# Improving BGP Convergence Delay for Large-Scale Failures

Amit Sahoo
Dept. of Computer Science
UC Davis
Davis, CA 95616, USA
asahoo@ucdavis.edu

Krishna Kant
Intel Corporation
Hillsboro, OR 97124, USA
krishna.kant@intel.com

Prasant Mohapatra
Dept. of Computer Science
UC Davis
Davis, CA 95616, USA
pmohapatra@ucdavis.edu

## Abstract

*Border Gateway Protocol (BGP) is the standard routing protocol used in the Internet for routing packets between the Autonomous Systems (ASes). It is known that BGP can take hundreds of seconds to converge after isolated failures. We have also observed that the convergence delay can be even greater for large-scale failures. In this study, we first investigate some of the factors affecting the convergence delay and their relative impacts. We observe that the Minimum Route Advertisement Interval (MRAI) and the processing overhead at the routers during the re-convergence have a significant effect on the BGP recovery time. We propose a couple of new schemes to reduce processing overload at BGP routers during large failures, which in turn leads to decreased convergence delays. We show that these schemes combined with the tuning of the MRAI value decrease the BGP convergence delay significantly, and can thus limit the impact of large scale failures in the Internet.*

## 1 Introduction

BGP [1, 2, 3] is the predominant protocol used for inter-domain routing in the Internet. BGP belongs to the class of *path vector* routing protocols wherein each node advertises the “best” route for each destination to each of it’s neighbors. A BGP node stores all the paths sent to it by its neighbors but uses and advertises only the one that is “best” according to some criteria. If this primary path fails, BGP selects the next best backup route, which is then advertised to its neighbors. However, there is no guarantee that the backup route is still valid. In case the backup route has also failed, it will be replaced only after a withdrawal is sent by the neighbor which advertised it. At that time, another backup route will be chosen. This absence of information about the validity of a route can cause BGP to go through a number of backup routes before selecting a stable one. Thus, there can be a considerable delay before the cycle of withdrawals/advertisements ends and all BGP nodes have a valid and stable path to the destination.

Several studies [4, 5, 6, 8, 7, 9, 10] have been carried out to study the fault tolerance and recovery characteristics of BGP. In particular, it was shown by Labovitz et al. [5] that the convergence delay for isolated route withdrawals can be more than 3 minutes in 30% of the cases, and could be as high as 15 minutes. Models [6, 7, 8] have been developed to estimate the BGP convergence delay, but the complexity of analysis has meant that simplifying assumptions need to be made, and most of the work in this field has been concentrated on single failures and simple networks.

Large scale/multiple failures in the Internet have not been studied enough; primarily because of their low probability of occurrence and the complexity involved in their analysis. As high value and more complex services pervade the Internet, large scale failures become more important for several reasons: (a) much more serious consequences of a given large scale failure, (b) new failure scenarios resulting from unmastered complexity of interactions, and (c) greater incentive on the part of adversaries (hackers, terrorists, etc.) to cause wide-spread system shutdowns or denial of service attacks.

From the BGP perspective, a large scale failure not only disables a large number of endpoints and routers, but also results in a flood of route updates to surviving BGP routers and which can increase the convergence delay, as reported in our previous work on the subject [11]. We also observed that the delay increased as the network grew in size, which means that the failure of a relatively small number of nodes can cause a long period of instability in a large network. As recent events have shown, communication networks are needed the most at the time of crisis, and therefore a short recovery time after a failure is highly desirable. Hence it is vital that we have a good understanding of the behavior of the Internet connectivity after a large scale failure.

In this paper we look at a number of factors affecting the convergence delay and propose methods to minimize it. We have studied in detail the impact of the Minimum

Route Advertisement Interval (MRAI) [1] on the BGP recovery time. The relative impacts of the size of failures, topological characteristics, and the update processing overheads are also analyzed. Based on the quantitative studies, we propose a scheme to dynamically select the MRAI so that the rate of generation of update messages during large scale failures can be controlled. We also propose a novel batching scheme that reduces the number of route advertisements during periods of instability by suppressing the effect of update messages that are stale or redundant. We show that the batching scheme can substantially reduce the convergence delays (by a factor of 3 or more). The convergence delays are reduced even further if we combine the proposed batching and the dynamic MRAI schemes.

The rest of the paper is organized as follows. We talk about the various factors affecting the BGP convergence delay in Section 2. Section 3 outlines the details about the tools and the configurations that we used for our experiments. We present and analyze the results of our experiments in Section 4. We summarize the results and talk about future work in Section 5 and we conclude with references.

