October 2008

Reliable Interdomain Routing Through Multiple Complementary Routing Processes

Yong Liao
University of Massachusetts, Amherst

Lixin Gao
University of Massachusetts, Amherst

Roch A. Guérin
University of Pennsylvania, guerin@acm.org

Zhi-Li Zhang
University of Minnesota

Proc. ACM ReArch'08 Workshop, December 2008, Madrid, Spain.

Reliable Interdomain Routing Through Multiple Complementary Routing Processes

Abstract
The Internet inter-domain routing protocol, BGP, experiences frequent routing disruptions such as transient routing loops or loss of connectivity. The goal of this paper is to address this issue while preserving BGP's benefits in terms of operational maturity and flexibility in accommodating diverse policies. In realizing this goal, we apply to inter-domain routing a common concept in the design of highly reliable systems, namely, the use of redundancy, which we introduce in a manner that maximizes compatibility with the existing BGP protocol. The basic idea is to run several, mostly unchanged BGP processes that compute complementary routes, so that in the presence of network instabilities a working path remains available to any destination. The paper outlines the design of this approach and compares it to previously proposed alternatives. The benefits of the scheme are demonstrated using actual BGP data and realistic simulations.

Keywords
Internet, routing, inter-domain, reliability

Comments
Proc. ACM ReArch'08 Workshop, December 2008, Madrid, Spain.

Reliable Interdomain Routing Through Multiple Complementary Routing Processes

Yong Liao[†], Lixin Gao[†], Roch Guerin[§], Zhi-Li Zhang[††]

ECE Department[†]
University of Massachusetts
Amherst, MA 01003, USA
(yliao,lgao)@ecs.umass.edu

ESE Department[§]
University of Pennsylvania
Philadelphia, PA 19104, USA
guerin@ee.upenn.edu

CSE Department[††]
University of Minnesota
Minneapolis, MN 55416, USA
zhzhang@cs.umn.edu

ABSTRACT

The Internet inter-domain routing protocol, BGP, experiences frequent routing disruptions such as transient routing loops or loss of connectivity. The goal of this paper is to address this issue while preserving BGP's benefits in terms of operational maturity and flexibility in accommodating diverse policies. In realizing this goal, we apply to inter-domain routing a common concept in the design of highly reliable systems, namely, the use of redundancy, which we introduce in a manner that maximizes compatibility with the existing BGP protocol. The basic idea is to run several, mostly unchanged BGP processes that compute *complementary* routes, so that in the presence of network instabilities a working path remains available to any destination. The paper outlines the design of this approach and compares it to previously proposed alternatives. The benefits of the scheme are demonstrated using actual BGP data and realistic simulations.

1. INTRODUCTION

With our growing reliance on the Internet, its reliability, in particular that of its routing system, has become ever more important. Fulfilling this need has, however, proven challenging, because the distributed nature of Internet routing decisions introduces unavoidable latency in reacting to network changes. This is particularly evident in inter-domain routing, where the shortcomings of the *de facto* standard routing protocol, BGP, are well known [10]. For instance, BGP may take as long as 30 minutes to converge after certain routing events [13], during which *transient* routing loops and failures frequently occur. Recent measurement studies [9,18] have shown that 55%∼85% of short-lived routing failures occur during BGP convergence, and that transient loops account for up to 90% of packet losses.

There have been numerous proposals to address this challenge. One approach is to speed-up BGP convergence; hence limiting the duration and thereby impact of transient routing loops and failures [1,3,15,17]. This can in part be realized by including *root cause information* (RCI) in routing updates, so as to speed up the removal of obsolete routes affected by a common failure. Another approach is to compute backup paths that can supplement the "best path" selected by BGP. The R-BGP protocol [11] is one such solution, but it also requires additional information in the form of RCI to compute *good* backup paths. Such solutions, therefore, introduce substantial overhead as well as modification to the operations or behavior of BGP which can affect the odds of successful deployment.

