Bringing Order To BGP: Decreasing Time and Message Complexity

Anat Bremler-Barr*, Nir Chen*, Jussi Kangasharju†, Osnat (Ossi) Mokryn*, and Yuval Shavitt‡
*Inter disciplinary Center Herzliya
(bremler,chen.nir,ossi)@idc.ac.il
† Darmstadt University of Technology
jussi@tk.informatik.tu-darmstadt.de
‡Tel-Aviv University
shavitt@eng.tau.ac.il

***Abstract*— The Border Gateway Protocol (BGP), defacto the routing protocol of the internet, generates lots of traffic after changes in the underlying backbone. Previous papers [1], [2] show that BGP suffers from high convergence delay and high message complexity after a fail down (detachment) of a network, due to path exploration caused by a *limited version of the counting to infinity problem*.**

Surprisingly, we show in this paper, that BGP suffers from unnecessary high message complexity also after an *up* event (reattachment of a network) due to another problem, the path exploration problem caused by *race conditions*. Race conditions in path exploration, is an inherent phenomenon of distributed networks, where the variable link delays, may cause the router to receive and send less preferred updates before receiving the more preferred update messages

This traffic has several implications: it burdens the backbone routers, it increases the convergence time, it may falsely trigger the route flap damping mechanism and it makes it harder for service providers and researchers to understand the root cause of a given change.

We suggest a minor modification to the waiting rule of BGP that pseudo-orders the network and thus reduces the latency by half and the message complexity from $O(DE)$ to $O(E)$ in an *up* event (where D is the Diameter of the internet and E is the number of connections between ASes). From an empirical study on raw BGP update dumps and simulation, we estimate that up to 25% of the sent messages in an *up* event, can be eliminated by our modification.

Keywords: Routing Protocols, BGP, convergence, path exploration

I. Introduction

Internet routing convergence is essential for the correct functioning of all Internet services. Yet, the Border Gateway Protocol (BGP), the de facto routing protocol of the internet, suffers from delayed convergence (up to 15 minutes) and heavy traffic after a fail down (detachment) of a network [1], [2]. The root cause of this convergence problem is due to a limited version of the *counting to infinity problem* [1], [2]. Abstractly the problem is, that following the failure of a destination or some links to a destination, there are pieces of incorrect information floating in the network for a relatively long period of time. These pieces of information are reminiscent of the paths to a destination that was detached from the network. To make things worse, some routers rely on the false information to generate more false information.

In this paper we show that, surprisingly, BGP suffers from unnecessary convergence delay and message complexity also after an *up* event, where a network is reattached. We found through analysis of BGP update dumps that after a reattachment of a network, a router sends on average 3.17 messages to its neighbors during the convergence time. In this case, BGP suffers also from delay convergence time of 1 to 3 minutes [2].

We show that the root cause of high message complexity and the longer delay is due to the result of *race conditions* in path exploration. Distributed protocols suffer from inherent race conditions, where the variable link delays may cause the router to receive and send less preferred updates before receiving the more preferred update messages (this problem was also mentioned in [3], [4]).

One of our main contributions is a surprisingly minor modification to the BGP protocol that reduces in the *up* event the latency by half and the message complexity from $O(DE)$ to $O(E)$, where E is the number of connections between ASes, and D is the diameter of the Internet in AS hops. The key element of our algorithm which we named **pseudo-ordering algorithm** is a simple modification to BGP's $minRouteAdver$ waiting rule, which sets the amount of time BGP enforces between the sending of consecutive announcements from a router to its neighbors.

Our modification and analysis rely on the fact that even though BGP is *not* a shortest path protocol [1] most routers usually select the shortest route among those announced by its neighbors (where some of the neighbors do not announce routes due to policy decision [5]). We validate our assumption using empirical results and show that this is a direct consequence of the common practice policy rules.

Distributed Algorithm wise, our algorithm succeeds in reducing the factor of D in message complexity (from $O(DE)$ to $O(E)$) since the D factor is due to the asynchronous nature of the network, where router may receive less preferred updates (with longer distance) before receiving the more preferred update messages (with shorter distance). Our suggested algorithm is similar in nature to some weak version of a synchronizer

[1] i.e. BGP does not choose the topologic shortest path

[6], hand tailored to the BGP routing problem. The basic idea of the algorithm is that each router waits enough time before it announces its preferred route to assure that it receives the message with the shortest route. Since BGP protocol today enforces a delay between the propagation of announcements anyhow (due to the $minRouteAdver$ rule), our modification does not add to the overall time, it just divides the delay in a more beneficial way. In fact, our modification is purely beneficial, since it reduces the time and message complexity of many cases without harming the complexity of the other cases.

For ease of presentation our paper is focused on *up* events. However, path exploration due to race conditions also affects the message complexity and time complexity including the case of *shorter* event, in which routes to a destination change to a shorter path. [2] Note that the convergence delay due to limited counting to infinity problem, is an orthogonal problem to the path exploration due to *race conditions*, and as such they have different solutions, that should be combined together to improve the overall convergence of the internet.

We show that race conditions in path exploration have a significant impact. We analyze the average case using simulation on SSFNet [7], and show that 22% of the messages in the average case of an *up* event and around 8.5% of all the BGP messages can be eliminated using our modification. We also verify our results by investigating BGP behavior during *up* events using BGP update dumps. We give a thorough analysis of the convergence characteristics and their distribution. We find that routers in edge autonomous systems (ASes) perform more path explorations due to race conditions, and that the closer a network is to the core the less likely it is for its routers to engage in an extensive path exploration. Using the BGP dumps we estimate that 25% of the messages in the average case of an *up* event can be eliminated.

Reducing both message complexity and delay during BGP convergence has many beneficial aspects. First, it plays a major role in providing QoS and highly available services on the Internet. Second, our modification reduces wrongly triggered events of a route flap damping (RFD) mechanism [8]. In this mechanism, a route that flaps with high frequency receives a penalty and the route is suppressed. The large number of messages during the path exploration due to race condition created as the result of a single *up* event may wrongly trigger the route flap damping mechanism. In the common configuration on Cisco and Juniper routers four messages with different attributes trigger route flap damping. A previous study [9] showed this behavior only in the case of a $down$ event. As we show in this paper, the route flap damping mechanism can also be triggered by an up event, since even during the convergence of an *up event* a router can receive more than four messages. Hence our research contributes one more piece of evidence against RFD [10]. Third, reducing the message complexity plays a major role on reducing the load on the routers (on the main CPU that is in charge of the control traffic). Fourth, it helps in simplifying BGP research, and thus can simplify the search for the root cause analysis of BGP [11], [12], [13]. One of the reasons the task is so complex is due to fact that a single event on the Internet can possibly cause multiple messages that need to be analyzed to pinpoint the cause of this event.

We predict that with the increase in multi-homing [14] and consequently an increase in the amount of alternate valid routes, the percentage of redundant messages caused by path exploration due to race conditions will only increase, and our modifications may be proven to be even more valuable.

The rest of the paper is organized as follows. The next section provides the relevant BGP background, focuses especially on the minRouteAdver rule and in Section III we detail related work. In Section IV we demonstrate the problem of path exploration due to race conditions. In Section V we introduce our pseudo ordering algorithm and analysis its worst case complexity. In section VI we discuss practical considerations of implementing the algorithm. In Section VII we provide numerical results obtained via simulation to demonstrate the effectiveness of pseudo ordering in the average case. In Section VIII we provide empirical results obtained from BGP dumps and estimate the benefit of our algorithm in real life.

II. BGP OVERVIEW AND BACKGROUND

BGP is a distance and path vector routing protocol. Each destination (prefix) entry in a router's routing table contains an $ASpath$ field, which is the preferred path associated with this destination for that router. The $ASpath$ is sent with each update on this destination to the neighboring peers. For each destination a router records the last announcement (with the $ASpath$) it has received from each of its peers (neighboring BGP routers). Then, for each destination the router chooses one of the peers as the next-hop on the preferred path to that destination. Usually the router picks the peer that announced the shortest $ASpath$, however BGP is much more sophisticated and enables a more complex path selection according to policy attributes. Specifically, BGP updates about a route include the following attributes: local pref, ASpath, med, nexthop router id [15]. The path with the highest lexical order attributes is selected as the route. The main motivation and usage of the $ASpath$ is in avoiding cycles in the routing protocol. This is achieved by each router simply invalidating any route that includes the router's own AS number in the path.

