Bringing Order To BGP: Decreasing Time and Message Complexity

Anat Bremler-Barr [*] Nir Chen[†] Jussi Kangasharju [‡]
Osnat Mokryn [§]and Yuval Shavitt[¶]

December 26, 2008

Abstract

The Border Gateway Protocol (BGP), the de facto routing protocol of the internet, generates excessive amount of traffic following 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 a high message complexity also after an *up* event (reattachment of a network). We analyze BGP dynamics data from raw update dumps and show that race conditions cause extensive path exploration that increases the amount of redundant updates. We show, based on these BGP dynamics, that up to 26% of the updates sent during *up* events are redundant. We also find that the effect of this phenomenon is bigger when the change occurs at the edge of the network.

We suggest a minor modification to the waiting rule of BGP that pseudo-orders the network and reduces the convergence latency of *up* events by half and the message complexity from $O(DE)$ to $O(E)$, where D is the Diameter of the internet and E is the number of connections between ASes. Our simulation results suggest that our modification may improve the convergence messages and time during all events, with the most noted improvement of up to 36% in the number of messages and 81% in time to convergence during *up* events in Internet like topologies. We show that our results hold also for partial deployment of the modification in only some of the routers.

Keywords: Routing Protocols, BGP, convergence, path exploration

[*]School of Computer Science, The Interdisciplinary Center, Herzliya, Israel. bremler@idc.ac.il
[†]School of Computer Science, The Interdisciplinary Center, Herzliya, Israel. chen.nir@idc.ac.il
[‡]University of HelsinkiJussi.Kangasharju @cs.helsinki.fi
[§]Dept. of Computer Science, Tel Aviv-Yaffo College, Israel. ossi@mta.ac.il
[¶]School of Electrical Engineering, Tel Aviv University, Israel. shavitt@eng.tau.ac.il

1 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 for this convergence problem is 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 3.18 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 during a convergence to an *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 (the MRAI timer), which sets the amount of time BGP enforces between the sending of consecutive announcements from a router to its neighbors.

Previous research showed that BGP policy causes path inflation and investigated its extent [5, 6, 7]. Spring at al. showed in [8] that among paths that conform to no-valley and prefer-customer policies, defined in [9], paths that traverse the fewest ISPs (shortest AS-path length) are usually chosen. They also showed that the prefer-customer policy hardly affects the paths lengths. We further prove that the prefer customer policy enables the router to

choose the route with the fewest ISPs. Our modification and analysis rely on this proof, found in Section 7.

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 [10], 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 MRAI 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.

We show that race conditions in path exploration have a significant impact. We analyze the average case using simulation on SSFNet [11], and show that 32% of the messages in the average case of an *up* event and around 48% 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 26% 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 high availability to services in the Internet. The delayed convergence in up events adds up to 1-3 minutes [2] when the service is not available, or available partly on unstable routes, which may cause a re-ordering of packets. Second, our modification reduces wrongly triggered events of a route flap damping (RFD) mechanism [12], further explained in 2.2. 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 [13, 14, 15]. 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 [16] 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, focusing especially on the MRAI rule and detailing related work. In Section 3 we demonstrate the problem of path exploration due to race conditions. In Section 4 we introduce our pseudo ordering algorithm and analyze its worst case complexity. In Section 5 we provide numerical results obtained via simulation to demonstrate the effectiveness of pseudo ordering in the average case. In Section 6 we provide empirical results obtained from BGP dumps and estimate the benefit of our algorithm in real life. Practical considerations and implementation issues are discussed in Section 7.

2 BGP Overview and Related Work

2.1 BGP Overview

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 and router id [17]. The path with the highest lexical order attributes is selected as the route.

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 *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 *MRAI* to 30 seconds [17])[1]. However, the delivery of a withdrawal message is never delayed to avoid *black holes*, where messages are sent to a destination which is no longer reachable.

There are four types of possible BGP events [2, 1]:

- Up: A previously unavailable destination is announced as available at a router.
- Down or Fail-down: A previously available destination is announced as unavailable at a router.
- Shorter: A preferred *ASpath* to a destination implicitly replaces a less preferred *ASpath* (e.g., the path is becoming shorter).
- 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.

2.2 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 done on the subject, from exploring the convergence time [2, 18] and properties [1, 19, 20, 21, 22] to suggestions for improvements [23, 24, 25, 26, 27]. 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, that occurs in distance vector routing protocols [28, 29][2]. However, in *up* events, it is the result of race conditions that are inherent in distributed protocols caused by the variable link delays.

Solutions like RCN [26, 22] and others [30] that aim at reducing message and time complexity for *down* events, try to eliminate incorrect information (obtained before the failure)

[1]The *MRAI* 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.

[2]BGP is a path vector algorithm, thus eliminating the possibilities of loops. However, it may converge to the longest possible route [29].

from traveling around the network, and thus to decrease the amount of messages that rely on old data. Hence, their solutions would not help in the case of up events, where the problem is not the result of count to infinity like scenario but rather from path exploration.

It is well known that the waiting rule of BGP (i.e., the *MRAI* rule) plays an important role in reducing the message complexity in *down* events. Without it, each router may explore every simple route to a destination and hence may send $n!$ messages [2]. Simulations done by [18] show that for each specific network topology there is an optimal value for *MRAI* 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 [31] which uses the *MRAI* 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.

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, commonly used in the routers in the Internet [23]. In this mechanism, a route that flaps with high frequency receives a penalty and the route is suppressed. In the common configuration on Cisco and Juniper routers four messages with different attributes trigger route flap damping. A previous study [23] 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 [32].

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. A short version of this work was presented at PODC 2007 [33].

3 The Path Exploration due to Race Conditions in *Up* Events

The problem of path exploration due to race conditions is rooted in the distributed characteristics of BGP: a router chooses its preferred path according to the information it receives from its neighbors (namely, the *ASpath* to each destination). Each time a router receives a more preferred update, it 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.