This paper shares the goal of those earlier proposals, i.e., improving the reliability of inter-domain routing, but it seeks to realize it while preserving the wealth of operational knowledge and expertise embedded in BGP. This rules out approaches that call for adding complex new information to BGP or significant modification of its behavior/operations (e.g., as embodied in RCI). The basic idea behind our approach is to use multiple (two) slightly extended BGP processes instead of one with extensive modifications. The two BGP processes operate and select paths nearly as standard BGP process, but exploit the AS-level path diversity of the Internet to compute *complementary* paths, i.e., paths that are not affected by the same sets of events.

Although this sounds straightforward, computing disjoint *policy compliant* paths, a key requirement for inter-domain routing, is notoriously hard [4], and even more so if it is to be accomplished under the constraint of minimal changes to BGP. In tackling this problem, we first identify possible simplifications brought about by the Internet structure and common routing policies. In particular, we establish that complementary routing solutions can be obtained by focusing only on the *"down-*

ReArch'08, December 9, 2008, Madrid, SPAIN

hill" portion of paths, i.e., the segments that extend from provider ASes to customer ASes towards the destination, if each AS is modeled as *one node* in AS graph. Once complementary routes are available, it remains to specify how to *use* them, and in particular identify which one is free of problems and should be used at any one time. We propose a simple approach to this problem and argue its effectiveness.

The bulk of the paper is devoted to describing the proposed scheme, articulating how it meets our original goals, i.e., improving BGP's reliability without significantly affecting its current operational characteristics. and comparing its efficacy to that of earlier proposals. The paper is organized as follows. Section 2 provides background information on inter-domain routing. Section 3 discusses the basic design principles behind our multi-process routing scheme. Details on its design and realization are given in Sections 4 and 5. Section 6 is devoted to an extensive evaluation of the scheme and its performance. Section 7 concludes the paper.

2. BACKGROUND

In this section, we first review Internet routing and transient BGP problems, and then introduce our classification of routing events used throughout this paper.

2.1 BGP and Transient Routing Problems

BGP is a policy-based path vector protocol which has been shown to converge under specific policy constraints [8]. In practice, neighboring ASes commonly engage in bilateral agreements, also called *AS relationships*, which determine and constrain their routing policies. The two most common are *customer-provider* relationship and *peer-peer* relationship [5]. Because of the economic interests associated with these relationships, ASes typically follow two common routing policies, *prefer-customer* - whenever they are available, -an AS always selects customer routes (routes learned from a customer), and *valley-free* –an AS does not advertise provider/peer routes (learned from a provider/peer) to other providers/peers. Assuming that the customer-provider relationships between ASes are *acyclic*[1], which holds in practice, the BGP protocol has been shown to be*safe*, i.e., it always converges to a stable set of routes if every AS adopts these two policies [6]. *This paper assumes that those conditions are satisfied.*

The safeness of BGP not withstanding, it only implies that routing "eventually" converges. However, while converging, ASes can experience *transient* loss of reachability, commonly referred to as *transient routing failures* [18]. Moreover, inconsistency in routing information across ASes during convergence can also result in *transient routing loops*. Eliminating such incidents, to the extent possible, is one of the goals of this paper.

[1]Namely, the provider of any AS cannot be a customer of that AS' customers, or a customer of a customer, and so on.

2.2 Routing Events

A first step in realizing this goal is to characterize the set of events that trigger BGP changes. Example of such "routing events" include link failure/recovery, BGP session reset, router crash/recovery, policy changes, etc. We group routing events into three classes: a *route withdrawal event* is an event that triggers one or more ASes to withdraw, explicitly or implicitly, (in case of failure), withdraw one or more routes; a *route addition event* is an event that triggers one or more ASes to announce a new route; and a *route change event* is an event that triggers an AS to announce route updates. Failure or policy changes are common causes for the first and third types of events, while recoveries and possibly policy changes are usually behind the second type of events. Our focus is on improving the resiliency of inter-domain routing against disruptions associated with a *single* routing event, i.e., eliminate or minimize associated packet forwarding disruptions.