BGP is an event driven (incremental) protocol where a router sends an update to its peers only when its preferred $ASpath$ to a destination has changed, due to a topology change or a policy change. There are two types of messages exchanged between peering BGP routers: *announcements* and

[2] In fact the number of *up* + *shorter* events is roughly equal to the number of the complementary events: *down* + *longer* (the event in which routes to a destination change to a longer path). Hence our modification helps to improve the time and message complexity of 50% of the events. We note that the limited version of counting to infinity problem in *down* event has a greater impact on the overall convergence delay than the impact of the race conditions problem in *up* events [2], [1]. Nonetheless the last has significant impact as well.

withdrawals. A router sends an announcement when its preferred $ASpath$ to a destination has been changed or when it has a route to a new destination. Withdrawal messages are sent when a router learns that a subnetwork (i.e., destination) is no more reachable through any of its interfaces. To avoid avalanches of messages and to limit the rate at which routers have to process updates, it is required in BGP that after sending an announcement for a destination a router waits a minimum amount of time before again sending an announcement for the same destination (it is recommended by the IETF to set this delay, called $minRouteAdver$ to 30 seconds [15]). The $minRouteAdver$ rule is applied in most implementations per peer rather than per destination, i.e., a router needs to wait at least 30 seconds from the last time it sends a message to that peer. However the delivery of a withdrawal message is never delayed because BGP tries to avoid *black holes*, in which messages are sent to a destination which is no longer reachable.

There are four types of events [2], [1] which occur while BGP is running in the Internet: 1) *Up* event - A previously unavailable destination is announced as available at a router. 2) *Down* event or *fail-down event*- A previously available destination is announced as unavailable at a router. 3) *Shorter* event - A preferred $ASpath$ to a destination implicitly replaces a less preferred $ASpath$ (e.g., the path is becoming shorter). 4) *Longer* or *fail over* - A less preferred $ASpath$ to a destination implicitly replaces a better (more preferred) $ASpath$. This happens, for example, if the preferred route has failed.

III. Related Work

BGP convergence time is the time it takes the routers participating in the inter-domain routing in the Internet to reach a consistent and updated view of the topology. Extensive research has been down on the subject, from exploring the convergence time [2], [16] and properties [1], [17], [18], [19], [20] to suggestions for improvements [9], [21], [22], [23], [24]. Most of the existing work focuses on the case of fail-down, where a network has become unreachable. *Down* and *Up* events differ in the root cause of the path exploration. In the more researched *down* events, path exploration is the result of a version of the counting to infinity problem. However, in *up* events, it is the result of race conditions that are inherent in distributed protocols caused by the variable link delays. Hence solutions like RCN [23], [20] and others [25] that aim at reducing message and time complexity for *down* events would not help in case of *up* events.

It is well known that the waiting rule of BGP (i.e., the $minRouteAdver$ rule) plays an important role in reducing the message complexity in *fail events*. In [2], it was shown that without this timer each router may explore all non existent simple routes to a destination and hence may send $n!$ messages. Simulations done by [16] show that for each specific network topology there is an optimal value for $minRouteAdver$ that minimizes the convergence time and the message complexity. However, being network specific, such a value does not exist for the continuously evolving Internet. This work is complementary to the one in [26] which uses the $minRouteAdver$ rule to improve the convergence after a *fail down* event. Our work investigates the reasons for late convergence after *up* events and suggests a way to improve them.

As far as we are aware this is the first paper that has analyzed the effect of the phenomenon of path exploration due to race conditions and presents a solution to reduce this negative effect.

IV. The Path Exploration due to Race Conditions in *Up* Events

For the sake of simplicity of describing the problem and proving the algorithms we assume the following throughout this section:

1) Routers select the shortest route [3] among those announced by its neighbors [4]. In BGP jargon: the metric of path selection is the shortest path policy that is used along *valid* AS paths (i.e., valley free [5]).
2) Each AS has one router (i.e., no IBGP)
3) Only one *up* event occurs each time, i.e., only one link or router becomes active and influences the convergence of *up* to that specific destination.
4) The $ASpath$ of an announcement is the real route of the ASes that the announcement traversed in the internet.

In Section VI we discuss the assumptions and the modifications required to the algorithm, in order to be applicable for real life implementation.

The problem of path exploration due to race conditions is rooted in the fact that in BGP, a router chooses its preferred path according to the information it receives from its neighbors (namely, the ASpath to each destination). Each time it receives a more preferred update, it also announces it to some or all of its peers (according to its routing policy). If the messages are received at a router in the inverse preference order (i.e., longer ASpaths arrive first due to shorter link delays on their routes) the router may announce each of the messages, and then so will its peers, thus propagating the less preferred messages before the more preferred messages.

Figure 1 demonstrates this problem. For simplicity we use in this section a simple BGP model as in [1], [27][5] with $minRouteAdver$ rule per destination (a similar result can be shown also if the rule of $minRouteAdver$ is used per peer). Each link delay (in seconds) is marked at the link's side.

The sequence of events described in Figure 1 is as follows: when destination dst becomes reachable again, the announcement about its reattachment starts to propagate in the network. Let us look at AS 3: it first receives the update message from AS 2 and only then from AS 1. Hence, AS 3 changed its

[3] In the case of several routes with the same shortest distance, it arbitrarily picks one of them

[4] where some of the peers do not announce routes due to policy decision [5]

[5] where per particular destination dst the network is considered as a directed Graph $G(V, E)$, where the set V of n nodes corresponds to the different AS's, and the set E to the links connecting neighboring AS's. We associate only one router with each AS.