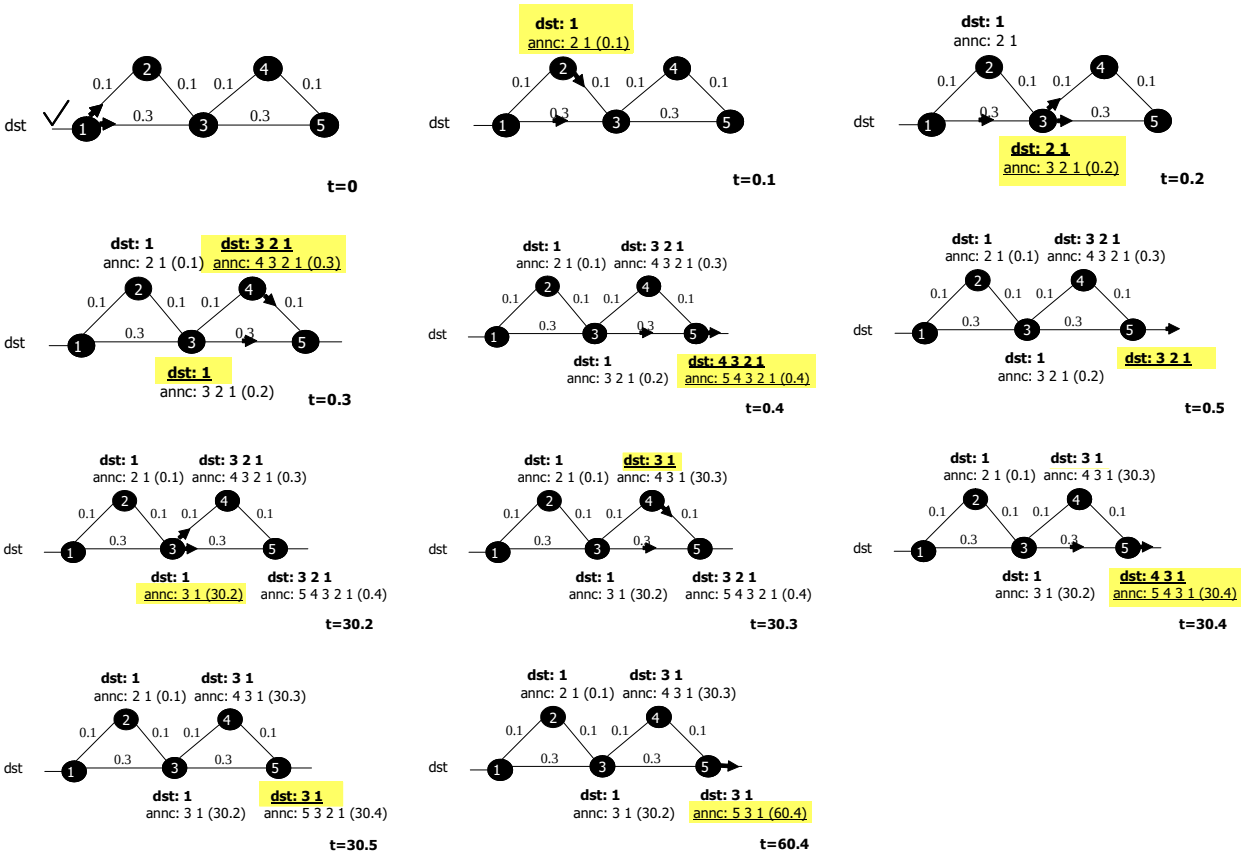


Figure 1: An example that illustrates the race conditions in BGP path exploration. The numbers near each router indicate the last announced message and the time this message is announced.

Figure 1 demonstrates this problem using a simple BGP model used also in [1, 34]. In this model, per a particular destination *dst*, the network is considered as an undirected Graph $G(V, E)$, with a set V of n nodes corresponding to the different AS's, and a set E of edges corresponding to the links connecting neighboring AS's. Each AS is represented as one router, and the $MRAI$ rule is implemented per destination[3]. Each link delay (in seconds) is marked by the link.

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.

[3]A similar result can be shown also if the rule of $MRAI$ is used per peer

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 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.

The *MRAI* rule has an important effect on reducing the message complexity. It has been proven that during *down* events, without the *MRAI* 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 of *up* events, when the *MRAI* timer is not used, can also be exponential in n. This is due to the race conditions during the path exploration.

Lemma 3.1 *Implementing BGP without the* MRAI *rule, the worst case message complexity is exponential in n (the number of ASes) and the time complexity is $O(Dh)$ during the convergence of* up *event, where h is the maximal delay on the link between two BGP speaking routers, and D is the network diameter.*

See proof of Lemma 8.1 in Appendix A. ∎

In [1] it was shown that BGP with the *MRAI* rule implemented has $O(DE)$ message complexity and $O(D \cdot MRAI)$ 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 3.2 *In a synchronous network the message complexity of an* up event *is $O(E)$ with time complexity of $O(Dh)$ (assuming a shortest path metric).*

The proof is straightforward, following the fact that in a fully synchronized system messages are sent according to a clock. 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. ∎

Most of the race conditions in the path exploration phenomenon are due to variances of link delays. It should also be noted that BGP implementation details create new opportunities for additional path explorations. The *MRAI* rule is applied also in IBGP, the BGP protocol

implemented between BGP speaking routers that belong to the same AS. The default in CISCO routers is to delay messages by five seconds between IBGP peer routers and 30 seconds between EBGP peer routers [35]. 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.

Throughout the paper, for simplicity, we assume the following:

1. Based on Spring at al. in [8] and the proof in Section 7, we assume that among paths that conform to no-valley policy, paths that traverse the fewest ISPs (shortest AS-path length) are usually chosen [4]. For simplicity, we refer hence forth to these paths as *shortest policy* paths.

2. Each AS has one router (i.e., no IBGP)

3. Only one *up* event occurs at a time, i.e., only one link or router becomes active and influences the convergence to that specific destination.

4. The *ASpath* in the announcement is not aggregated, and contains the full route the update traversed through.

Section 7 discusses the assumptions and the modifications required in the algorithm for real life adaptation and implementation issues.

4 Pseudo-Ordering BGP

In order to eliminate the path exploration due to race conditions, we set the following waiting rule:

```
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.
```

[4] In the case of several routes with the same AS-path lengths, the router arbitrarily picks one of them

We show two versions of this rule:

1. *Basic version (Delta= $D \cdot h$ seconds)* - 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=$min(l \cdot h, Dh)$ seconds)* - 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: $min(l \cdot h, Dh)$ seconds. We prove a message complexity of $O(E)$ and show that the time complexity decreases by half.