## 2 BGP Convergence Delay

Previous works [5, 8] have concluded that the Minimum Route Advertisement Interval (MRAI) [1, 3] is one of the most important BGP configuration parameters affecting the convergence delay. The MRAI governs the rate at which a node can send route advertisements to a neighbor. After a node has sent an advertisement to a neighbor, it has to wait for at least the MRAI before it can send a new route advertisement for the *same* destination to the *same* neighbor. The straightforward way to implement the MRAI would be on a per-destination basis, i.e. maintain a separate timer for each destination and each neighbor. The timer is started when the router sends an update for the corresponding destination to the neighbor in question. Thus, the next update can be sent only after the timer has expired. However the large number of destinations in the Internet makes this approach unviable and a per-peer scheme is more prevalent in the Internet today. In the per-peer scheme, the router maintains just one timer per neighbor and that timer is used to control the updates for all the destinations. This makes the scheme more scalable.

Griffin and Premore [7] studied the effect of MRAI on the convergence delay after a fault in simple BGP networks. They found that as the MRAI is increased, the convergence time first goes down to a minimum and then increases linearly. They observed that the optimal MRAI was dependent on the size of the network, the configured processing delay for the update messages, and the path-vector scheme in use. In particular, they found that the optimal value increased with an increase in the processing delay and the network size. The authors also looked at the variation in the number of update messages as the MRAI was increased and found that the message count decreased until the MRAI was close to the optimal value and then remained constant. The authors concluded that the default value of 30 seconds for the MRAI is “somewhat arbitrary” and in the ideal scenario we would have a different MRAI for each AS.

One of the factors responsible for the behavior observed by Griffin and Premore [7] is the processing overhead of BGP updates. Let’s assume that a node *A* sends an update to a neighbor *B* at time *t*. Let’s also assume that the MRAI is high enough so that all incoming update messages have been processed by the time *t+ MRAI*. If the MRAI is increased further, it means that the nodes have to wait longer before sending the update messages and this increases the convergence delay. Thus, in this phase, the number of update messages sent remains roughly constant and the delay increases linearly.

If we start decreasing the MRAI value, updates are generated at a faster rate and the processing load at the nodes will increase. So a node could possibly send out an update to a neighbor before it has processed all the queued update messages. If one of the remaining update messages changes the route which was just advertised, then another update needs to be sent. Not only does the neighbor have to process an extra update message, it might also send an extra update message to its peers, thus increasing workload on other downstream nodes. This ultimately leads to higher convergence delay.

The BGP convergence delay is also dependent on a number of other parameters such as the size of the network, the size or extent of the failure, the average degree of the nodes, the degree distribution, the processing overhead, etc. [1, 3] In our previous study [11], we looked at a number of factors affecting the BGP convergence delay. For those experiments we used and MRAI of 30 seconds (default value used in the Internet) and we did not simulate any processing overhead. Therefore the results we obtained were similar to what one would observe if the MRAI were much greater than the optimal value. We found that the convergence delay increased as the size of the network and the average degree of the nodes was increased. Finally we observed that the convergence delay was reduced if the degree distribution was non-uniform (combination of high and low degree nodes).

We now talk about some other related work in brief. Labovitz et al. [5] developed a model for BGP convergence and showed that the convergence delay after a route withdrawal in a complete graph with *n* nodes is *(n-3)*MRAI* at best and *O(n!)* at worst. They [6] later extended their model and determined that the upper bound for the time required for a route to converge is dependent on the MRAI and the length of the longest path from the source to the destination.

Pei et al. [8] developed a more general model in which they also considered the processing delay for an update message. They considered scenarios where the BGP nodes were not overloaded and derived upper bounds for the convergence delay for such scenarios. While these models can be used to determine the limits of the convergence delay for simple failures, it is still not possible to estimate the delay for arbitrary failures in arbitrary networks.

Deshpande and Sikdar [12] proposed a couple of MRAI related methods to reduce the convergence delay. The first method cancels a running MRAI timer if that can improve the convergence delay and the second method uses the MRAI for a destination only if the route for that destination has changed at least a specified number of times. The authors showed that these schemes reduced the convergence delay; however the number of update messages went up considerably.

## 3 Methodology

We used a number of different topologies for our studies. A modified version of BRITE [13] was used for topology generation and the SSFNet [14] simulator was adopted for the BGP simulations.

### 3.1 Topology Generation

BRITE can generate topologies with a configurable number of ASes and with multiple routers in each AS. BRITE supports a number of AS topology generation schemes such as Waxman [15], Albert-Barabasi [16], and GLP [17]. We modified BRITE to allow more flexible degree distributions. For most of the experiments we used topologies with simple degree distributions and with just one node per AS. This was done to minimize the effect of variations in the degree distribution and the size of the ASes on the results. We did confirm the results using more complex networks with multiple routers per AS.