3. MULTIPLE ROUTING PROCESSES

3.1 Basic Idea

We seek to provide robustness against single routing events, by providing each AS with two distinct sets of routes that are *complementary*. This is realized by having two slightly modified BGP routing processes running in each AS, i.e., a *red* process and a *blue* process. The *red* and *blue* paths they compute seek to satisfy a key property, namely, *node disjointness*, i.e., not share AS nodes except for the source and destination (recall that our focus is on AS-level paths, with each AS a "node" in the path). This ensures that they are not affected by the same sets of events.

Each routing process is basically a slightly modified BGP process that relies on the standard BGP path selection process. BGP messages are unchanged, except for the addition of two new path attributes. One is to assist in coordinating the red and blue processes in an AS[2]; the other is used to determine which process has stable paths in the presence of routing events.

Differentiating the red and blue routing processes can be realized, for example, through distinct TCP ports. Alternatively, a single process handling two routing instances can be used. Both are semantically identical, and the use of the term "process" is meant primarily as an abstraction helpful in describing the solution.

3.2 Downhill Node Disjointness

Having stated our primary goal, computing node disjoint paths, we pause to point out that realizing it while preserving BGP's distributed, policy-based computations is non-trivial [4]. Fortunately, our path disjointness constraint can be relaxed to disjointness only in the

[2]There is no coordination between processes of different colors in different ASes. The red (or blue) process selects path among those announced by the red (or blue) processes of neighboring ASes.

"downhill" section. Because of the "valley-free" property, and AS path typically consists of an *uphill portion* followed by a *downhill portion,* which consists of a sequence of provider-to-customer links, together with the ASes at the two ends of each link (the other part of the AS path, if it exists, is the *uphill portion*). The ability to relax the node disjointness constraints of paths to the downhill portion is based on Lemmas 3.1 and 3.2.

Note that we model an AS as *a single node* in AS graph and consider eBGP only. Lemma 3.1 states that route addition and route change events will not trigger transient problems. This is because no ASes lose routes after such routing events, so that transient problems can be altogether avoided. When considering iBGP, someone getting a new route can cause others to lose routes, not to mention the complicated interactions because of the MED attribute. However, the recently proposed centralized routing [2,7] get around the iBGP problems. A complete proof of the Lemma is in [14].

LEMMA 3.1. *No transient routing loops or failures occur after route change or route addition events.*

Lemma 3.2 considers route withdrawal events that commonly give rise to transient BGP problems. It establishes that under the constraints of common routing policies, network events in the uphill portion of a path will not trigger transient loops or failures during BGP convergence. In other words, a link or AS failure in a higher tier AS (provider) does not create transient failures or loops at an AS while its BGP process adapts to the changes. More formally (see again [14] for a proof)

LEMMA 3.2. *A route withdrawal event in the uphill portion of an AS path to a destination does not produce transient routing loops or failures during BGP convergence.*

The above two lemmas establish that routes computed by two routing processes and only disjoint in their downhill portions still offer complementary routing choices in the face of any routing event. Based on this insight, we proceed to present the design of our multi-process routing scheme, the *SelecTive Announcement Multi-Process routing protocol (STAMP).*

4. THE STAMP PROTOCOL

In this section, we first introduce STAMP and then formally establish its properties.

4.1 Protocol Overview

In order to realize complementary route selection between its red and blue processes, the STAMP protocol imposes minimal coordination between the two processes by way of selective route announcements, i.e., it constrains the ASes to which red and blue paths are announced[3]. Because disjointness is only required in the downhill portion, this selectivity only applies to providers and not to announcements to peers and customers, i.e., announcements of both blue and red paths proceed freely to all peers and customers. The goal is then to have blue announcements propagate along one set of providers, and red announcements along a disjoint set of providers. The challenge is to realize this in a distributed manner and with simple rules that can be easily incorporated into BGP. One option is to give one color, say, blue, strict priority over the other, so that an AS preferentially advertises blue routes to its providers in case it received them on both processes. Such an approach is, however, overly restrictive and can severely affect the odds of all ASes successfully acquiring a red path to all prefixes. In order to minimize the likelihood of such an outcome without undue complexity, we rely on two measures.