4.1 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 in cases where a shorter path metric is used. This fixed delay assures that the router receives all the relevant routes with the shortest route before it announces its preferred route.

Lemma 4.1 *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 is* $O(E)$.

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

Lemma 4.2 *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. To do that, we first have to determine current values of D and h. Due to the small world phenomenon of the Internet [36], 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) [31]. h is defined as the maximal link delay between two BGP routers. Roughly this can be estimated as the inter router link delay (propagation, transmission and queuing delays) plus the BGP processing time in the router. The inter link delay can be estimated as tens of milliseconds (observation using Netdimes database [37]). The BGP processing time is on the average well less than 150 milliseconds and is in the worst case 400 milliseconds[38]. Hence we can bound h by 1 second in the worst case. This bound also takes into account multiple IBGP routers in the AS [5]. Hence the delay imposed at each router by the pseudo ordering algorithm is 12 seconds. [6]

[5] In this case we need to take into consideration that the message may traverse two non adjacent IBGP routers within the AS, and hence traverse multiple IGP routers. However, it is concretive to assume that the delay between them is bounded by a few hundred of milliseconds ([39] and observation using Netdimes database [37])

[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.

According to the new rule, every router delays the messages by Dh, hence the convergence time is $O(D^2h + Dh)$ in the worst case and $O(D^2h)$ in the best case.

We show now that the worst case complexity is less than that of BGP: The worst case convergence time of the pseudo-ordering algorithm $= D^2h + D \cdot h \sim 12 \cdot 13$, where in BGP $= D \cdot MRAI \sim 12 \cdot 30$.

In the average case, it is as follows: in most BGP implementations [2], an announcement message is delayed by $MRAI = 30$ seconds per peer, rather than per destination. Therefore, an announcement message is delayed by 15 seconds on the average at each router. In our algorithm the delay $= D^2h + Dh \sim 12 \cdot 13$ where in BGP $= D \cdot \frac{MRAI}{2} \sim 12 \cdot 15$. [7]

4.2 Adaptive Pseudo Ordering ($Delta{=}min(lh, Dh)$)

The pseudo ordering imposes a fixed delay at each router. We can improve the delay further by linking the delay on each announcement to the *ASpath* field length carried in that announcement. The idea is that a message that announces a route of length l needs to wait at a router only $l \cdot h$ seconds, but not more than Dh seconds, thus enabling all competing messages of a shorter path length to arrive earlier at the router and enables it to send only the preferred one. The limitation of Dh is in order to handle the *down* event, where the router may explore impossible paths with length larger than D.

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 4.3 *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. According to the shortest path metric policy, it first receives and announces m_g and only then it receives m_s.

[7] Our modification increases the best case, since in the theoretically best case in the original MRAI 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 MRAI the rule is per peer, and studies [2, 40] show that the rule is always set since routers send at least one message per peer every 30 seconds.

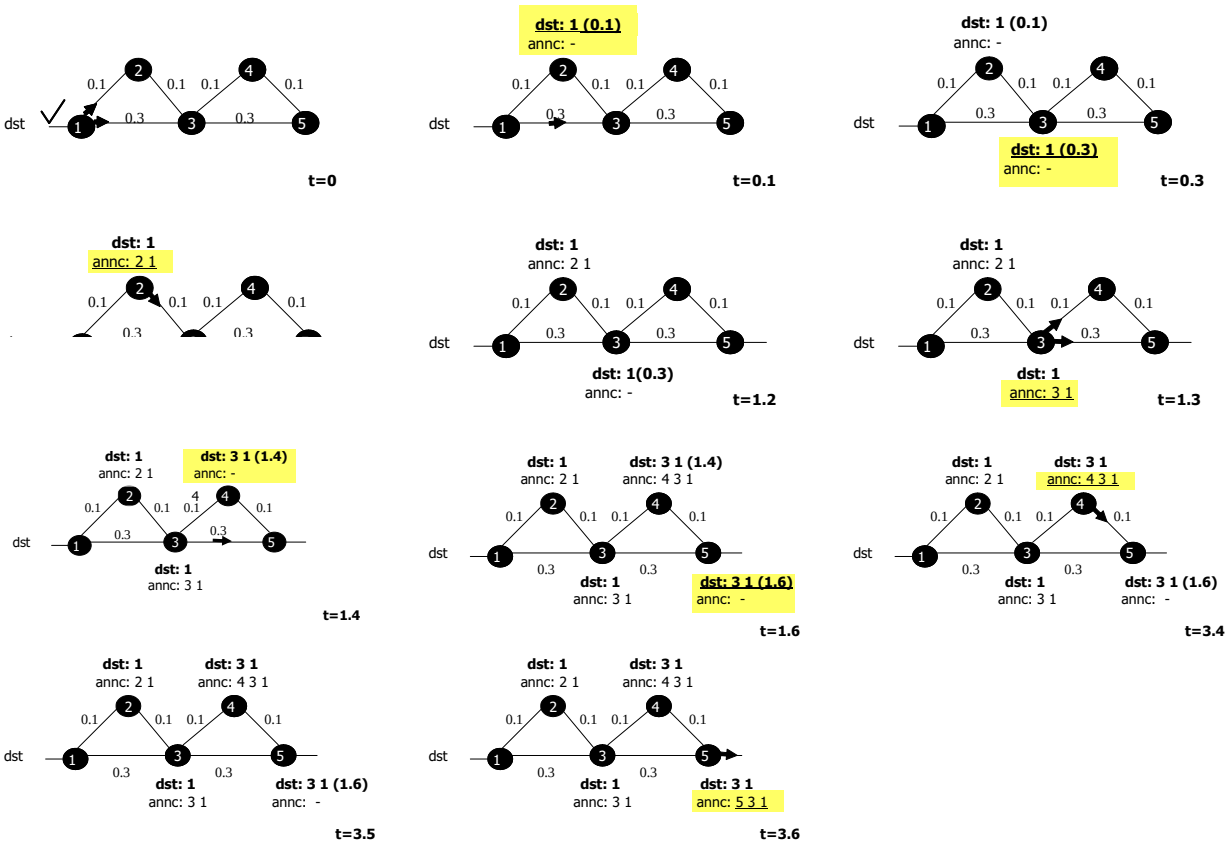


Figure 2: The scenario of Figure 1 using adaptive pseudo ordering algorithm with $h = 1$.