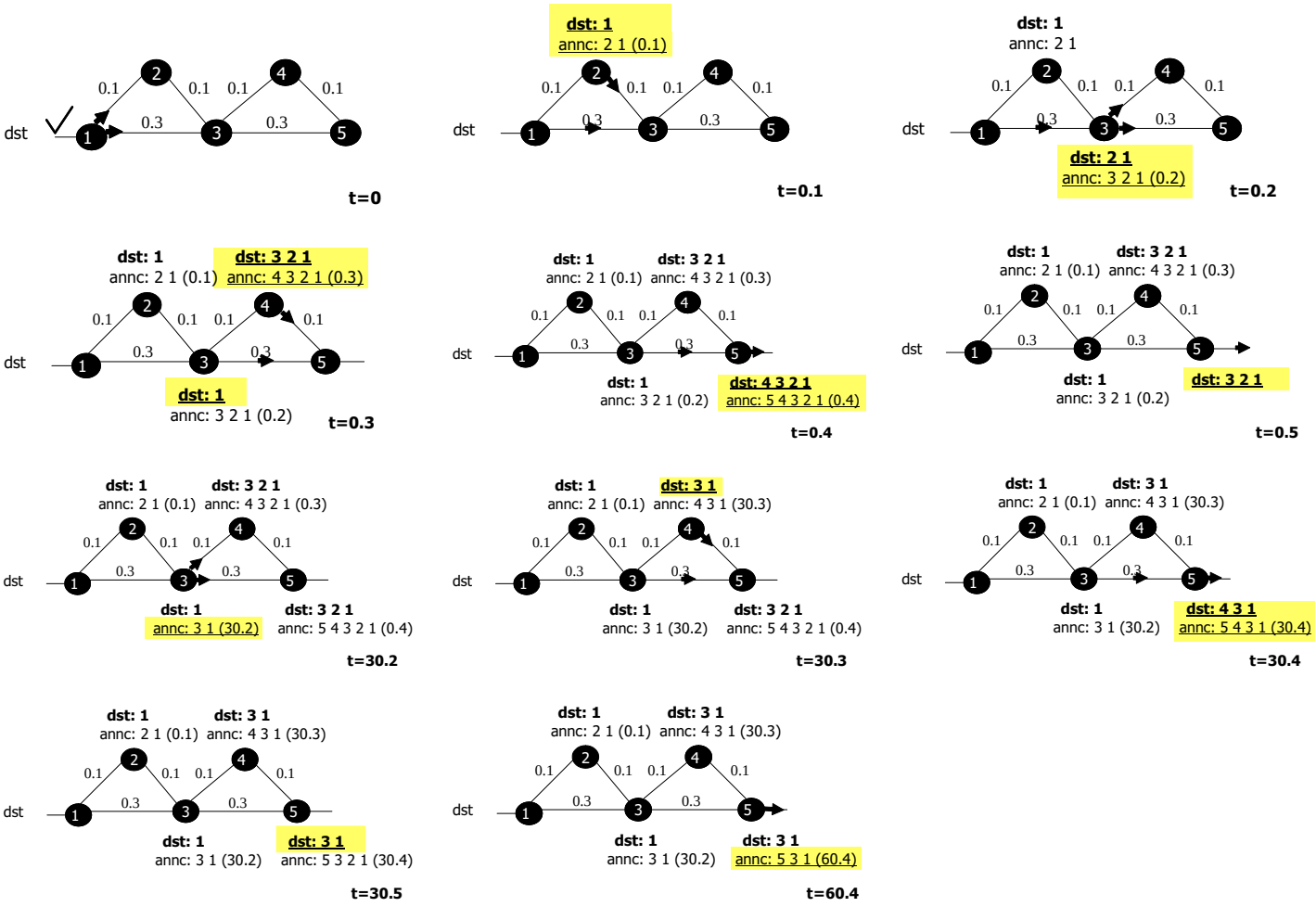


Fig. 1. An example that illustrates the race conditions in BGP path exploration. The numbers near each router indicate the last announced message and the time this message is announced.

preferred path twice, each time announcing it to its neighbors. The first announcement carries $ASpath$ 321 and the second $ASpath$ 31. In a similar way it can be seen that AS 5 sent 3 messages.

Most of the race conditions in the path exploration phenomenon is due to variance of the link delay, but it should be noted that BGP implementation details create new opportunities for additional path exploration. The $minRouteAdver$ rule is applied also in IBGP, the BGP protocol within the same AS BGP speaking routers. The default in CISCO routers is to delay messages by 5 seconds in IBGP peer routers and 30 seconds in EBGP peer routers [28]. Most ASes have several such IBGP routers. The IBGP routers are connected in a full mesh, causing messages to traverse either one or two IBGP routers inside the AS. Hence, the number of ASes along a path differs from the number of traversed BGP (either EBGP or IBGP) routers along it, leading to cases where longer routes are chosen, since IBGP router names are not appended to the ASPATH field. This may lead also to a highly random delay between the different routes, regardless of the actual length of the announced paths.

The $minRouteAdver$ rule has an important effect on reducing the message complexity. It has been proven that in *down* events, without the $minRouteAdver$ waiting rule, the message complexity can get to $n!$, where n is the number of ASes (due to the counting to infinity problem). Surprisingly, we prove in the following lemma that the message complexity without $minRouteAdver$ can be exponential in n also in *up* events, in case of BGP policy. This is due to the race conditions in path exploration.

Lemma 4.1: Implementing BGP without $minRouteAdver$ rule, the message complexity is exponential in n and the time complexity is $O(Dh)$ during the convergence of *up* event, where h is the maximal link delay between two BGP speaking routers in case of BGP policy.
See proof of Lemma 10.1 in Appendix X. ∎

In [1] it was shown that BGP with $minRouteAver$ rule has $O(DE)$ message complexity and $O(D \cdot minRouteAdver)$ time complexity. However, we show in the next section that with a minor modification, we can reduce the message complexity to the optimal limit obtained in a synchronous network,

which is $O(E)$, as stated in the following Lemma:

Lemma 4.2: In a synchronous network the message complexity of *up event* is $O(E)$ with time complexity of $O(Dh)$ (assuming shortest path metric).

The proof is straightforward and follows from the messages being sent according to a clock in a fully synchronized system. The interval between two consecutive clock ticks is set to the maximum link delay, so that by the next clock tick, excluding link failures, all sent messages have arrived. Therefore, messages traversing shorter routes are bound to arrive before messages traversing longer routes. ∎

V. Pseudo-Ordering BGP

In order to eliminate the path exploration due to race conditions, we set the following waiting rule and in this way we achieve the pseudo-ordering of the messages. The proposed solution relies on the assumption that routers select the shortest route among those announced by its neighbors.

```
A router announces its preferred
  new ASpath of length l to its peering,
iff at least Delta seconds
  have passed from the time
  its preferred ASpath has changed.
```

We show two versions of this rule:

1) *Basic version (Delta= $D \cdot h$)* - We set the delay Delta to $D \cdot h$, where D is the diameter of the network and h is the maximal link delay between two BGP speaking routers. We prove that by following this waiting rule, BGP message complexity reduces to $O(E)$ and we show that this waiting rule does not affect the time much, in practice.
2) *Adaptive version (Delta=$l \cdot h$)* - In this case the router takes into account the length of the ASpath sent in the message, l, and thus the delay Delta is set to: $l \cdot h$. We prove a message complexity of $O(E)$ and show that the time complexity decreases by half.

A. Basic version of pseudo-ordering (Delta=Dh)

We show in the following lemma that by delaying a message a fixed delay of: $D \cdot h$ before it is announced, we eliminate completely the path exploration due to race conditions. This fixed delay assures that the router receives all the relevant routes with the shortest route before it announces its preferred route.

Lemma 5.1: In an *up event*, using the basic version of the pseudo ordering rule, a router sends one announcement message with its shortest path to its neighbors.

Proof: We prove this by contradiction. Let us examine two arbitrary messages, m_s and m_g, where m_s is a message with an ASpath of length s, and m_g a message with an ASpath of length g, and $g \geq s$. Let us assume the opposite, namely, a router announces more than one message during an *up* event, hence, according to the shortest path metric policy, it first receives and announces m_g and only then it receives m_s.

The latest time the router *receives* the message about its preferred shortest path of length s :$Dh(s-1)+sh$ (Each of the previous $s-1$ routers delays the message by Dh and each of the s previous links add additional maximum delay of h).

The earliest time the router *can announce* the message with the less preferred ASpath of length g, is in time Dhg. Contradiction, since it is after the time it received the more preferred route s since $Dh(s-1)+sh < Dhg$ (since $s < D$, $g \geq s$). ∎

Corollary 1: In an *up event*, using the pseudo ordering rule, the total message complexity in the internet is $O(E)$.

The pseudo ordering rule reduces the message complexity to $O(E)$ also in the event of *shorter* event.

Lemma 5.2: In *shorter* event, using the pseudo ordering rule, a router sends a message about its shortest path to its neighbors only once.

A Sketch of the Proof: The proof is very similar to the proof of an *up event*. The only difference is that in *shorter* event, the contradiction is received by looking at the time it takes from the time the link is up until the time a router sends the announcements about the existence of a new route. ∎

Claim 2: In an *up* event, using the pseudo ordering rule, the convergence time of *up* event is $\sum_{i=1}^{i=D}(D \cdot h + h) = D^2h + Dh$ where D is the diameter of the Internet.
The proof is straight forward. ∎

Next we discuss the implications of changing the BGP waiting rule on the convergence time. We discuss both the worst case and average case delay and show that we do not harm either. According to the new rule, every router delays the messages by Dh, hence the convergence time is $O(D^2h + Dh)$. In BGP, the worst case convergence time is $O(minRouteAdver \cdot D + Dh)$. We show here that in today's Internet, $O(D^2h) \sim O(minRouteAdver \cdot D)$, where $minRouteAdver = 30$ seconds.