First, when a multi-homed AS (i.e., an AS with multiple providers) announces its *own* prefixes, it selects a *single* blue provider, i.e., one to which it announces its prefixes through its blue process only, with the remaining providers acting as red providers, i.e., they learn about the AS' own prefixes only over their red process[4]. This initial "coloring" of how an AS advertises its own prefixes to its providers ensures that each of them can be reached through red and blue paths associated with different last hop providers.

Our second measure seeks to ensure the successful propagation of at least one blue path, while not overly penalizing the possible propagation of red paths. For this purpose, we introduce a new BGP path attribute, *Lock*, which takes a value of 1 when set, and which is set by the origin AS when advertising prefixes to the selected blue provider. Subsequent providers that receive a blue route with the *Lock* attribute set are required to propagate it further to one other (provider) AS with the *Lock* attribute set, and possibly to other (provider) ASes but this time without the *Lock* attribute set. This guarantees the creation of a blue downhill path to all prefixes, and given that paths of both colors are always announced to peers and customers, the availability of a blue path to that prefix from all ASes. Providers that receive a blue route with the *Lock* attribute unset are not required to propagate it. Furthermore, propagating red routes to providers (except a locked blue provider if present) is given precedence over the propagation of blue routes. This maximizes the odds that red paths eventually reach all ASes.

4.2 Properties of STAMP

Having described STAMP, we outline why it is safe

[3]Note that this can be readily implemented using standard BGP mechanisms such as filters, and that all prefixes are still announced to all provider; only the color of the process used to announce them varies.

[4]For a single-homed origin AS, this occurs at its first multi-homed direct/indirect provider.

and produces *complementary* paths. Details are again available in [14].

We first note that the only difference between STAMP and BGP is that a STAMP routing process selectively announces routes to providers. Selective announcements only limit the routes announced in STAMP, when compared to BGP. Hence, each STAMP routing process remains safe as long as BGP itself is safe. When it comes to the paths generated by STAMP, note that its red and blue processes never announce their best routes to the same providers. Hence, if both the red and blue routing processes of an AS have paths to a prefix, then the paths must be downhill node disjoint. Based on Lemmas 3.1 and 3.2, this implies that the red and blue routing processes are complementary which we summarize in Theorem 4.1, whose proof is in [14].

THEOREM 4.1. *Under one routing event, the red and blue routing processes in STAMP are complementary.*

Theorem 4.1 not withstanding, while in STAMP red and blue paths to a given prefix are complementary, its does not ensure that *both* of them exist. As mentioned above, a blue path will always exist, albeit not always as a customer route, but a red path may not. In order for a red AS path to exists, at least one red path must successfully propagate to a tier-1 AS (see [14] for a formal proof). In Section 6, we use an AS graph derived from BGP routing tables to show that given the high level of connectivity in the current Internet, the odds of this happening are high (over 92%).

5. USING STAMP ROUTES

Once STAMP has computed routes, their use in forwarding packets is obviously of importance. This section briefly reviews packet forwarding rules without diving into how to implement the forwarding function, e.g., using techniques such as packet marking adopted in [12, 16], or virtual interface as proposed in [11]. A virtual interface based solution is backward compatible with the current infrastructure but adds complexity, which is avoided by packet marking that, however, requires the use of some header bits, e.g., DS bits. Note that both techniques have certain overheads which are the cost we have to pay in order to achieve reliable data forwarding.

5.1 Packet Forwarding

A first assumption is that the source AS (or the first AS with that capability) assigns an initial "color" to packets it originates. A transit AS can then receive packets from either color[5] for a given destination, but must, whenever possible, forward packets of a given color using routes of the same color. The exception to this rule is when the intended route is experiencing instabilities (more on this below), in which case the AS changes the packet color and forwards it using the route of the other color. Such a change can, however, only be performed once for each packet to avoid loops [12].