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 $\sum_{i=1}^{i=g} i \cdot h$.

This is a contradiction, since it is after the time it received the more preferred route s: $\sum_{i=1}^{i=s-1} i \cdot h + s \cdot h \leq \sum_{i=1}^{i=g} i \cdot h$ (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.*

Algorithm	Time	Message
Synchronous BGP	Dh	E
BGP with MRAI	$MRAI \cdot D$	DE
Pseudo Ordering	$D^2h + Dh (\sim\leq MRAI \cdot D)$	E
Adaptive Pseduo Ordering	$\frac{D^2h+3 \cdot Dh}{2} (\sim\leq \frac{MRAI \cdot D}{2})$	E

Table 1: The Convergence Complexity (Time and Message) of *up* event where h is the bound on one hop delay, D is the Internet Diameter, and E the number of the links between ASes in the internet.

The proof is straight forward. ∎

Hence, the convergence time is almost half of the convergence time of the pseudo ordering rule (which converges at $D^2h + Dh$) and thus also of BGP.

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 $l \cdot h$ seconds 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 1 summarize the worst case analysis of the current status of BGP in *up* event and the analysis of our algorithm.

5 Simulation

In this section we use simulation to further evaluate the effect of the basic and adaptive pseudo ordering algorithms on the convergence characteristics of BGP in the average case for the different event types *(up,down,shorter,longer)* and for the different BGP flavors (per peer,per destination). It is difficult to analyze the average case of BGP since the *MRAI* rule's performance depends on the status of the network, including the topology and the initial value of *MRAI*. In order to analyze the average case, we run simulations of the original BGP and our basic and enhanced adaptive pseudo ordering algorithm on SSFNet simulator on various

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

[9] In section 6 we explain how we can decide which sequence of messages about the same destination are trigged by the same up event.

topologies and *MRAI* values. The SSFNet simulator was patched with 'Framework for Traffic Engineering Protocols for SSFNet'-patch [41].

We ran the simulation on four different topologies as depicted in Table 2: 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, the Internet topology that was used also in [1, 42, 43, 44] and the 208-node Internet core topology 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 *MRAI* has been set is chosen randomly between 0-30 seconds. We repeated the test 1000 times and averaged the results.

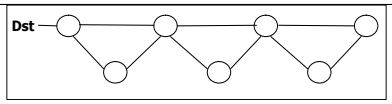
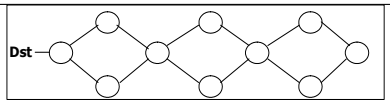
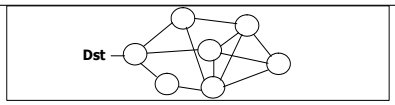
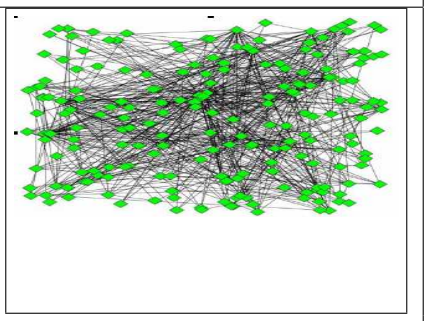
Topology	Figure
Triangle	
Diamond	
Internet	
Internet with 208 nodes	

Table 2: The four different topologies that are used for the SSFNet simulation runs.

First we evaluate our two algorithms, the basic pseudo-ordering algorithm, termed *basic* and the adaptive pseudo-ordering algorithm, termed *adaptive* on a real-like topology, the 208-node Internet core topology. For the basic version, we set the delay *Delta* to 15 seconds, and measure the number of messages and time during the convergence process of either *up, down,*

shorter or *longer* events. The results can be seen in Table 3. Both algorithm outperform the classic BGP algorithm (with the MRAI timer set per peer), in both message volume and time to convergence, for each of the events, with the most noted results for the *up* and *shorter* events. While the message volume is of the same scale for both of our algorithms, it is clear that time wise the adaptive algorithm is superior.

	Messages				Time			
Event	BGP	Basic	Adaptive	Improve	BGP	Basic	Adaptive	Improve
Up	1548	986	986	36%	121	91	23	81%
Down	42615	39789	37261	12.6%	1158	679	587	49.3%
Shorter	1509	1109	1104	26.84%	118	92	23	80.5%
Longer	2209	2031	1942	12.1%	130	105	30	76.9%

Table 3: The average message complexity and the average time convergence during different events *(Up,Down,Shorter,Longer)* for the 208-node Internet core topology using BGP per peer ,Basic Pseudo Ordering algorithm and Adaptive Pseudo Ordering algorithm .

To further evaluate our adaptive algorithm, we compare it with two other BGP flavors: The common practice version, with a constant delay of *MRAI* per peer, and a version with a constant delay of *MRAI* per destination. In Table 4 we summarize our results for the convergence properties during *up* events. It can be seen that as predicted, the adaptive pseudo-ordering modification does not create any redundant messages due to path exploration, and thus halves the message volume in all topologies during the convergence process. The per *dst* performance is considerably worse, with up to 161% redundant messages. The adaptive pseudo-ordering algorithm achieves fast convergence time, which is even better than the performance of BGP when *MRAI* is implemented per *dst* (the per peer implementation is dramatically inferior to the per *dst* one). It can be seen that *MRAI* per *dst* has a tendency to create a large number of redundant messages, twice as many as in the classic BGP implementation, where the *MRAI* timer is almost always set. This setting delays the sending at each node and increases the overall delay, but eliminates some of the path exploration, resulting a smaller message volume over longer time periods.

Table 5 shows the results of the simulation for *up, down, shorter* and *longer* events on the different topologies, with different initial values for the *MRAI* in the classic version (per peer)

	Messages				Time			
	MRAI per		Adaptive	Improve	MRAI per		Adaptive	Improve
Topology	Peer	Dst		(per Dst)	Peer	Dst		(per Dst)
Triangle	13	17	10	41.17%	48	11	8	27.27%
Diamond	20	25	15	40%	81	29	23	14.8%
Internet	23	29	10	65.5%	40	27	4	85.18%
Internet208	1548	2571	986	61.65%	121	96	23	76.04%

Table 4: Simulation results summary for the different topologies during an up event. The table shows the average number of messages and time till convergence for each of the following algorithms: Common BGP, named *MRAI per peer*, with a constant *minRouterAdver* delay per peer; BGP with a constant *minRouterAdver* delay per destination, name *MRAI per dst*, and BGP with the adaptive pseudo ordering modification, named *Adaptive*.