To do that, we first have to determine current values of D and h. Due to the small world phenomenon of the Internet, the diameter D of the Internet AS graph is rather small. We can estimate a diameter of $D \sim 12$ with the current policy routing constrains (i.e., real $ASpath$ lengths) [26]. Clearly, it is sufficient to set $h = 1$ second [29] [6], hence the the delay imposes at each router by the pseudo ordering algorithm is 12 seconds. Hence, it is clear that the worst case complexity is even less than that of BGP (since at the worst case time convergence of pseudo-ordering algorithm $= D^2h + D \cdot h \sim 12 \cdot 13$ where in BGP $= D \cdot minRouteAdver \sim 12 \cdot 30$). In most BGP implementations [2], an announcement message is

[6]In extreme situations of global instability (such as during a worm attack), $h = 1$ may not hold, nor will a delay bound of 12 seconds suffice for some periphery ASes, in which case the peripheral local message complexity may increase. However, our main aim is to reduce the message complexity in the average case and not in the very rare extreme scenarios

delayed by $minRouteAdver = 30$ seconds per peer, rather than per destination. Therefore, an announcement message is delayed by 15 seconds on the average at each router, which is of the same order as the pseudo-ordering delay (since at the average case time convergence of pseudo-ordering algorithm $= D^2h + Dh \sim 12 \cdot 13$ where in BGP $= D \cdot \frac{minRouteAdver}{2} \sim 12 \cdot 15$). [7]

B. Adaptive Pseudo Ordering (Delta=lh)

The pseudo ordering imposes a fixed delay at each router. However, it is sufficient to delay the announcement according to its ASpath length, thus improving both message complexity and convergence latency. The idea is that a message that announces a route of length l needs to wait at a router only $l \cdot h$ seconds, thus enabling all competing messages of a shorter path length to arrive earlier at the router and enables it to send only the referred one.

Figure 2 shows the scenario of Figure 1 using the adaptive pseudo ordering rule. Notice that every node announces messages to its neighbors only once.

Lemma 5.3: In an *up event*, using the adaptive version of the pseudo ordering rule, a router sends one announcement message with its shortest path to its neighbors.

Proof: We prove this by contradiction. Let us examine two arbitrary messages, m_s and m_g, where m_s is a message with an ASpath of length s, and m_g a message with an ASpath of length g, and $g \geq s$. Let us assume the opposite, namely, a router announces more than one message during an *up* event, hence, according to the shortest path metric policy, it first receives and announces m_g and only then it receives m_s.

The latest time the router receives the message about its preferred shortest path of length s : $\sum_{i=1}^{i=s-1} i \cdot h + s \cdot h$.

The earliest time the router can announce the message with the less preferred ASpath of length g, is in time $\sum_{i=1}^{i=g} i \cdot h$.

Contradiction, since it is after the time it received the more preferred route s since $\sum_{i=1}^{i=s-1} i \cdot h + s \cdot h \leq \sum_{i=1}^{i=g} i \cdot h$ (since $g \geq s$). ∎

Corollary 3: In an *up event*, using adaptive version of the pseudo ordering rule, the total message complexity in the internet is $O(E)$.

Claim 4: In an *up* event, using the adaptive delay pseudo ordering rule, the convergence time of *up* event is $\sum_{i=1}^{i=D}(i \cdot h + h) = \frac{D^2h+3Dh}{2}$ where D is the diameter of the Internet.

The proof is straight forward. ∎

[7] Our modification increases the best case, since in the theoretically best case in the original $minRouteAdver$ rule a router may not delay the message if it did not send a message in the previous 30 seconds. While, using our pseudo-ordering rule each router add some additional delay. However, this is only theoretical best case, since in the common implementation today of $minRouteAdver$ the rule is per peer, and studies [2], [30] show that the rule is always set since routers send at least one message per peer every 30 seconds.

	Time	Message
BGP no MinRouteAdver	Dh	$2^{n/2}$
Synchronous BGP	Dh	E
BGP with MinRouteAdver	$30D$	DE
Pseudo Ordering	$D^2h + Dh (\sim\leq 30D)$	E
Adaptive Pseduo Ordering	$\frac{D^2h+3Dh}{2} (\sim\leq \frac{30D}{2})$	E

TABLE I

THE CONVERGENCE COMPLEXITY (TIME AND MESSAGE) OF *up* EVENT WHERE h IS THE BOUND ON ONE HOP DELAY, AND D IS THE INTERNET DIAMETER, n THE NUMBER OF ASES AND E THE NUMBER OF THE LINKS BETWEEN ASES IN THE INTERNET

Hence, the convergence time is almost half of the convergence time of the pseudo ordering rule (which converges at $D^2h + Dh$). Since we argue that the convergence time of the pseudo ordering rule approximates the convergence time of BGP, it follows that the adaptive delay pseudo ordering reduces the convergence time of BGP by a half.

We can further reduce the average convergence time of *up* event by suggesting the **Enhanced Adaptive Pseudo-Ordering algorithm**: The key idea is to wait at least $l \cdot h$ before announcing a message with $ASpath\ l$, from the time the router received *some message of length* $l-1$ [8] *during the convergence process of the current up event*.[9] It is easy to see that the proof of the worst case still holds, but this modification improves the average case convergence time.

Table I summarize the worst case analysis of the current status of BGP in *up* event and the analysis of our algorithm.

VI. IMPLEMENTING CONCERNS

Our modification requires a minor change in the BGP program at the routers (add a delay to the outgoing message), without any need for global protocol modification. In Juniper routers there is already the ability to configure a similar parameter. Juniper routers have a parameter named out-delay, which sets the time period between storing route information in the routing table and advertising the route to the BGP neighbors [31]. However we did not find any record of any practical recommendations of how to set up this delay. This work can be seen as a recommendation of how to configure the out-delay parameter.

Next we revisit some of our earlier assumptions and show that either they can be relaxed or that they describe the current situation in the internet.

A. BGP path selection policy

BGP is a policy based protocol, which advertises routes according to strict peering rules based on business agreements between the ISPs [5]. In Section VIII, we analyze real BGP

[8] The received message is of length $l-1$ before the router appends its AS number to the $ASpath$

[9] In section VIII we explain how we can decide which sequence of messages about the same destination are trigged by the same up event.