For our experiments, we predominantly used a "skewed" distribution in which most of the ASes had a low number of inter-AS links while the rest had a significantly higher number of inter-AS links. In particular we used a "70-30" distribution in which 70% of the nodes had low degree and the remaining 30% had higher degree. We used this topology for most of the experiments because we found that in the real AS topology in the Internet [18], about 70% of the ASes were connected to less than 4 other ASes. An artificial degree distribution obviously gives us the freedom to modify it as we see fit, and to observe the effects of the modifications on the convergence delay. For example, we experimented with the average degree and the ratio of low to high degree nodes. We have also found that simple degree distributions generate more consistent results which makes it easier to detect trends in the results. We did however use the degree distribution from the *actual* connectivity data, to verify the results.

For most of our experiments we used 120 node topologies. This was dictated partly by the fact that the Java Virtual Machine could allocate a maximum of 1.5 GB of memory on the 32 bit machines that we used and hence we could simulate at most $\sim$250 nodes . The benefit of using the 120 node topologies was that we could verify the results using networks that were half as big (without being really small) and twice as big (still within the scope of our experimental setup). The running time for the BGP simulations with 120 node networks was also much more manageable than 200 or 250 node networks, and this allowed us to experiment with many more scenarios and schemes.

Although large scale failures could be scattered throughout the network, many failure scenarios (e.g., those caused by natural and man-made disasters) are expected to be geographically concentrated. We randomly placed all the routers on a 1000x1000 grid and then considered failures in contiguous areas of the grid (usually the center of the grid to avoid edge effects). In our previous work, we examined the impact of such geographical aspects as non-uniform location density and edge failures, but did not find the impact of these aspects very pronounced. In fact, as one might expect, as the area of failure increases, these location aspects become less and less important. For all links, we used a one way delay of 25 ms (transmission, propagation and reception delays).

For scenarios with multiple routers per AS we selected the number (1-100) of routers in an AS from a heavy tailed distribution. Studies of the real Internet topology have found that the geographical extent of an AS is strongly correlated to the AS size (i.e., number of routers in the AS) [19]. We assumed a perfect correlation and made the geographical area (the region over which the routers of an AS are placed) of an AS proportional to its size (number of routers). Internet studies also show that larger ASes are better connected [20]. Therefore, we assigned the highest inter-AS degrees to the largest ASes.

### 3.2 BGP Simulation

We used the SSFNet simulator for our experiments because it has been used extensively in the research community for large scale BGP simulations and BRITE can export topologies in the format used by SSFNet. In the simulations, the *path length* (i.e., number of hops along the route) was the only criterion used for selecting the routes and there were no policy based restrictions on route advertisements. All the timers were jittered as specified in RFC 1771 [1] resulting in a reduction of up to 25%. In our experiments the MRAI timer was applied on a per-peer basis rather than

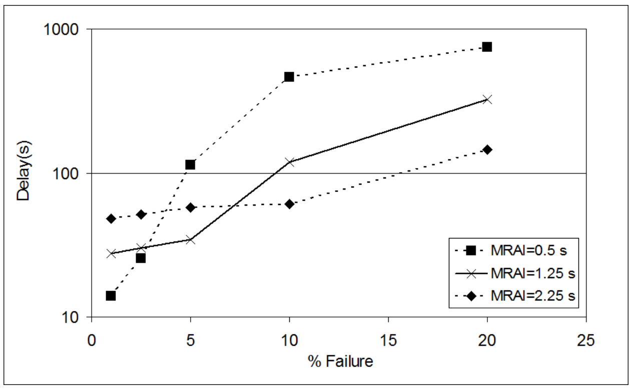

**Figure 1. Convergence delay for different sized failures**

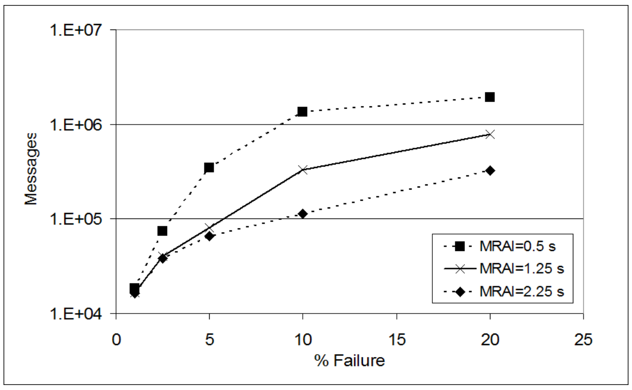

**Figure 2. Number of generated messages for different MRAI values**

a per-destination basis, as is commonly done in the Internet. The BGP update processing delay was modeled using the mechanisms available in SSFNet. In our experiments the processing delays were uniformly distributed between 1 and 30 milliseconds. The processing delays are higher than those found in the latest routers today, partly to compensate for the small size of our network. With a higher processing delay we attempt to imitate the processing overhead caused by a failure in a much larger network.