[10] The table shows that while the most notable improvement in convergence properties exists in the cases of *up* and *shorter* events, the pseudo-ordering rule, when not improving BGP convergence properties, does not harm them. While our implementation deals with the race conditions during up and shorter events, we can partly help with the convergence of *down* and *longer* events, as the table shows. In those cases the main reason for the long convergence is not the race condition but the counting-to-infinity like problem. The reason for the improvement in those cases is the fact that many announcements are involved in *down* and *longer* events, and our algorithm reduces some of them. In a previous work [31] we showed a detailed solution for the counting to infinity problem.

To conclude: the enhanced adaptive pseudo-ordering BGP reduces in *up* event the message complexity by up to 35% and the convergence time by up to 82% compared with current BGP implementation with *MRAI* per peer. Averaging on all type of events we receive reduction of up to 14% in the total message complexity (22% for the 208-node topology , which is the more realistic topology) and still a substantial reduction in the convergence time (by 65%).

To obtain a realistic understanding of the possible gain from implementing our modification, we evaluate the convergence properties when the modification is implemented incrementally in

[10] 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.

the network, i.e., the fraction of BGP speakers that implement the modification is incremented in each run of the simulation. Figure 3 shows the message volume during convergence and the time to convergence against the percentage of BGP speakers implementing our modifications. The rest of the speakers run the classic BGP. At the beginning, all BGP speakers run the classic BGP, and at the end of the simulation, they all run BGP with the adaptive pseudo ordering algorithm. The results, summarized in Figure 3, show that even with only 10% of the BGP speakers implementing the adaptive pseudo ordering algorithm, there is a substantial gain in both the time and message volume during convergence. The gain improves with the increase in the number of BGP speakers implementing our modification. Interestingly, when 60% of the BGP speakers run with our modification, the message volume is almost at its best value, reaching its best when 90% of the BGP speakers run with our modification. Time wise, the more routers implementing our modification the better the convergence time is, and with a 100% of the BGP speakers running with our modification, convergence time is less than third the time with the classic BGP implementation.

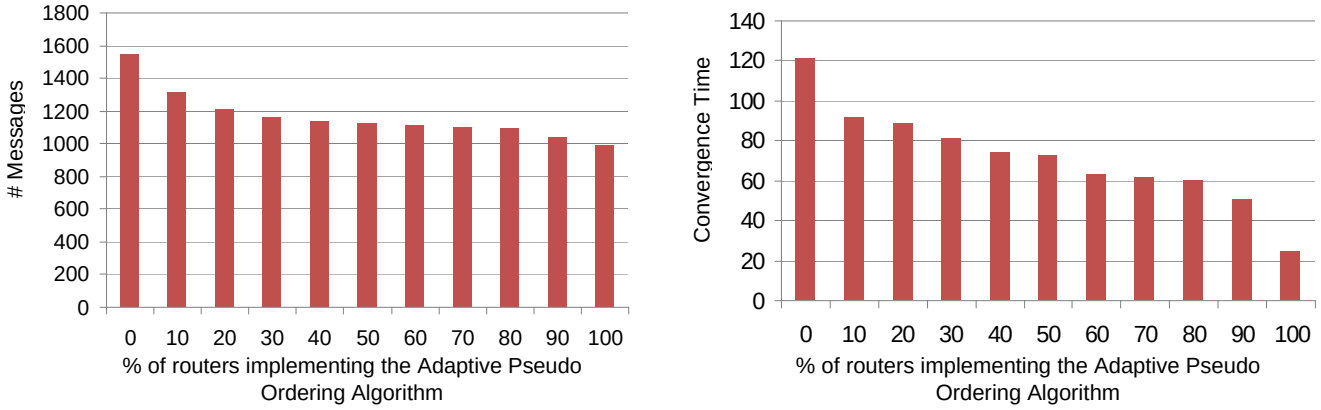


Figure 3: These graphs show the convergence behavior during an incremental deployment of the Adaptive Pseudo Ordering Algorithm in the 208-node Internet core topology. (a)Number of messages.(b) Time of convergence.

6 Empirical Investigation

In this section we analyze empirical results of *up* events in real BGP raw data dumps, obtained from [45]. The data, gathered from several of the RIPE RIS project Remote Route Collectors

Topology	Event	Messages			Time		
		BGP	Adaptive	Improve	BGP	Adaptive	Improve
Triangle	Up	13	10	23%	48	8	83%
Triangle	Down	17	17	0%	3	3	0%
Triangle	Shorter	11	10	9%	48	10	79%
Triangle	Longer	20	20	0%	65	23	65%
Diamond	Up	20	15	25%	81	23	72%
Diamond	Down	19	19	0%	4	4	0%
Diamond	Shorter	16	15	6%	88	29	67%
Diamond	Longer	13	13	0%	72	27	63%
Internet	Up	23	10	57%	40	4	90%
Internet	Down	45	41	9%	35	9	74%
Internet	Shorter	11	11	0%	27	6	78%
Internet	Longer	10	10	0%	41	9	78%
Internet208	Up	1548	986	36%	121	23	81%
Internet208	Down	42615	37261	13%	1158	587	49%
Internet208	Shorter	1509	1104	27%	118	23	81%
Internet208	Longer	2209	1942	12%	130	30	77%

Table 5: The average message complexity, the average time convergence and their % of improvement in different events *(Up,Down,Shorter,Longer)* for different topologies using BGP per peer and Adaptive Pseudo-Ordering algorithm.

