receive the failure withdrawal from A, they will find that all the backup routes contain conflicts and declare the destination as unreachable. Similarly, as soon as B, C, and D receive the route change from A, they will find that all the backup routes contain conflicts and conclude the best routes are the new routes they receive.

Because the invalid routes are immediately marked as infeasible after receiving the route withdrawal or route change, the 30-second MinRouteAdver timer does not affect the enhanced BGP in this experiment and results in the substantial reduce of the convergence time.

In order to further test the compatibility of our modified BGP and other BGP routers, we deployed the modified

	BGP	Enhanced BGP
Convergence time:	64.9s	0.1s
Update Messages:	24	12

Fig. 8. Testbed Results for Implicit Withdrawals

topo. \ degree	1	$2-5$	$6-10$	>10	avg. degree
1	4	9	7	5	6.16
2	4	9	4	8	6.88
3	2	10	5	8	7.36
4	2	12	2	9	6.64
5	6	10	8	1	4.56

Fig. 9. Topology statistics for network with 25 nodes

MRTD on the CAIRN Testbed[19]. The CAIRN testbed peers with the research Internet and no deployment problems were encountered during a week long test.

B. Simulation Results

To enhance and complement the testbed experiments, simulations were conducted on relative large networks. The network topologies being used in the simulations were derived from the BGP routing table of the Oregon Route Views server[13], dated as 04/02/2001. The process to generate the topology is shown in the following example. To generate a 25-AS topology, we randomly select 25 ASes from the routing table which can construct a connected subgraph of the Internet. Five different 25-AS topologies are automatically generated and used for each round of simulation. The average convergence time and the average number of messages transmitted during the convergence are calculated. The variance of the results remains very small. Figure 9 shows some topological statistics of the five 25-AS topologies we used in the simulation. Simulations on networks with other sizes are conducted similarly.

The simulation uses the tools developed by SSFNET project [20], which has a built-in BGP simulator and is suitable to simulate very large networks. The link delay parameter is configured to be 0.23 second for the current simulation, while we plan to randomize this parameter in the future work. One particular AS in the network will first advertise a prefix to its peers. After the whole network comes to be stable (no more messages exchanging), the origin AS withdraws the prefix. The convergence time is calculated as the interval between the time when the prefix is withdrawn and the time when the network returns to the stable state. Both the original BGP simulator and the simulator that implemented our approach run on the same topologies we generate as described above. The comparison of the results for original BGP and enhanced BGP are shown in Figure 10, 11 and 12. Please note that there are only one router in each AS, thus no traffic engineering or AS partition.

AS #	Convergence time		Num of messages	
	original	enhanced	original	enhanced
10	1.012843	0.782688	41.6	36.2
15	54.887645	1.197116	248	198.6
20	126.440259	1.473362	733.6	516.2
25	223.716732	1.335263	1599.8	855.8
31	404.026594	1.243178	3747	1455.4
37	456.210823	1.289237	4742.2	1617.2
43	551.763434	1.289164	7390	2312.6
49	733.211137	1.335274	12274.4	3311.4
55	815.027126	1.243204	15106.8	3765.2
61	823.810093	1.335212	17096	4193

Fig. 10. Comparison of original BGP with enhanced BGP

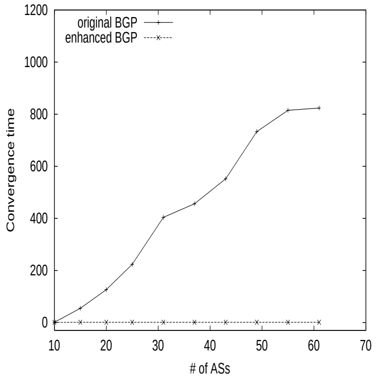

Fig. 11. Comparison of convergence time

As shown in the figures, both the convergence time and the number of messages are reduced substantially in the enhanced BGP. As the network size increases, more backup routes become available. Because original BGP explores all the backup routes before convergence and has to wait for the MinRouteAdver timer to expire, the convergence time and the number of update messages for original BGP grows almost linearly as the network size increases. The network diameter also increases a little as the network size increases, resulting longer propagation delay of the UPDATE messages. However, compared to 30-second MinRouteAdver timer, the effect of network diameter increase on convergence time and number of update messages is small.