[5]Note that if an AS only has a blue route to a destination, it will never receive a red packet for that destination since it never announced a red route to its neighbors.

5.2 Detecting and Avoiding Route Instabilities

Recall that our goal is to always select a routing process whose paths are not affected by transient problems. Realizing this goal involves addressing two issues: (1) detecting problematic behavior in routing processes, (2) identifying which routing process is free of transient problems. According to Lemma 3.1 and 3.2, problems arise only when a routing process loses routes (withdrawals). Whether the trigger for a route events is a withdrawal is known to the AS adjacent to where the failure (or policy change) took place, but not necessarily to ASes further away. In order to preserve the knowledge of what originally triggered a routing event, we add a new path attribute ET (Event Type) to BGP update messages (this is our second "minor" modification to BGP). The ET attribute is 1-bit of information that indicates whether the update was caused by losing a route (ET=0) or not (ET=1). A detailed discussion on how to set ET can be found in [14].

The ET attribute is then used as follows to determine which route can be used: If an AS is using a best route computed by one process and that process loses the route (receives an update message with ET=0 or the adjacent link/node fails), the AS switches to the route selected by the other process. If both processes receive update messages with ET=0, either process that still has a route can be used.

Theorem 5.1, whose proof can again be found in [14], formally establishes that this achieves reliable packet forwarding in the presence of any single routing event. Note that we assume each router is equipped with the functions such as forwarding according the "color" of the packet, detecting potential transient routing problems, and changing the "color" of the packet.

THEOREM 5.1. *In case of any single routing events, STAMP ensures that no packet will be looped or blackholed once the ASes adjacent to where the routing event occured have detected the event.*

6. EXPERIMENTAL EVALUATION

Given that STAMP may not always succeed in discovering blue and red paths at all ASes even when they exist, we first evaluate STAMP's performance along that dimension. Next, we compare STAMP with other schemes under various failure scenarios by simulations.

In order to carry out meaningful and realistic evaluations, we conduct our experiments using an AS Internet topology derived from BGP routing tables collected by the RouteViews project. The underlying AS relationships are inferred using Gao's algorithm [5].

6.1 Realizing Disjoint Paths

Can STAMP ensure that all ASes have both blue and red paths to a destination? This depends on the AS topology as well as on how the ASes select their locked blue provider. We initially assume that the locked blue provider is selected randomly among all providers of an AS. Given the AS topology, we can then compute the odds that ASes have both blue and red paths to a destination. Let Φ_m be the probability that all ASes have both red and blue routes to multi-homed AS m, and denote λ as the number of all possible paths from m to any tier-1 AS. If path $l_i, 1 \leq i \leq \lambda$, is selected as the "locked blue path" from m to a tier-1 AS and a disjoint path from m to another tier-1 AS exists, we say that l_i is a "good" locked blue path since we know that STAMP will then be able to find a red path. If there are $\lambda^{'}$ good locked blue paths, $\Phi_m = \frac{\lambda^{'}}{\lambda}$. *For a single-homed AS s, $\Phi_s = \Phi_m$ if m is the first multi-homed (direct/indirect) provider of s.*

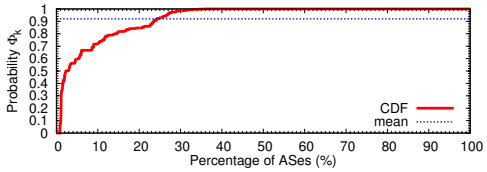


Figure 1: The CDF of Φ_k.

In Figure 1, we plot the CDF (Cumulative Distribution Function) of Φ_k for all destinations. We see that less than 10% of destinations have $\Phi_k \leq 0.7$. Conversely, more than 75% of destination ASes have a probability greater than 0.9 that all other ASes can reach them through both red and blue paths. On average, all ASes have both red and blue paths to any destination AS with probability 0.92.