As stated earlier, we assumed a contiguous failure area for large scale failures. We further assumed that all routers and links in the failed area become unoperational. The scenarios where only the links (but not the routers) fail are unlikely for large scale failures and are not considered here. For the simple topologies, the routers failures are really AS failures since there is only one BGP router per AS.

## 4 Results

In this section we present and discuss the results of our experiments. The results that we show here were obtained using topologies with 120 ASes but we did run a smaller number of experiments with topologies of 60 and 240 ASes to verify the results. Those results exhibited the same trends as those for 120 ASes.

### 4.1 Effect of MRAI

We had already found in our previous study [11] that with MRAI=30 seconds, the convergence delay was dependent on the size of the failure. Our first goal here was to find out if variation in the MRAI value affected failures of different sizes differently. For this set of experiments we used topologies with 120 ASes/Nodes and a 70-30(See section 3.1) distribution. 70% of the ASes/nodes had a degree in the range 1-3 while the remaining 30% had degree 8. The resulting average degree was 3.8. We show the convergence delay for different sized failures with a number of MRAI values in Fig 1. We restricted the size of the failures to 20% because larger failures are probably not realistic and may take down too many routers to be interesting. From the results we can see that a low MRAI results in a low convergence delay for small failures, but the delay increases sharply as the size of the failure goes up. For higher MRAI values, the convergence delay for small failures is greater (than that for low MRAI values), but the increase in the delay for larger failures is less steep.

Fig 2 shows the number of generated messages for the three different MRAI values. The trend is similar to what we saw for the convergence delays. For small failures, the number of messages is low and about the same for all the MRAI values. The message count for MRAI=0.5 seconds shoots up as the size of the failure is increased and that is reflected in the convergence delays. The increase in the number of messages for the other two MRAI values is more gradual and hence a similar behavior is observed for the delays.

In Fig 3 we present the above results in a different way. Here we have plotted the convergence delay vs. the MRAI values for three different failure magnitudes. If we look at the curve for 5% failure we can see that the convergence delay goes down until the MRAI is equal to about 1.25 seconds, and then increases. This is similar to the results observed by Griffin and Premore [7]. We can see that increasing the MRAI beyond the optimal value ($\simeq$0.5 seconds for 1% failure) doesn't affect the number of update messages but increases the convergence delay (Figs 1 and 2). On the other hand, decreasing the MRAI below the optimal value ($\simeq$1.25 seconds for for 5% failure) increases both the number of messages and the convergence delay (Figs 1 and 2).

We see that larger failures result in more update messages which in turn lead to a higher processing load. For

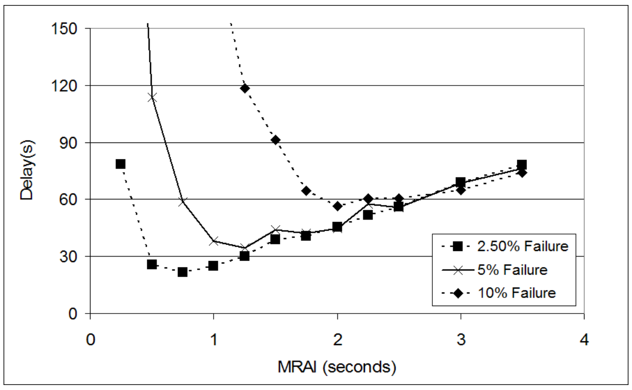

**Figure 3. Variation in convergence delay with MRAI**

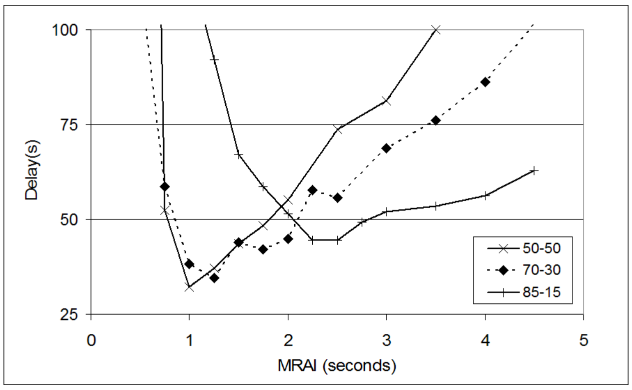

**Figure 4. Convergence delay for different topologies**

example, an MRAI value of 0.5 seconds is ideal for 1% failure but too small for 5% failures. Thus, *it is not possible to select a single "ideal" MRAI value for a network (or even an AS) if we take multiple failures into account.* This result points to potential MRAI adjustment schemes based on the extent of failure. For example, one could set the MRAI to a low value (consistent with the expectation that most failures are small), but then find a way of increasing it as the extent of large failures is revealed. This point is discussed in more detail later.