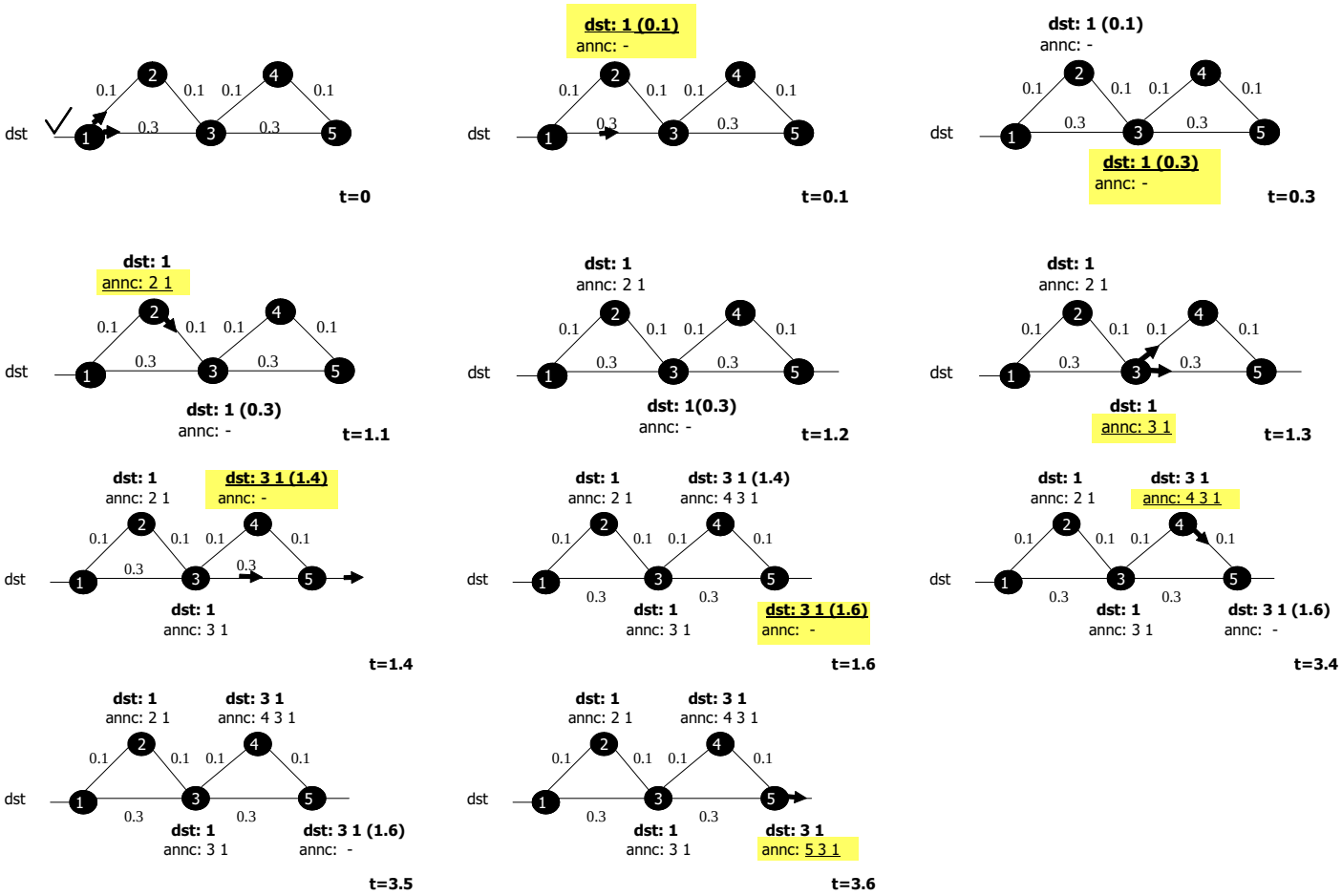


Fig. 2. The scenario of Figure 1 using adaptive pseudo ordering algorithm with $h = 1$.

dumps and estimate that in real life BGP, with its complicated policy, our algorithm reduces the message complexity in *up* event by 25%. Surprisingly, we show also that 66% of the *up* events converge to the shortest path. In this section we explain why the policy, in which routers select the shortest route among those announced by its neighbors, is still the most common used metrics in today's policy routing BGP.

There are two basic ways to override the default shortest path selection preference [32]: 1. Outbound filtering of routes. 2. by using Local Preference.

Outbound filtering is the common method to enforce the business relations between the ISPs (provider, customer, peer or sibling). The common practice is that an AS exports all of the announcements to its customers, but to provider or peer it just exports the announcements about its own network or the networks of its customers and filters out all other announcements. The main motivation is that AS does not want to be a transit AS for its providers. Our algorithm, as the BGP protocol, considers only *advertised*, therefore legal, routes. Hence our algorithm assumes shortest path policy under the peering rules (In BGP jargon: valid routes - valley free routes[5]).

Another mechanism to override the shortest path policy is by configuring the local preference attributes. A common policy configured using local preference is to prefer routes from customers over routes from peers and providers and routes from peers are typically preferred over those from providers [33]. We argue in the following claim that due to the common practice of the outbound filtering rules, in most of the cases one of the paths with the shortest length is chosen. We base our analysis on the common practice of BGP as described above and also on the reasonable assumption that in most cases a provider resides on an equal or a smaller tier than its customer and peer ASes reside on a similar tier level. We discuss and relax the assumption about the division of the ASes to tiers after the proof of the following claim.

Let AS X configure its routing policy by using local preference.

Claim 5: X will chose a route whose length is the smallest from all the routes that were announced by it neighbors.

Proof: Assume the opposite. Hence, due to local preference an AS X chooses a longer route. There are two possible cases where local preference could make X choose a longer route. Case 1: X prefers a longer route through a customer than a shorter one (from provider or peer) or Case 2: X prefers

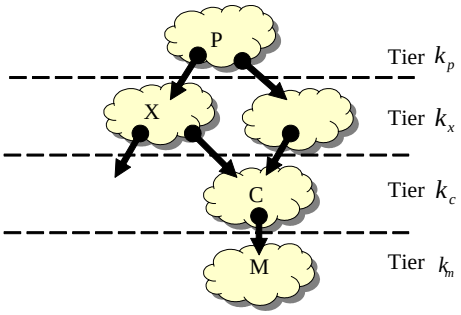


Fig. 3. Illustration of Claim 5

a longer route from peer than a shorter one from provider. We here give proof that case 1 is not possible; case 2 can be proven in a similar way.

Figure 3 illustrates case 1. Let X be in tier k_x, hence any customer C of X is in tier k_c where $k_c >= k_x$, and any provider or peer P is in tier k_p, where $k_p <= k_x$. From the outbound filtering rule, customer C announces only about networks M in tiers k_m where $k_m >= k_c$ (its network or its customers). This results in a contradiction. The customer announces a route with ASpath of length $k_m - k_c$ which is shorter or equal to the route announced by P with ASpath $k_m - k_p$ (since $k_m - k_c \leq k_m - k_p$). ∎

One may wonder, if our assumption about the division of the ASes to tiers, does not oversimplified the complex relationship between ASes. I.e., using the example of Figure 3 could it be that on some occasions M also has a direct link to P. i.e., M has one local provider, C, and one global provider in an higher tier,P. Note, that this is not a typical case, since in this case the connections is to two different providers from different leagues from both an economic and service perspective. Moreover, even if this is the case in order to have a shorter path from P than from C we need M to be at least four tiers below P. This further reduces the probability of this event, since the number of tiers in the internet is small (usually around 4-5 tiers).

In cases where a router receives multiple announcements and does not choose the shortest path, our algorithm would not improve the message complexity and convergence time [10] but we argue that it would not make it worse in the average case. Since, in the overall delay our algorithm does not add additional delays , but just divides the delay differently between the routers along the route.

B. Implementing the protocol in IBGP (ASes may have more than one router)

We suggest an algorithm for routers which are both EBGP and IBGP routers. The modified algorithm works within two phases. In the first phase, each router converges onto the preferred route according to its EBGP router peers' announcements. Then it announces its routes only to other IBGP routes (i.e., within its own AS). In the second phase, the router converges onto the preferred route after receiving the relevant routes from all its IBGP peers. It then announces the converged route to all of it peers (including EBGP peers). Hence a router produces two announcements to its IBGP peers and one to its EBGP peers. The mechanism of two phases is established in order to insure that a router will send only one EBGP message during the whole convergence process.

Delay-wise, in the first phase a router announces its preferred route to its IBGP peers based on the announcement from its EBGP peers either after delaying the announcement according to our pseudo ordering (Dh or lh) or upon receiving an announcement from an IBGP peer. The final announcement the router sends to all of its peers (IBGP and EBGP) in the second stage is delayed $2 \cdot h$ time from its last IBGP announcement in the second phase. The idea is to use the delay of $2 \cdot h$, which is equal to the RTT inside AS, to insure that its first phase announcement is received by all of its IBGP peers and that all of its IBGP peers have time to send it their preferred route. Note that receiving of a message from an IBGP router triggers a router to send the first phase message if it has not already send it.

The proof and the accuracy of our algorithm is based on the bounds on the time it take a message to traverse AS or link. Hence, the case where there is only one router at AS should be changed correspondingly to the case where there are multiple BGP routers at the AS. If there is only one BGP router at the AS, the router should wait $Dh + 2h$ (in the basic version) or $(l + 2) \cdot h$ in the adaptive version. Overall the increment in convergence time over a path with size L due to the interaction of IBGP and EBGP is no more than $2h \cdot L$ and hence minor.

A similar solution works when a route reflector is used in the AS.

C. No concurrent up events to the destination

This means that only one link or router changes its state, becomes active, and influences the convergence of *up* event to that specific destination. This assumption is statistically reasonable in most cases. However, as we explain above, our algorithm might not be able to improve convergence properties, but neither does it harm the convergence properties in cases where this assumption does not hold.