An intuitive option to improve the odds of STAMP constructing both red and blue paths, is to let the source AS "intelligently" select its locked blue provider rather than select it randomly as other ASes do. This can be realized with minimal information, and we show in [14] that this can increase the percentage of ASes with red and blue paths to all destinations from 92% to 97%.

6.2 Performance Under Failure – Comparison to Other Schemes

The previous experiments focused on evaluating the ability of STAMP to provide protections again *any* single routing event, which does not account for the actual impact of each possible failure scenario, e.g., some failures may not have an impact even for ASes for which STAMP did not succeed in identifying both red and blue paths. In order to better assess STAMP's actual benefits in the presence of failures, we developed an event-driven simulator to replicate routing dynamics. We implemented BGP, R-BGP, and STAMP in the simulator. For all protocols, both processing and transmission delays are modeled as a random variable uniformly distributed in $[10ms, 20ms]$. The BGP MRAI timer is peer-based and set to 30 seconds multiplied by a random factor uniformly distributed within $[0.75, 1.0]$.

6.2.1 *Single link failure*

We simulate routing convergence after a multi-homed AS fails one of its provider links. The destination AS is randomly selected across 100 simulation instances. The average (across all 100 scenarios) number of ASes having transient problems is shown in Figure 2. BGP has more than 6,000 ASes experiencing transient problems. Although R-BGP handles single link failure very well, it requires RCI mechanism, which adds significant complexity to the routing system. Nevertheless, we include it as a benchmark against which to compare STAMP. Note that without RCI, R-BGP results in over 2,000 ASes being affected in some ways by failures. STAMP has about 350 ASes that experience transient problems. Considering that the actual Internet is likely to be more densely connected than the partial AS topology we derived from BGP tables, STAMP should perform better in practice.

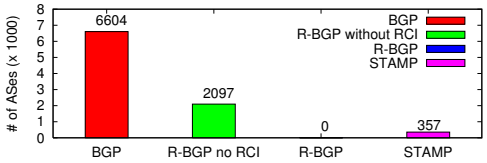
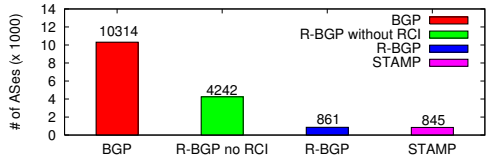


Figure 2: Number of ASes with transient problems under single link failure.

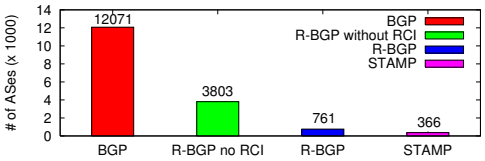
6.2.2 *Multiple link failures*

Next, we consider scenarios where multiple links fail simultaneously (or policy changes affect multiple ASes). We distinguish between two cases: i) the two failed links are not connected to the same AS; and ii) the two failed links are connected to the same AS. In the first case, an origin AS fails one of its provider links and another randomly selected indirect provider link (multi-hop away from the origin AS). In the second case, an origin AS fails a link to one of its providers and that provider also fails one of its own provider links.

The average number of ASes experiencing transient problems are presented in Figure 3. When the two failed links are not connected to the same AS, both STAMP and R-BGP perform similarly, while when the two failed links are connected to the same AS, STAMP experiences about half fewer problems than R-BGP. This is because multiple link failures at the same AS correspond to a "single" routing event for STAMP; something against which its node-disjoint path selection offers protection. A similar set of conclusions hold in the presence of single node (AS) failures, which correspond to an AS withdrawing a route from all its neighbors.



(a) two failed links not connect to the same AS



(b) two failed links connect to the same AS

Figure 3: Number of ASes with transient problems under multiple link failures.

Note that when R-BGP is not afforded the benefit of RCI, its performance again degrades significantly.

6.3 Additional Experiments

We also conducted experiments to evaluate STAMP in terms of partial deployment, convergence delay, and protocol message overhead. Because of space limit, we only briefly review the results here and refer again to [14] for further details.