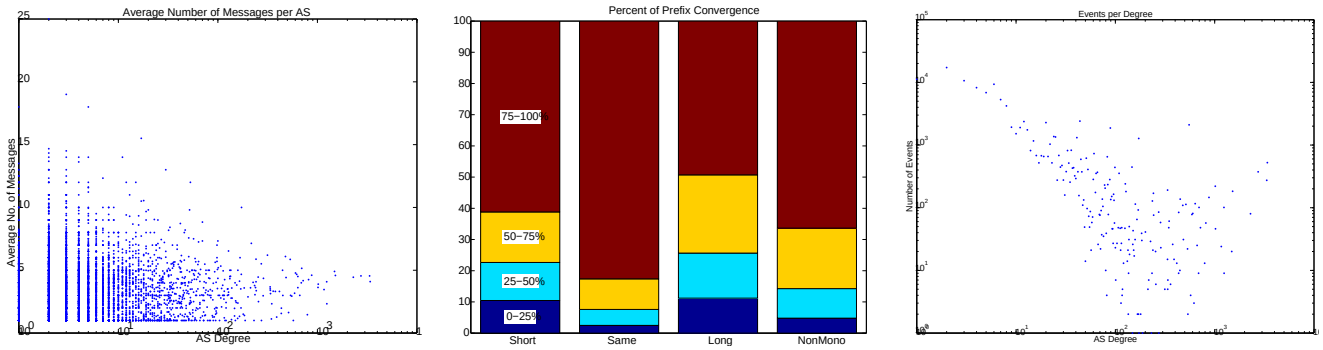
(RRC's) throughout the world, represents the sequence of *up* events occurring in the first 14 days of June 2005 as seen from each of these collectors. We investigate the characteristics of *up* events and analyze the potential effect of implementing the BGP Pseudo-Ordering Algorithm. For a better understanding of the data, we analyze it using the AS degree, taken from a map constructed from the union of the routeview project [46] and the Dimes project [47] 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 [14] it is shown that consecutive updates arriving less than 70 seconds apart belong to the same event with a high probability. Following this rule of

RRC	Number *Up* event	Avg msg	c_shorter		c_same		c_longer		c_nonMono		Potential Improve.
			%	Avg. msg	%	Avg. msg	%	Avg. msg	%	Avg. msg	
2	36127	1.11	2.24	2.18	97.16	1.08	0.43	2.47	0.17	3.27	9.35%
5	140846	4.28	4.62	5.39	67.54	2.66	5.89	4.47	21.94	8.99	30.92%
7	1749	1.56	3.77	2.18	84.51	1.16	5.43	2.07	6.29	6.26	11.46%
10	142	1.18	0	0	100	1.18	0	0	0	0	15.25%
11	106678	2.44	14.19	2.67	33.18	1.54	33.63	2.32	19	4.06	17.05%
All	285542	3.18	7.9	3.95	58.6	2.03	15.5	3.4	18	6.4	26%

Table 6: Characteristics of *Up* events convergence properties

thumb, we identify the reattachment of a subnetwork if 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. [11]



(a) (b) (c)

Figure 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

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 $ASpath$. [12]). c_{same}-

[11] 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.*

[12] For example the first announcement is with $ASpath$ of length 5 and the second announcement to the same

Describes all the events in which BGP converges monotonically, and all its announcements carry a different *ASpath* of a specific constant length. Includes also all cases in which only one announcement is sent during the event. c_{longer}- Describes all the events in which BGP 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 6 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} [13] with more than one message, and receive that the potential improvement is, on average, 26%. We find that the average number of messages is between 1.11 to 4.28, averaged to 3.18. This finding seems to contradict previous findings, calculating the number of messages sent during events generated by the RIPE Beacons [14]. 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* events implies that route flap damping (RFD [12]) mechanism, that is commonly implemented in today's routers[14], 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 [12] a route that changes its *ASpath* more than four times, will be suppressed for up to half an 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 hour the route fails, RFD will prevent propagation of the new state in the Internet, causing packets to route incorrectly and be lost.

A closer look at the data given in Table 6 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.[15] 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

destination is with *ASpath* of length 3

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

[14]In [48] it is claimed, that *Originally proposed in the early days of the commercial Internet, route flap damping is generally assumed by the operator community to be widely deployed in today's infrastructure*

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

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 to 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). [16]

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 [20] 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[17]. While our modification cannot reduce messages that result from path exploration due to race conditions with complex routing policy, it does succeed in reducing approximately 26% of the overall messages in *up* events.

[16] 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.

[17] 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

7 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 [49]. 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.

Concerning the load on the router, our technique requires the setting of a timer per destination. But since it reduces drastically the number of messages sent per event, the overall load on the router certainly does not increases compared to traditional BGP implementations with timer per destination, while improving the message and time complexity.

To further improve the CPU load time, we also suggest the following optimization. The key idea is to reduce the accuracy of the wait interval each message incurs in the router, from the precise time it should be sent according to our algorithm (which requires a timer per destination) to a precision with granularity of $\frac{1}{k}$ seconds, where k is the optimization parameter. This optimization reduces the number of timers to the following: The maximum waiting time in seconds multiplied by k, plus one, i.e. to $30k + 1$. In this scheme, if a message was to be sent in time δ, it will now be sent in time t, such that exists i, so $t = \frac{i}{k}$ and $\frac{i-1}{k} < \delta \leq \frac{i}{k} = t$. Note that this rounding would impose an extra delay to a message of up to $\frac{1}{k}$ seconds per link. This has impact on the estimate of h, the maximal link delay, and hence its value should be updated by an additional $\frac{1}{k}$ seconds. This change does not alter any of our claims, and a router would still send only one message per event. For example, for $k = 5$ seconds we get that $h = h + 0.2 = 1.2$ seconds and we need to maintain only 1501 timers. Hence the algorithm has a negligible effect on the algorithm performance.

Implementation wise, the router maintains a cyclic array of $30 \cdot k + 1$ queues and a pointer to the current queue, where the i^{th} queue after the current queue would maintain the set of messages that need to be sent in the $\frac{i}{k}$ seconds. Each $\frac{1}{k}$ seconds the router sends all the messages in the current queue and frees all the resources of those messages, while moving the current queue pointer to point to the next queue. To handle the case of a new message per destination that outdates a previous message in the queue, we do the following: The entry that corresponds to the destination in the forwarding table maintains also a pointer to the next message per the destination that should be sent, if such exists. If the best route to the

destination changes, the router checks if a previous message per the destination exists in the queues (i.e., it has not been sent yet), in which case the message would be deleted. The new message is then inserted to the $lk + 1$ queue with respect to the current queue, where $l = 30$ for the basic version, and for the adaptive version l is the length of the $ASpath$ of the message that triggered this announcement.

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.