In Fig 4, we plot the convergence delays for three topologies with different degree distributions (but same average degree). One of the topologies is the 70-30 degree distribution that we have been using. This topology has an average degree equal of 3.8. In the "50-50" topology, 50% of the nodes have degree in the range 1 to 3, whereas the rest have a degree of 5 or 6 in order to get an average degree of 3.8. The "85-15" topology has 85% nodes with degree 1 to 3, and the rest with degree 14, again with an average of 3.8. Fig 4 shows the variation in the convergence delay for 5% failure vs. MRAI value for the three different topologies. The minimum convergence delay for the "50-50", "70-30" and "85-15" topologies is achieved with MRAI roughly equal to 1.0, 1.25 and 2.25 seconds respectively. There is a distinct trend here, and it is related to the degree of the high degree nodes in each of the topologies. Nodes with high degree are likely to receive the largest number of messages and hence they are the most likely to get overloaded. Thus, for MRAI equal to 1.0 second, very few, if any, nodes in the "50-50" topology (high degree 5 and 6) seem to be overloaded. But with the same MRAI, a larger number of nodes in the "85-15" topology (high degree 14) can be expected to be overloaded, leading to a high convergence delay; and we have to increase the MRAI considerably to remove the overload. We also experimented with topologies which had multiple routers per AS and an inter-AS degree distribution that we derived from Internet AS connectivity data [18]. We restricted the maximum degree in the distribution to 40 (average degree ∼3.4) because we have only 120 ASes/nodes. We observed a V shaped curve for these topologies also. We talk about the results for that scenario later on in the paper.

After looking at the effect of the degree distribution on the convergence delay, we investigate the effect of the average degree on the same. In Fig 5, we plot the convergence delay for two topologies with the same type of degree distribution (50-50), but different average degree. One of the topologies is the same as the one that we saw in Fig 4, with average degree 3.8. In the other topology however, the high degree nodes have a degree of 13 or 14 resulting in an average degree of 7.6. We see that both the optimal MRAI and the convergence delay are greater for the topology with the higher degree. The larger optimal MRAI can be attributed to the greater degree of the high degree nodes, as we explained in the previous paragraph. In fact the optimal MRAI for the topology with average degree 7.6 (high degree 13 and 14) is ∼2 seconds, which is about the same as the optimal MRAI for the "85-15" topology (high degree 14). The increase in the convergence delay is because of the larger number of alternate paths that have to be considered.

## 4.2 Degree Dependent MRAI

We have seen in the previous section that the convergence delay seems to be closely linked to the behavior of the nodes with the highest degree. This leads to the idea of choosing a higher MRAI at higher degree nodes. This issue is explored in this section. We again used 120 node topologies with 70-30 degree distribution. In this network, 70% of nodes have a degree in the range 1 to 3, and we use MRAI=0.5 seconds at those nodes. We used MRAI=2.25 seconds at the remaining 30% of the nodes that have a de-

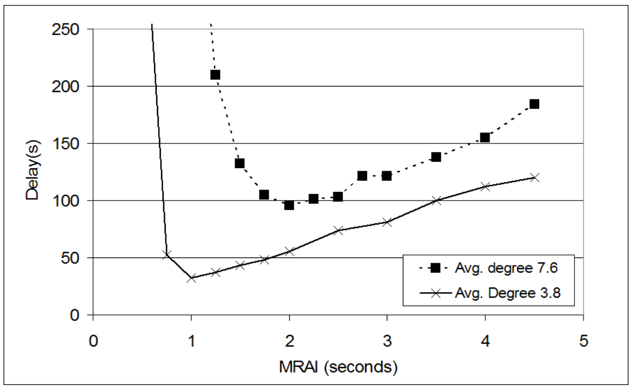

**Figure 5. Effect of average degree on convergence delay**

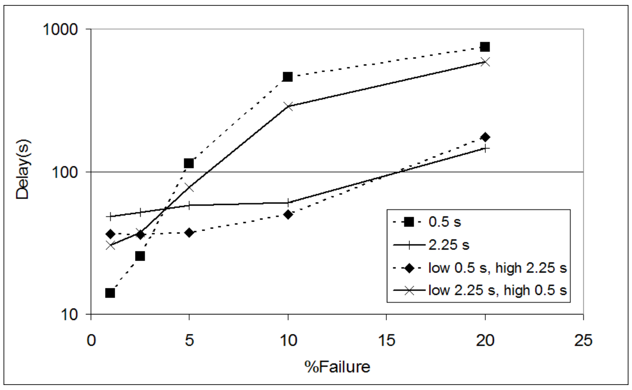

**Figure 6. Effect of degree dependent MRAI**

gree of 8. This case is marked as (low 0.5, high 2.25) in Fig 6. For comparison, we also examined the reversed situation, i.e., MRAI=2.25 seconds at low degree nodes and MRAI=0.5 secs at high degree nodes. This situation is marked as (low 2.25, high 0.5) in Fig 6. Two other cases, in which all nodes have the same MRAI (0.5 and 2.25 seconds) are also shown for comparison.