D. The $ASpath$ of an announcement is the real route of the ASes that the announcement traversed in the Internet

The cases where the $ASpath$ does not match the real route of the announcement traversed, may be due to several reasons. The first and the common case is the inflation of AS, where the system administrator duplicates its AS in order to reduce the preference of that route. Note, that since this route preference is according to the $ASpath$ length after inflation, we should also wait before announcing according to its length after the inflation. The second is that of updates which use the option of AS_SET [15]. In this rare case the ASpath is not correlated to

[10] Even in a synchronous network, it is enough for one router not to choose the shortest path to harm the message complexity for all the routers. Let us look again at the Network in Figure 1, assuming we work in a synchronous network. If AS 3 prefers the route of 12 to the route of 1, then all the other routers $4, 5, \ldots$ would announce two messages ($13\cdots$ and $123\cdots$ even if their preference is shortest path.

the actual ASpath length. However AS_SETs are rarely used in the Internet [34] [11].

VII. Simulation

In this section we analyze the average case using simulation. It is difficult to analyze the average case of BGP since the $minRouteAdver$ rule's performance depends on the status of the network, including the topology and the initial value of $minRouteAdver$. In order to analyze the average case, we run simulations of the original BGP and our enhanced adaptive ordering BGP on SSFNet simulator on various topology and $minRouteAdver$ values.

The simulation was done on three different topologies (see Table II, each node in the graph corresponding to one AS): the triangle topology (the topology of our basic example at Figure 1), the diamond topology which demonstrates the case of a graph with multiple equal length paths, and an Internet topology that was used also in [1] to demonstrate a common connections between ASes in the Internet. The delay on each link is chosen randomly between 0 - 0.9 seconds, and the last time the $minRouteAdver$ has been set is chosen randomly. We repeated the test 1000 times and averaged the results.

Table II summarizes the results of our simulation in case of *up* event using the three algorithms: $minRouteAdver$ per $peer$, $minRouteAdver$ per dst and the enhanced adaptive pseudo ordering algorithm with $h = 1$. As we have already proved, the adaptive pseudo-ordering rule does not create any redundant messages due to path exploration. In contrast, BGP creates up to 33% redundant messages in a topology with multiple equal paths, when $minRoutAdver$ is implemented per peer, whereas the per dst performance is considerably worse, with up to 67% redundant messages. The adaptive pseudo-ordering algorithm achieves fast convergence time, which is even better than the performance of BGP when $minRouteAdver$ is implemented per dst, whereas the per peer implementation is dramatically inferior to the per dst implementation. We notice also that $minRouteAdver$ per dst has a tendency to create a large number of redundant messages, twice as many as in the case of BGP with $minRouteAdver$ per peer. This behavior is apparently due to the $minRouteAdver$ timer being almost always set in the per-peer setting. This slows down the sending, and hence we get a higher convergence time, but eliminate some of the path exploration. This results in a smaller amount of messages over longer time periods.

To conclude: the enhanced adaptive pseudo-ordering BGP reduces in *up* event the message complexity (by 22%) and the convergence time (by 80%) from the current BGP implementation with $minRouteAdver$ per peer. Averaging on all type of events we receive reduction of up to 8.5% in the total message complexity and still substantial reduction in convergence time (by 70%).

[11] In case we want BGP to be compatible with the AS_SET option, we can only implement the basic pseudo ordering rule in the internet, and we cannot use the adaptive delay pseudo ordering rule.

Table III shows the results of simulation of *up, down, shorter* and *longer* events on the different topologies again averaging on various initial value of $minRouteAdver$ in the original BGP.[12] The table shows that while the most dominate improvement of convergence properties is shown in case of *up* and *shorter* events, the pseudo-ordering rule does not harm the convergence in other types of event.

VIII. Empirical Investigation

In this section we analyze empirical results of up events in real BGP raw data dumps, obtained from [35]. The data, gathered from several of the RIPE RIS project Remote Route Collectors (RRC's) throughout the world, represents the sequence of *up* events occurring in the first 14 days of June 2005 as seen from each of these collectors. We investigate the characteristics of *up* events and analyze the potential effect of implementing the BGP Pseudo-Ordering Algorithm. For a better understanding of the data, we analyze it in comparison to the AS degree, taken from a map constructed from the union of the routeview project [36] and the Dimes project [37] databases.

To evaluate the behavior of the network with Pseudo-Ordering algorithm, we would like to identify *up* and *shorter* events in the data. $Shorter$ events are harder to spot, since it is hard to distinguish between $shorter$ and $longer$ if the routing policy is general without knowing the preference of every router. Hence, we focus in this section on investigating the characteristics of up events. To do so, we need to identify the detachment of a subnetwork, and then set its reattachment as the starting point. In [12] it is shown that updates arrive less than 70 seconds after the previous update with high probability belong to the same event. Following this rule of thumb, we identify the reattachment of a subnetwork if it is announced *more* than 70 seconds after it was withdrawn. An up event is identified according to the following: An up event starts with the first announcement of a previously withdrawn prefix, and ends when at least 70 seconds have passed from the last announcement on that prefix. [13]

We divide the convergence patterns of up events in the following manner: $c_{shorter}$- Describes all the events in which BGP converges monotonically to the shortest AS path (i.e., we capture different announcements at the router with decreasing length of ASpath . [14]). c_{same}- Describes all the events in which BGP converges monotonically, and all its announcements carry a different ASpath but with same length ASpath. Includes also all cases in which only one announcement is sent during the event. c_{longer}- Describes all the events in which BGP

[12] In longer and shorter events we assume that the failure link is the link connects between the AS that the destination belongs to and one of its neighbors.

[13] Specifically, An up event is identified according to the following: *if a prefix is announced first at least 70 seconds after it was withdrawn, then each consecutive announcement for that prefix is sent within 70 seconds after the last announcement that belongs to the same event. The event terminates when 70 seconds without an announcement passes for that prefix.*

[14] For example the first announcement is with ASpath of length 5 and the second announcement to the same destination is with ASpath of length 3

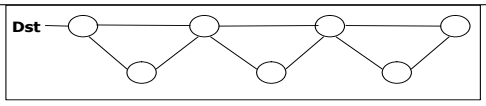
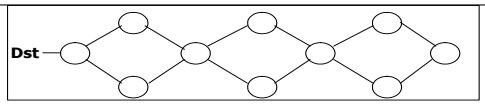
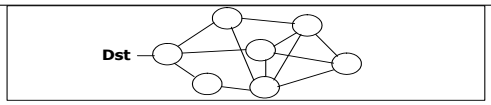
Topology	Figure	Redundant Messages (%)			Time		
		Peer	Dst	Order	Peer	Dst	Order
Triangle Topology		+8%	+42%	+0%	48	11	8
Diamond Topology		+33%	+67%	+0%	81	29	23
Internet Topology		+28%	+61%	+0%	40	27	4

TABLE II

THE AVERAGE INCREASE IN MESSAGE COMPLEXITY AND THE AVERAGE TIME CONVERGENCE IN UP EVENT DUE TO PATH EXPLORATION FOR EACH OF THE THREE TOPOLOGIES. WE RUN EACH OF THE TOPOLOGIES ON THREE VERSIONS OF BGP ALGORITHM: PEER ($minRouterAdver$ PER PEER, AS PER COMMON PRACTICE), DST ($minRouteAdver$ PER dst) AND ORDER (THE ENHANCED ADAPTIVE-PSEUDO ORDERING ALGORITHM)

Topology	Event	Redundant Messages (%)		Time	
		BGP	Order	BGP	Order
Triangle	Up	+8%	+0%	48	8
Triangle	Down	+13%	+13%	3	3
Triangle	Shorter	+10%	+0%	48	10
Triangle	Longer	+30%	+30%	65	23
Diamond	Up	+33%	+0%	81	23
Diamond	Down	+12%	+12%	4	4
Diamond	Shorter	+6%	+0%	88	29
Diamond	Longer	+0%	+0%	72	27
Internet	Up	+28%	+0%	40	4
Internet	Down	+9%	+0%	35	9
Internet	Shorter	+0%	+0%	27	6
Internet	Longer	+0%	+0%	41	9

TABLE III

THE AVERAGE INCREASE IN MESSAGE COMPLEXITY AND THE AVERAGE TIME CONVERGENCE IN DIFFERENT EVENT (UP,DOWN,SHORTER,LONGER) FOR DIFFERENT TOPOLOGIES USING BGP PER PEER AND ORDER ALGORITHM.