7.1 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 [9]. In Section 6, we analyze real BGP dumps and estimate that in real life BGP, with its complicated policy, our algorithm can reduce the message complexity by 26%. Surprisingly, we show also that 66% of the *up* events converge to the shortest path. In this section we show why the policy, in which routers select the shortest policy 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 [50]: 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 an 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, or what we have termed in Section 3: Shortest policy paths, i.e., paths that conform to the no valley rule ([9]) and have the shortest AS-Path field. In the rest of this section, when we refer to a shortest path, it is a shortest policy path.

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 [51]. 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.

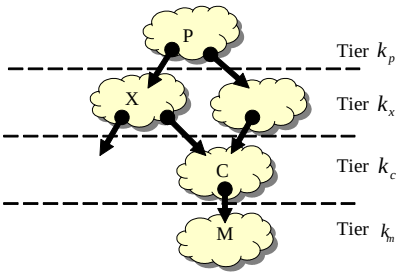


Figure 5: Illustration of Claim 5

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 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 5 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 5 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 [18] 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.

7.2 Implementing the protocol in IBGP (ASes may have more than one BGP router)

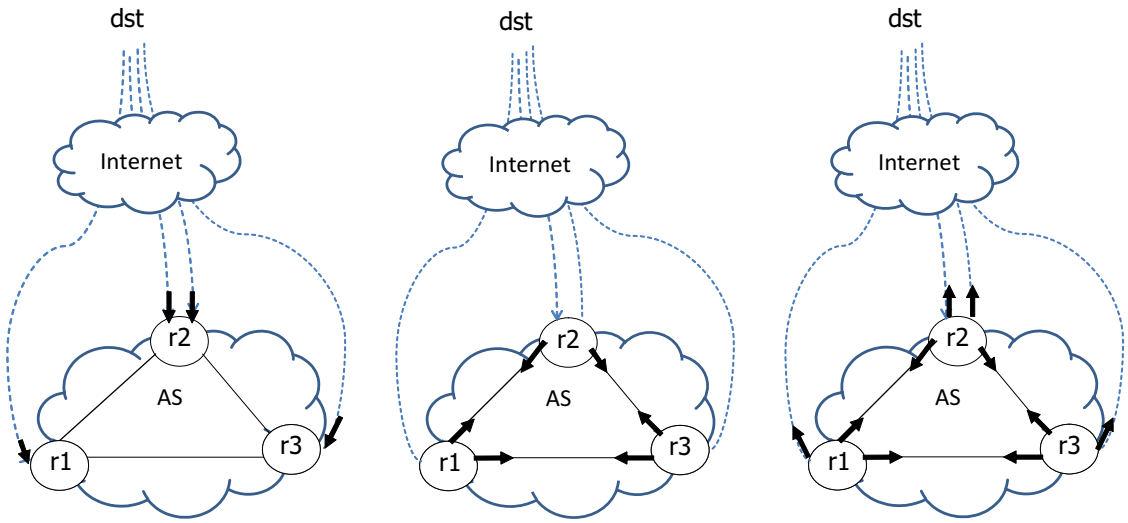


Figure 6: (a) The router waits lh seconds until converging to the best route learnt from its EBGP neighbors (b) First phase: The router sends IBGP message about its new best route, if either lh seconds elapsed from the change, or it has received an IBGP route updated from one of its neighbors (c) Second phase: After 2h seconds from the previous sent IBGP message, the router surely converged to the best route and thus sends it to all its neighbors (EBGP and IBGP)

[18] 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.

When the AS has multiple BGP routers, a full IBGP clique runs inside the AS, or a Route Reflector is used. We first discussed the full IBGP mesh case. In this scenario, the modified algorithm works in two phases. In the first phase, each router converges to 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, see Figures 6 (a) and (b)). In the second phase, the router converges to the preferred route after receiving the relevant routes from all its IBGP peers. It then announces the preferred route to all of it peers (including EBGP peers, see Figure 6 (c)). 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. In the second phase, the final announcement the router sends to all of its peers (IBGP and EBGP) is delayed 2h time from the last IBGP announcement in the first phase. The idea is to use the delay of $2h$, 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 back their preferred route, since the received message triggers an IBGP router to send the first phase message if it has not already sent it.

In some Cisco routers, IBGP MRAI is set to 0. I.e., each change in the EBGP information, is immediately propagated to all others IBGP routers. Our suggested solution above indeed adds a delay of order of Dh (or lh) until the information is propagated inside the AS by the IBGP message, but also decreases the number of IBGP messages. Another possible solution is to add no delay, by allowing multiple IBGP messages (as the current state where MRAI=0). Again the solution works with two phases, only in the first phase the EBGP router sends an IBGP messages each time its best route changes due to the EBGP message. In the second phase, the final announcement to all of its neighbors (EBGP and IBGP) is delayed by $Dh + h$ (in the basic version) or $(l + 1)h$ in the adaptive version from the first sent IBGP message. The $Dh + h$ delay ensures, that only one EBGP message is sent. The extra h is due to the extra one link maximum delay of a message transfer inside the AS.

Note that we suggest here two extreme solutions to the IBGP routers, one that ensure only two IBGP messages per event but with the cost of a delay in information propagation inside the IBGP, and the second allows multiple IBGP messages, but with no delay in the propagation inside the AS. We note also a hybrid solution, in which the Dh delay is divided

between the two phases,exists . In this case there will be a moderate number of IBGP messages and a moderate delay in the propagation of information between the IBGP routers.

We now consider the case of an AS using a router reflector and suggest a similar solution. When using a router reflector, the overhead of IBGP messages is minimized. Hence, each time an EBGP router receives new information that changes its best route an IBGP message to the route reflector is immediately sent. The route reflector would wait $Dh + h$ from the first IBGP message for a destination, until it would send the best route to the IBGP routers, which then in turn would immediately send a message to their EBGP neighbors. Note that in case of k hierarchical route reflectors a similar solution can be used where the $Dh + kh$ delay is incurred only in the top route reflector in the hierarchy. It is reasonable to bound $k = 2$, since hierarchies greater than 2 are rare.