From the figure we can see that with the “low 0.5, high 2.25” scheme, the convergence delay can be decreased significantly. In fact, the delay is almost the same as that with a constant MRAI of 2.25 seconds for large failures but significantly lower for small failures. For the “low 2.25, high 0.5” case, the delay for larger failures is close to that with a constant MRAI of 0.5 seconds and is very high. So, the convergence behavior for large failures is largely dependent on the higher degree nodes in a network. And we can keep the convergence delay for large failures low by using a comparatively greater value of MRAI at high degree nodes. However we also see that for small failures, the convergence delay for “Degree dependent MRAI” is significantly higher that that for MRAI=0.5 seconds. This deficiency can be addressed by changing the MRAI dynamically, as discussed next.

## 4.3 Dynamic MRAI

As remarked earlier, if we have a scheme that can quickly estimate the size of the failure and set the MRAI accordingly, we can minimize the convergence delay for different types of failures. However such a scheme would probably be complex and might add significant overhead to the BGP convergence process. So we decided to focus on schemes that can dynamically change the MRAI at a node by monitoring the status of the node and the received updates. As mentioned earlier, a low MRAI value can lead to a large number of update messages, multiple overloaded nodes and large convergence delays. If a node is overloaded, then increasing the MRAI at that node will not only reduce the number of update messages it generates but will also cut down the number of invalid routes that it sends to its neighbors. So, this type of scheme can reduce the convergence delay by reducing the overall processing overhead in the network and by decreasing the number of invalid routes during the convergence process.

We implemented a scheme in which we varied the MRAI at a node between three different values. From the observed convergence delays for 120 node networks with 70-30 degree distributions we chose the values 0.5, 1.25 and 2.25 seconds. The selection was based on the observations that MRAI equal to 0.5 seconds resulted in the least convergence delay for small (1-2.5%) failures, while MRAI equal to 1.25 seconds was best for 5% failure and 2.25 seconds was good for failures in the 10 to 20% range. The MRAI is set to 0.5 seconds in the beginning because small failures are much more likely and in that scenario we will automatically incur the least delay. In our scheme, we monitor the queue length of update messages as an indicator of overload. We convert the queue length into *unfinished work* by multiplying it by the average processing delay. If the unfinished work is greater than a threshold (*upTh*), then we increase the MRAI if possible. If the unfinished work is less than another threshold (*downTh*), then we decrease the MRAI if possible. It must be noted that even if we decide to change the MRAI, we do not modify the values of the running timers; instead, the change takes effect only when the timers are restarted after an update has been sent. We did this to keep the implementation simple.

We show the effects of using this dynamic MRAI scheme in Fig 7. For this set of results we set the *downTh* to 0.05 seconds and the *upTh* to 0.65 seconds. From Fig 7 we can see that the dynamic MRAI scheme performs quite well. The convergence delay for small (1-2.5%) failures is actu-

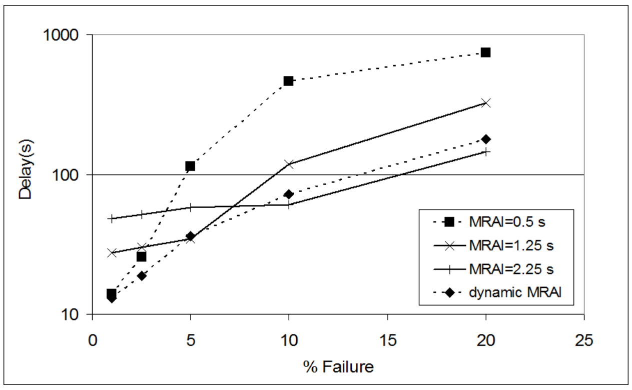

**Figure 7. Effect of dynamic MRAI**

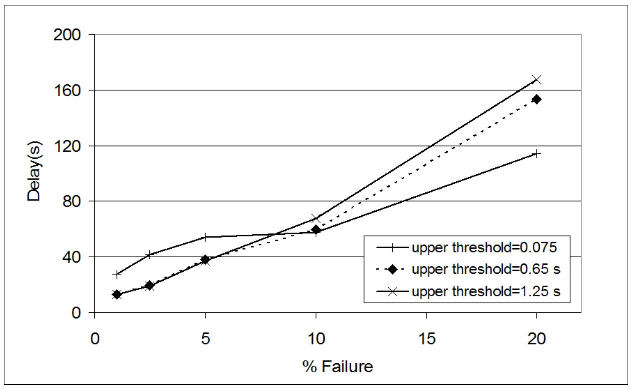

**Figure 8. Effect of *upTh* on convergence delay**

ally lower than that with MRAI=0.5 seconds. This tells us that some nodes get overloaded even for small failures. For 5% failure, the convergence delay for the dynamic scheme is about the same as that for MRAI=1.25 seconds. For larger failures, the delays for the dynamic scheme are higher than that for MRAI=2.25 seconds, but less than that for MRAI=1.25 seconds and much less than that for MRAI=0.5 seconds.