In terms of partial deployment, we focused on a scenario where STAMP is deployed only at tier-1 ASes, and found that this would result in about 75% of all ASes having two downhill node disjoint paths to any destination. In terms of protocol overhead, STAMP using two parallel routing processes generate less than twice the number of updates as one standard BGP process. If STAMP is implemented as one process handling two instances, we expect the number of updates to be similar to that of BGP. Finally, when it comes to convergence delay, we found that in spite of the possibility of back-tracking caused by its selective announcement rules, STAMP actually converges faster than standard BGP in response to the same routing event.

7. CONCLUSION

The paper proposed a multi-process routing solution, STAMP, which seeks to mitigate the occurrence of transient problems (loops and black-holes) in today's inter-domain routing. Given the wealth of operational knowledge embedded in BGP and the many benefits of its flexibility, STAMP sought to realize this goal with minimal changes to BGP (it requires two new simple attributes and coordination between its two processes that can be realized using existing BGP mechanism, e.g., selective path announcements). STAMP computes two *complementary* AS paths, which we show can be accomplished by focusing only on the downhill portion of AS paths, and using a simple heuristic for path selection. STAMP was evaluated through extensive experiments, which demonstrated its benefits in terms of greater routing stability compared to BGP. These improvements were comparable, and for some important failure scenarios, better than those of previous proposals that also called for more extensive and potentially complex modifications to BGP, e.g., in the form of RCI.

8. REFERENCES

[1] A. B. Barr and et al. Improved BGP convergence via ghost flushing. In *INFOCOM*, 2003.

[2] M. Caesar and et al. Design and implementation of a routing control platform. In *NSDI*, 2005.

[3] J. Chandrashekar and et al. Limiting path exploration in BGP. In *INFOCOM*, 2005.

[4] T. Erlebach and et al. Cuts and disjoint paths in the valley-free path model. In *CAAN*, 2004.

[5] L. Gao. On inferring autonomous system relationships in the internet. *IEEE/ACM Trans. Netw.*, 9(6):733–745, 2001.

[6] L. Gao and J. Rexford. Stable internet routing without global coordination. In *SIGMETRICS*, 2000.

[7] A. Greenberg and et al. A clean slate 4d approach to network control and management. *SIGCOMM Comput. Commun. Rev.*, 35(5), 2005.

[8] T. Griffin and G. Wilfong. An analysis of BGP convergence properties. *SIGCOMM Comput. Commun. Rev.*, 29(4), 1999.

[9] U. Hengartner and et al. Detection and analysis of routing loops in packet traces. In *IMW '02*, pages 107–112, New York, NY, USA, 2002. ACM.

[10] N. Kushman and et al. Can you hear me now?!: it must be BGP. *SIGCOMM Comput. Commun. Rev.*, 37(2), 2007.

[11] N. Kushman and et al. R-BGP: Staying connected in a connected world. In *NSDI*, 2007.

[12] A. Kvalbein and et al. Fast IP network recovery using multiple routing configurations. In *INFOCOM*, 2006.

[13] C. Labovitz and et al. Delayed internet routing convergence. In *SIGCOMM*, 2000.

[14] Y. Liao and et al. Multi-process inter-domain routing. Technical Report TR-08-CSE-09, ECE Department, UMass Amherst, 2008. `http://rio.ecs.umass.edu/~yliao/mpr-tech.pdf`.

[15] J. Luo and et al. An approach to accelerate convergence for path vector protocol. In *Proceedings of Globecom*, 2002.

[16] M. Motiwala and et al. Path Splicing. In *SIGCOMM*, Seattle, WA, August 2008.

[17] D. Pei and et al. BGP-RCN: improving BGP convergence through root cause notification. *Comput. Netw. ISDN Syst.*, 48(2):175–194, 2005.

[18] F. Wang and et al. On understanding of transient interdomain routing failures. In *Proceedings of IEEE ICNP*, 2005.