RRC	Number Up event	Avg msgs	Sync imprv	c_shorter		c_same		c_longer		c_nonMono	
				%	Avg. msgs	%	Avg. msgs	%	Avg. msgs	%	Avg. msgs
2	36127	1.11	9.39%	2.24%	2.18	97.16%	1.08	0.43%	2.47	0.17%	3.27
5	140846	4.28	47.08%	4.62%	5.39	67.54%	2.66	5.89%	4.47	21.94%	8.99
7	1749	1.56	12.24%	3.77%	2.18	84.51%	1.16	5.43%	2.07	6.29%	6.26
10	142	1.18	17.61%	0%	0	100%	1.18	0 %	0	0%	0
11	106678	5.27	27.71%	14.19%	2.67	33.18%	1.54	33.63%	2.32	19%	4.06
All	285542	3.17	26%	7.9 %	3.43	58.6%	2.07	15.5%	2.72	18%	7.03

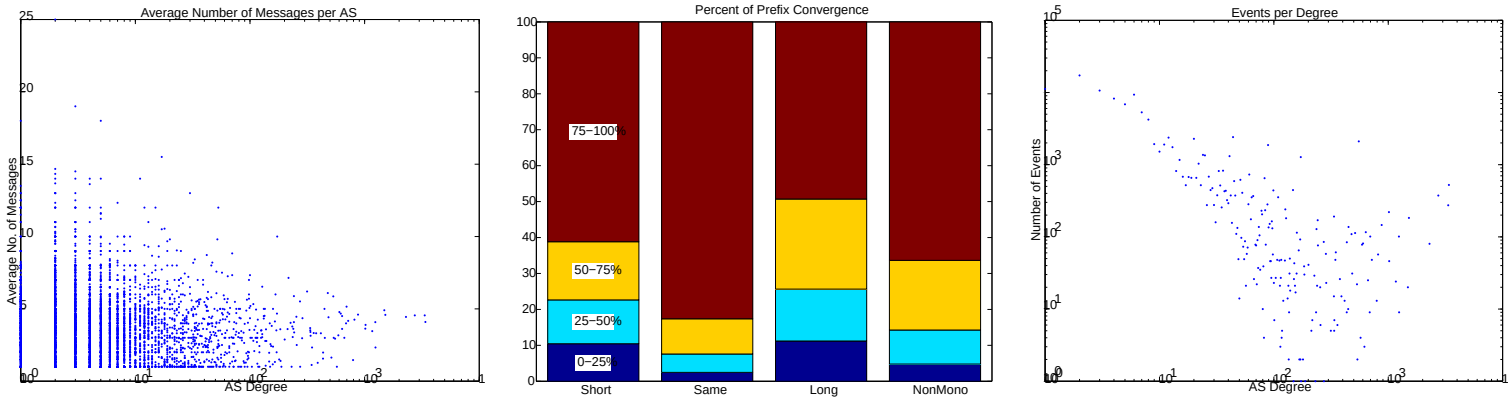
TABLE IV

CHARACTERISTICS OF Up EVENTS CONVERGENCE PROPERTIES

converges monotonically to the longest AS path. $c_{nonMono}$- Describes all the events in which BGP converges, but the sequence of ASpath lengths is not monotonic. Table IV shows the convergence patterns of *up* events. We find that the majority of the events converge monotonically, and only on average in 15.5% of the cases, converge to longer routes. We estimate the improvement gained from the pseudo-ordering as the sum of events of $c_{shorter}$ and c_{same} [15] with more than one message, and receive that the potential improvement is, on average, 25%. We find that the average number of messages is between 1.11 to 5.27, averaged to 3.17. This finding seems to contradict previous findings, calculating the number of messages sent during events generated by the RIPE Beacons [12]. We note that the difference comes from the effect of the periphery ASes, as can also be seen from Figure 4 (a), which shows that lower degree ASes send more messages during the event.

The high number of messages on average in up event

[15]Since in this cases the multiple messages are probably are caused to the path exploration due to race conditions



(a) (b) (c)

Fig. 4. (a) Number of messages during up event per AS degree (b) Distribution of convergence per pattern (c) The cumulative number of Up events for all the ASes of some degree

implies that route flap damping (RFD [8]) mechanism, that is commonly implemented in today's routers, in some of the cases falsely suppresses routes because of the path exploration due to the race conditions phenomenon. In the common configuration of route flap damping [8] a route that changes its $ASpath$ more than four times, will be suppressed for up to half of hour. The implication of the false activation of RFD, is that the route at the router may be fixed not to the most preferred route. Moreover, if in the next half an hour the route fails, RFD will prevent propagation of the new state in the Internet, causing packets to route incorrectly and to be lost in the Internet.

A closer look at the data given in Table IV also leads to the conclusion that different policies at the routers cause a larger number of messages. The largest average number of messages happens when the convergence pattern is non monotonic, i.e., a router changes its preferred path due to a policy decision, causing its peers too also change the preferred route to the one it chose.[16] Hence, complicated policies may lead to a delay in BGP convergence. Figure 4(a) depicts the number of messages each AS sends on average during *up* events per AS degree. There is a distinct decrease in the number of sent messages as the degree increases. Clearly, lower degree ASes, which have a higher probability to reside on the periphery, are engaged in a more extensive path exploration due race conditions and thus send on the average a higher number of messages than the ASes with the higher degrees. Figure 4(b) shows for each convergence pattern, the distribution of convergence to that pattern during an *up* event. For example, 60% of the prefixes that converged onto a shorter path at least once, will converge onto a shorter path in over 75% of the events again, and only 10% will converge onto the shorter path in less than 25% of the events again. Interestingly, the results show that the convergence patterns are rather steady, and in more than 70% of the *up* events a prefix is likely to repeat the same convergence pattern (we obviously excluded all events that generated only one message from this figure). [17]

Figure 4(c) depicts the number of *up* events per all the ASes of some degree. There is a clear decline in the total number of events as the degree increases with a slope of about -2 in a log log plot. I.e., the number of events decreases quadratically with the AS degree. Our results show that the higher degree ASes - i.e., those closer to the core of the Internet - tend to be more stable and converge with a small number of messages, and the lower degree ASes - i.e., the periphery ASes - tend to generate more events and engage in more extensive path exploration due to race conditions. [1] shows a similar phenomenon in *down* events (i.e. in the case of path exploration due to the limited version of counting to infinity). Small ASes, with longer routes to the destination, have a larger number of alternate routes to explore. These local effects at the periphery generate a good portion of the volume of control traffic in the Internet. It may be safe to say that the "action" in BGP is probably at the edge of the network rather than in the middle. This conclusion is also reached by [18] using a different method. To sum up: One conclusion from this work is the understanding that complex policy routing in BGP, increases the amount of control messages sent in the Internet backbone. By analyzing updates of BGP we show that the most of the events that trigger a high number of messages are due to a complex routing policy in one of the routers in the route to the destination[18]. While our modification cannot reduce messages that are due to path exploration due to race conditions with complex routing policy, it does succeed in reducing approximately 25% of the overall messages in up event.

[16] A router changes its preferred path to the destination if its closest peer to the destination changes its preferred path to it.

[17] We estimate that the convergence patterns are steady since they are mainly influenced from the policy rules which are pretty much steady in the internet, since they are manually configured by the network administrators.

[18] This according to the analysis of $c_{nonMono}$ which cannot happen if the routing uses shortest path policy and we show that it causes a high number of messages per event

IX. Concluding Remarks

In this paper, we show that one of the factors that increases the number of BGP messages in the internet backbone is the race conditions in path exploration.