Thus we see that with this dynamic scheme we were able to get the close to the minimum convergence delays for a wide range of failures. We also found the number of messages generated by the dynamic MRAI scheme are a little above what we get if we use an MRAI of 2.25 seconds. We tested this scheme for topologies with 240 nodes as well. We obviously had to change the MRAI values but we kept the thresholds the same. The results were again very good and similar to what we have shown here, and are omitted to avoid repetition.

Now we look at the performance of the dynamic scheme if the thresholds are varied. We first set *downTh* to 0 and experimented with a number of *upTh* values. We have shown some of the results in Fig 8. If *upTh* is low, then the behavior is similar to having a constant high MRAI, because too many nodes increase their MRAI. Thus we see that with a low threshold, the convergence delay for small failures is comparatively high, but the delays for large failures are low. As we increase the threshold, less nodes increase their MRAIs. Hence the convergence delays for small failures go down but the delays for the larger failures go up. However we see that increasing the *upTh* to 1.25 seconds from 0.65 seconds doesn't have a big impact on the convergence delay. We are able to get good results for a range of *upTh* values.

Next we have a look at the effect of the *downTh* value on the delays. We show the results for those experiments in Fig 9. Here we have set *upTh* to 0.65 seconds. As we increase *downTh*, more nodes decrease their MRAI and the delays for larger failures are increased. We again observe similar results for a range of values.

We reran the experiments with the dynamic scheme only at the high degree nodes to see how the results are affected. We have seen in the previous section that the convergence delay for large failures is heavily dependent on the MRAI of the high degree nodes, and therefore it made sense only to change the MRAI of those nodes. However we found that the results were effectively the same as when we had the dynamic scheme at all the nodes. This was because the low degree nodes rarely (if ever) got overloaded and hence the MRAI at those nodes stayed at the minimum value. We also tested out some other schemes for dynamically varying the MRAI. In the first scheme, we used the processor utilization to detect overload and to change the MRAI. We got promising results with that scheme as well. In the second scheme, we monitored the number of update messages received at a node. This scheme was not very successful as it was difficult to set the up and down thresholds.

The dynamic MRAI scheme does have a couple of deficiencies. The first one has to do with the selection of the different parameters. For our experiments we first measured the convergence delays for different MRAI values, and then picked the MRAIs that resulted in the least delay for different failure magnitudes. This approach is viable for small or moderate sized networks, but for large networks like the Internet (more then 20,000 ASes) one will have to estimate the MRAI values. We are currently looking at the theoretical basis for the selection of the parameters. Secondly, with the dynamic scheme, the convergence delay for large failures is somewhat higher than that with MRAI=2.25 seconds. There are a couple of reasons for this. First, all nodes start off with an MRAI=0.5 seconds and it takes a while for the queues at the overloaded nodes to exceed the *upTh*. And the MRAI change takes effect only after the timer expires. We are exploring ways to reduce the response time and improve this aspect of the scheme.