On the other hand, the convergence for the enhanced BGP is very fast regardless of the network size increasing. Please note that it is possible that a failure far away from the local router results in invalidating all the backup routes of the local router, but the withdrawal received from one peer doesnot necessarily invalidate all the backup routes. This could happens when the peer AS may not appear in all of the backup routes. As a result, an invalid backup route is selected, advertised, and further propagated, increasing the intermediate update message number and convergence time. However, Figure 12 shows that majority of invalid

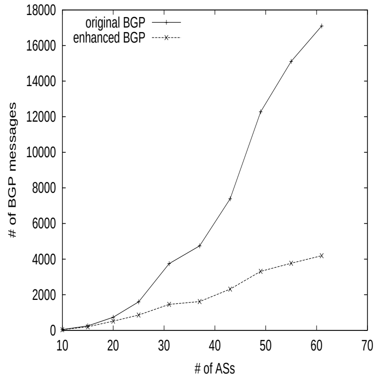

Fig. 12. Comparison of number of messages

backup routes are marked as infeasible immediately after the failure. Thus the total number of intermediate invalid UPDATE messages are reduced greatly comparing to the original BGP. The convergence time is almost the time to propagate the withdrawal from the originator to the farthest AS. Thus highly related to the network diameter. Given that the network diameter does not change much when the network size increases, (probably because of the hierarchy structure of Internet) the convergence time remains low in the enhanced BGP.

Simulations on the route change, as well as simulations considering traffic engineering will be conducted in the future.

VII. Summary

Instead of blindly accepting all BGP UPDATEs, our basic approach is to let a router check route consistency using the information it has learned from previous updates and from other neighbors. In particular, in this work we used the information provided in the AS path to define route consistency assertions and used these assertions to identify infeasible routes. By taking this broader view and exploiting the relationships between routes, we were able to reduce the BGP convergence time after route changes substantially as shown in our simulation results.

In this paper we showed how to implement our assertions in BGP, we also developed a backwards compatible implementation to verify our design by modifying the MRTD routing software. We demonstrated that, both through simulation and via measurement over our BGP testbed, our approach reduces reduce the BGP convergence time by up to 2 3 orders of magnitude.

Future work includes simulations on the effects of route change assertions, deriving traffic engineering information from Oregon Route Views Server and include this information into simulations, and conduct experiments on testbeds with larger and more complex topologies. More analysis and simulations will also be conducted to better understand the differences between the Entry RouterID or Exit RouterID approaches.

VIII. Acknowledgements

We would like to express our appreciation to the people who reviewed an early version of this paper and pointed the need to accommodate traffic engineering while solving the routing convergence problem.

References

[1] Y. Rekhter and T. Li, “Border Gateway Protocol 4,” RFC 1771, SRI Network Information Center, July 1995.
[2] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian, “Delayed Internet Routing Convergence,” in *Proceedings of ACM Sigcomm*, Aug. 2000.
[3] C. Labovitz, “Delayed Internet Routing Convergence,” http://www.research.microsoft.com/ labovit/, Aug. 2000.
[4] C. Labovitz, G. Malan, and F. Jahanian, “Internet Routing Instability,” in *Proceedings of ACM Sigcomm*, Sept. 1997.
[5] C. Labovitz, A. Ahuja, and F. Jahanian, “Experimental Study of Internet Stability and Wide-Area Network Failures,” in *Proceedings of FTCS99*, June 1999.
[6] C. Labovitz, R. Wattenhofer, S. Venkatachary, and A. Ahuja, “The Impact of Internet Policy and Topology on Delayed Routing Convergence,” in *Proceedings of IEEE INFOCOMM*, Apr. 2001.
[7] Gary Malkin, “Routing Information Protocol Version 2,” RFC 2453, SRI Network Information Center, Nov. 1998.
[8] Christian Huitema, *Routing in the Internet*, Prentice-Hall, 2000.
[9] J.J Garcia-Lunes-Aceves and S. Murthy, “A Loop-Free Path-Finding Alogirthm: Specification, Verification and Complexity,” in *Proceedings of the IEEE INFOCOM*, Apr. 1995.
[10] Pierre A. Humblet, “Another Adaptive Distributed Shortest Path Algorithm,” *IEEE Transactions on Communications*, vol. 39, no. 6, pp. 999–1003, 1991.
[11] Z. Xu, S. Dai, and J.J. Garcia-Luna-Aceves, “A More Efficient Distance Vector Routing Algorithm,” in *Proceedings of IEEE MILCOM*, Nov. 1997.
[12] Y. Afek and A. Bremler, “Self-Stabilizing Unidirectional Netowrk Alogirhtms by Power-Supply,” in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, Jan. 1997.
[13] “The Route Views Project,” http://www.antc.uoregon.edu/route-views/.
[14] R. Chandra, P. Traina, and T. Li, “BGP Communities Attribute,” RFC 1997, SRI Network Information Center, Aug. 1996.
[15] R. Chandra and J. Scudder, “Capabilities Advertisement with BGP-4,” RFC 2842, SRI Network Information Center, May 2000.
[16] Jon Postel, “Transmission Control Protocol,” RFC 793, SRI Network Information Center, Sept. 1981.
[17] E. Chen, “Route Refresh Capability for BGP-4,” RFC 2918, SRI Network Information Center, Sept. 2000.
[18] “MRTD: The Multi-Threaded Routing Toolkit,” http://www.mrtd.net.
[19] “The CAIRN Testbed,” http://www.cairn.net.
[20] “The SSFNET Project,” http://www.ssfnet.org.