The proof and the accuracy of our algorithm is based on the bounds of the time it takes a message to traverse an AS or a link. Hence, the case where there is only one BGP router per AS should be changed correspondingly to the case where there are multiple BGP routers in the AS (which in all our solutions can be bounded by $Dh + 2h$). If there is only one BGP router at the AS, the router should wait $Dh + 2h$ (in the basic version) or $(l + 2)h$ 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 $2hL$ and hence minor.

7.3 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.

7.4 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 case is the usage of *ASpath* with aggregation [52]. Our adaptive solution requires that the

ASpath reveals the actual length of the route. Hence, in order to use the adaptive solution we require that the BGP implementation in the Internet will use the option of aggregation with the use of AS_SET [17]. In this case, the regular implementation of BGP counts *ASpath* length correctly: the *ASpath* length equals to the number of ASes in the *ASpath* plus one; which is the AS that appears in the AS SET, which the route goes through. Otherwise, if the aggregation without AS SET option is allowed in the Internet than we can use only the basic pseudo ordering algorithm, which still improves the current BGP implementation.

Another alternative is to reveal the accurate AS path by using a transitive community in BGP message that counts the number of ASes from the aggregate network , which each AS along the route updates.

8 Concluding Remarks

In this paper we show that race conditions in path exploration, often initiated at the edge of the network, is one of the big contributors to the flow of BGP updates in the Internet backbone.

We suggest minor modification rules to the MRAI timer in BGP, that eliminate path exploration in cases where the shortest valid route policy is used, and reduce the number of messages in heterogenous policy routing (when policy constraints prevent from following the shortest path). Our modification introduces a productive way to prevent the common case of path exploration while improving the convergence delay, by dividing the delay introduced by the MRAI timer along the route in a different way. Using RAW BGP dumps we estimate that approximately 26% 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] H. Tangmunarunkit, R. Govindan, and S. Shenker, "Internet path inflation due to policy routing," *SPIE ITCom*, 2001.

[6] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin, "The impact of routing policy on internet paths," *IEEE INFOCOM*, 2001.

[7] L. Gao and F. Wang, "The extent of as path inflation by routing policies," *IEEE Global Internet Symposium*, 2002.

[8] N. Spring, R. Mahajan, and T. Anderson, "Quantifying the causes of path inflation," *ACM SIGCOMM*, 2003.

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

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

[11] "SSFnet," http://www.ssfnet.org/.

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

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

[14] 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.

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

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

[17] Y. Rekhter and T. Li, "A border gateway protocol 4 (BGP4)," Jan. 2006, rFC 4271.

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

[19] 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

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

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

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

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

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

[25] 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.

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

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

[28] R. Perlman, *Interconnections, Bridges and Routers*. Addison-Wesley, 1992, 1992.

[29] B. Halabi, *Internet Routing Architectures*. New Riders Publishing, Cisco Press, 1997.

[30] 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.

[31] A. Bremler-Barr, Y. Afek, and S. Schwartz, "Improved BGP convergence via ghost flushing," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 10, pp. 1933– 1948, Dec. 2004.

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

[33] A. Bremler-Barr, N. Chen, J. Kangasharju, O. Mokryn, and Y. Shavitt, "Bringing order to BGP: Decreasing convergence message complexity (brief announcment)," in *PODC*, 2007.

[34] L. Gao and J. Rexford, "Stable internet routing without global coordination," in *SIGMETRICS*, June 2000.

[35] "CISCO products command reference," http://www.cisco.com/en/US/docs/ios/12_3/iproute/command/reference/ip2_n1g.html.

[36] R. Albert and A.-L. Barabási, "Topology of evolving networks: local events and universality," *Physical Review Letters*, vol. 85, no. 24, pp. 5234–5237, 11 Dec. 2000.

[37] "The dimes project," http://www.netdimes.org.

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

[39] "Att delay," http://ipnetwork.bgtmo.ip.att.net/pws/network_delay.html.

[40] T. G. Griffin, "Tutorial: An introduction to interdomain routing and BGP," Aug. 2001, in conjunction with SIGCOMM 2001.

[41] "Software developed by Anja Feldmann's research group at TU Munchen," http://www.net.in.tum.de/software/.

[42] B. Premore, "Multi-AS topologies from BGP routing tables," http://www.ssfnet.org/Exchange/gallery/asgraph/index.html.

[43] M. Lad, X. Zhao, B. Zhang, D. Massey, and L. Zhang, "Analysis of BGP update surge during Slammer worm," 2003.

[44] G. P. Rodrigo, "Convergence time reduction in the BGP4 routing protocol using the "ghost-flushing" technique and other proposals," 2004.

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

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

[47] Y. Shavitt and E. Shir, "DIMES: Let the internet measure itself," *ACM SIGCOMM Computer Communications Review*, vol. 35, no. 5, pp. 71–74, 2005.

[48] Z. Mao, R. Govindan, G. Varghese, and R. Katz, "Route flap damping exacerbates Internet routing convergence," in *Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications.* ACM Press New York, NY, USA, 2002, pp. 221–233.

[49] "Juniper techpubs: out delay command," http://www.juniper.net/techpubs/software/junos/junos91/swconfig-routing/out-delay.html.

[50] M. Caesar and J. Rexford, "BGP policies in ISP networks," *IEEE Network Magazine, special issue on interdomain routing*, 2005.

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

[52] "Understanding route aggregation in BGP, document id: 5441," http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080094826.shtml.

Appendix A

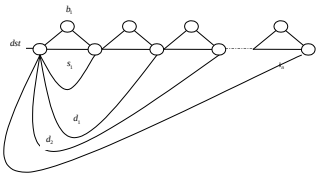


Figure 7: Example of exponential message complexity in up event

Lemma 8.1 *The worst case message complexity of an up event without the*MRAI *rule is exponential in n and the time complexity is Dh, where D is the diameter and h is the maximal link delay.*

Proof: Let us look at the graph in Figure 7, 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, there are two optional paths that can be traversed between the pair of nodes: A bypass, in which two edges are traversed, or the direct edge between the nodes. 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 direct 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). We assume a complex BGP policy where the routers in the network prefer routes that use direct links near the destination. I.e., a path $1{***}\ldots*$ (where * can be 0 or 1) is always preferred over any path of type $0{***}\ldots*$. Similarly, a path $11{***}\ldots*$ is always preferred over any path of type $10{***}\ldots*$. Since every such new route requires at least one message, we receive $2^{(\frac{n}{2})}$ messages. ∎