We suggest minor modification rules to BGP that eliminate path exploration in cases where the shortest valid route policy is used, and reduces the number of messages in heterogenous policy routing (where part of the ASes do not use the shortest path metric).

Our insight is that BGP today delays the messages anyhow (due to $minRouteAdver$ rule), and hence we should use this delay in a more productive way to reduce the common case of path exploration due to the race conditions. Our modification does not add to the overall time convergence, on the contrary it improves it, by just dividing the delay along the route in more beneficial way. Using RAW BGP dumps we estimate that approximately 25% of the overall messages in *up* events can be reduced.

References

[1] C. Labovitz, R. Wattenhofer, S. Venkatachary, and A. Ahuja, "The impact of internet policy and topology on delayed routing convergence," in *Proc. INFOCOM*, April 2001.

[2] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanianitz, "Delayed internet routing convergence," in *Sigcomm*, September 2000.

[3] H. Zhang, A. Arora, and Z. Liu, "A stability-oriented approach to improving bgp convergence," in *IEEE Symposium on Reliable Distributed Systems (SRDS)*, 2004.

[4] C. Labovitz, R. Wattenhofer, S. Venkatachary, A. Ahuja, F. Jahanian, and A. Bose, "Understanding the large-scale dynamics of internet routing protocols," in *The Global Internet: Measurement Modeling and Analysis*, 2000.

[5] L. Gao, "On inferring automonous system relationships in the internet," *IEEE Transactions on Networking*, 2002.

[6] B. Awerbuch, "Complexity of network synchronization," *Journal of the Association for Computing Machinery*, vol. 32, no. 4, pp. 804–823, 1985.

[7] "Ssfnet," http://www.ssfnet.org/.

[8] C. Villamizar, R. Chandra, and R. Govindan, "Bgp route flap damping , rfc 2439," NOVEMBER 1998.

[9] Z. M. Mao, R. Govindan, G. Varghese, and R. H. Katz, "Route flap damping exacerbates internet routing convergence," in *SIGCOMM*, 2002.

[10] P. Smith and C. Panigl, "Ripe routing working group recommendations on route-flap damping," 2006.

[11] A. Feldmann, O. Maennel, Z. M. Mao, A. Berger, and B. Maggs, "Locating internet routing instabilities," in *ACM SIGCOMM*, 2004.

[12] 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*, 2005.

[13] R. Teixeira and J. Rexford, "A measurement framework for pin-pointing routing changes," in *Proc. ACM SIGCOMM Network Troubleshooting Workshop*, 2004.

[14] D. Dolev, S. Jamin, O. Mokryn, and Y. Shavitt, "Internet resiliency to attacks and failures under bgp policy routing," *Computer Networks magazine*, 2005.

[15] Y. Rekhter and T. Li, "A border gateway protocol 4 (bgp4)," 1999, draft-ietf-idr-bgp4-09.txt.

[16] T. G. Griffin and B. J. Premore, "An experimental analysis of bgp convergence time," in *ICNP 2001*, Nov. 2001.

[17] T. Griffin and G. Wilfong, "An analysis of BGP convergence properties," in *Proceedings of SIGCOMM*, Cambridge, MA, August 1999, pp. 277–288. [Online]. Available: citeseer.nj.nec.com/griffin99analysis.html

[18] D. Pei, L. Zhang, B. Zhang, R. Izhak-Ratzin, and R. Oliveira, "Quantifying path exploration in the internet," in *IMC*, 2006.

[19] D. Pei, B. Zhang, D. Massey, and L. Zhang, "An analysis of convergence delay in path vector routing protocols," in *Computer Networks*, 2006.

[20] J. Chandrashekar, Z. Duan, Z.-L. Zhang, and J. Krasky, "Limiting path exploration in bgp," in *INFOCOM*, 2005.

[21] A. A. Hongwei Zhang, "Brief announcement: Continuous containment and local stabilization in path-vector routing," in *Symposium on Principles of Distributed Computing*, 2005.

[22] H. Zhang, A. Arora, and Z. Liu, "Lsrp: Local stabilization in shortest path routing," in *IEEE-IFIP International Conference on Dependable Systems and Networks (DSN 2003)*, 2003.

[23] D. Pei, M. Azuma, D. Massey, and L. Zhang, "Bgp-rcn: Improving convergence through root cause notification," *Computer Networks*, 2005.

[24] B. Zhang, D. Massey, and L. Zhang, "Destination reachability and bgp convergence time," in *IEEE GLOBECOM*, 2004.

[25] D. Pei, L. Wang, X. Zhao, D. Massey, L. Zhang, and A. Mankin, "Improving bgp convergence with consistency assertions," in *Proceedings of the IEEE INFOCOM*, 2002.

[26] A. Bremler-Barr, Y. Afek, and S. Schwartz, "Improved bgp convergence via ghost flushing," in *INFOCOM*, 2003.

[27] L. Gao and J. Rexford, "Stable internet routing without global coordination," in *SIGMETRICS*, June 2000, http://www.cisco.com/ nodnik/walla.html.

[28] "Cisco products command reference," http://www.cisco.com/en/US/products/sw/iosswrel/ps5187/products_command_reference

[29] A. Feldmann, H. Kong, O. Maennel, and A. Tudor, "Measuring BGP pass-through times," in *Passive and Active Measurement Workshop (PAM)*, 2004.

[30] T. G. Griffin, "Tutorial: An introduction to interdomain routing and bgp," August 2001.

[31] "Juniper techpubs: out delay command," http://www.juniper.net/techpubs/software/junos/junos53/swconfig53-ipv6/html/ipv6-bgp-summary28.html.

[32] M. Caesar and J. Rexford, "Bgp policies in isp networks," *EEE Network Magazine, special issue on interdomain routing*, 2005.

[33] F. Wang and L. Gao, "Inferring and characterizing internet routing policies," in *ACM SIGCOMM Internet Measurement Conference*, 2003.

[34] D. Meyer and K. Patel, "Bgp-4 protocol analysis," http://www.faqs.org/ftp/rfc/pdf/rfc4274.txt.pdf.

[35] "Routing information service," http://www.ripe.net/ris/.

[36] "University of oregon route views project," http://www.routeviews.org/.

[37] Y. Shavitt and E. Shir, "Dimes - letting the internet measure itself," arXiv, Tech. Rep. cs.NI/0506099, June 2005, available at http://www.arxiv.org/abs/cs.NI/0506099.

X. Appendix

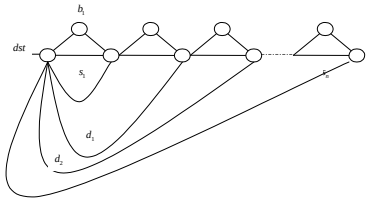


Fig. 5. Example of exponential message complexity in up event

Lemma 10.1: The message complexity of up event without $minRouteAdver$ rule is exponential with the number of n and the time complexity is Dh with BGP policy.

Proof: Let us look at the graph in Figure 5, where there are $\lfloor \frac{n}{2} \rfloor$ triangles and additional direct links from the first node near the destination to all other nodes in the graph. In each triangle, to go from one node to another, we can choose to go over two edges (bypass) or over one edge. Let us denote by 0 the decision to go over the bypass (denoted in the figure by b), and by 1 the decision to choose the straight edge (denoted in the figure by s). An up event in an asynchronous network (with arbitrary delays on the links) can go over all the possible routes, i.e., $000\ldots00$(always decides to take the bypass),

$000\ldots001$, $000\ldots010$, $000\ldots011$ up to $11111\ldots11$ until the network converges to the path using the direct links (d) (in Dh time). This can happen in case of a complex BGP policy where the routers in the network prefer routes that use direct edges near the destination. I.e., a path 1***...* (where* can be 0 or 1) is always prefer than any path from type 0***...*. Similar a path 11***...* is always more prefer than any path from type 10***...*. Since every such new route requires at least one message, we receive $2^{(\frac{n}{2})}$ messages. ∎