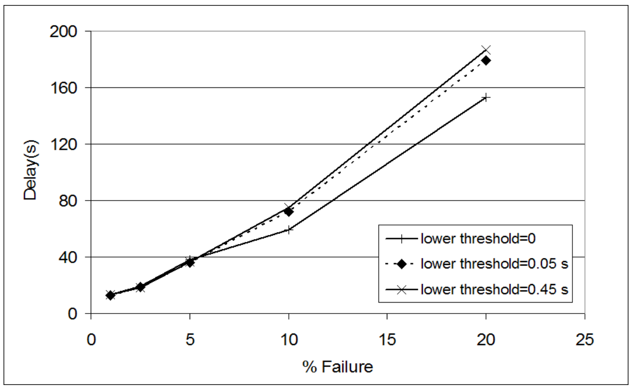

**Figure 9. Effect of *downTh* on convergence delay**

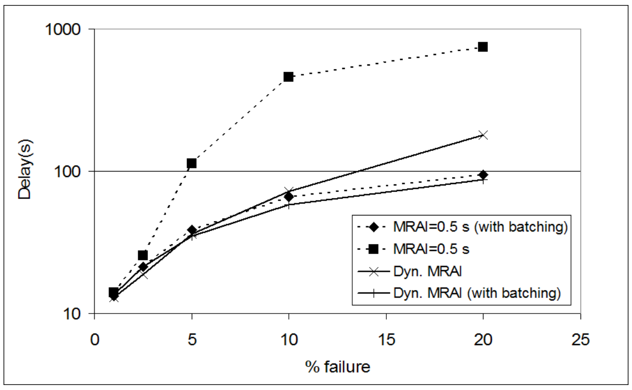

**Figure 10. Performance of Batching Scheme**

## 4.4 Batching of Update Processing

The default implementation of BGP processes all messages in the FIFO order and this may result in the generation of invalid updates and unnecessary processing of some messages. As an example, suppose that node *A* sends an update to neighbor *B* at time *t* and at that time there are four pending update messages in the queue. The first and third messages advertise a new route for node *X* while the second and fourth messages advertise a new route for node *Y*. Let's also assume that each update results in a new route for the corresponding destination, and that none of these routes pass through B. The updates will be processed in FIFO order by default. If the MRAI timer expires before the last two messages have been processed, then *A* will send two updates to *B* (one each for *X* and *Y*). Two more updates will be sent out after the final two updates have been processed. So, in all four updates were sent from *A* to *B*. However, if the timer expired after all the update messages had been processed, then only two update messages will be generated. We can reduce the number of update messages by reordering the messages in the queue without depending on the MRAI expiry. For example, if we move the third message (for destination *X*) to the second position, then both the update messages for *X* will be processed before the MRAI timer expires. So one update message (for *X*) will be sent after the timer expires, and another one (for *Y*) will be sent after the final two messages are processed. This leads to the idea of batched processing.

In the batched processing scheme, we effectively maintain a separate logical queue for each destination. When an update arrives we extract the destination, and queue it appropriately. Even with a large number of destinations, this can be implemented efficiently using hashing. The result of this queuing mechanism is that we can process all updates for a destination together and thereby address the problem identified above. Furthermore, we can delete multiple update messages from the same neighbor, as the older updates are now invalid. The price of destination based queuing should be small as compared to the benefits achieved.

We show the convergence delays for the batching scheme in Figs 10 and 11. We have also shown the convergence delays for the dynamic MRAI scheme for comparison. For the batching scheme, we set the MRAI to 0.5 seconds. We observe that the batching scheme is able to reduce the convergence delay for larger failures significantly while keeping the delays low for small failures. We also see that the delays are better than that for the dynamic MRAI scheme. If we combine the batching and dynamic MRAI schemes, then we are able to decrease the delays even further. The primary aim of the batching scheme is to reduce the number of updates generated by overloaded nodes. As shown in Fig 11, the number of messages is much less than that with MRAI=0.5 seconds and is in the same range as the number of messages for MRAI=2.25 seconds.

We also carried out experiments to observe the effect of the batching scheme with other MRAI values. We show the convergence delay for 5% failure, with different MRAIs, in the "70-30" topology in Fig 12. We observe that the convergence delay decreases significantly with batching if the MRAI is less than the optimal value; however batching does not have much of an impact otherwise. This is to be expected because the batching scheme is effective only when there are overloaded nodes in the network. If the queue of update messages is small, batching might not be possible at all. Even if some messages are rearranged, the difference in the time at which any particular message finishes execution will not be significant.

In order to verify our results, we tested the effect of batching and the dynamic MRAI scheme on the more realistic topologies that we mentioned in Section 4.1. For those topologies we found that the optimal MRAI for small

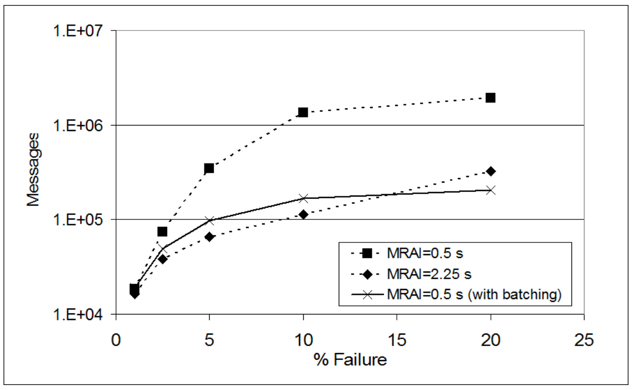

**Figure 11. Number of messages generated by the batching scheme**

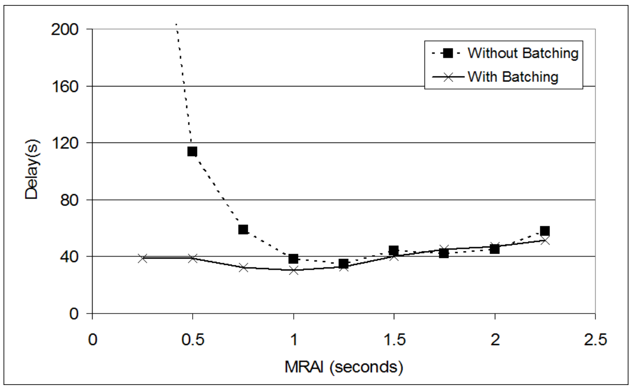

**Figure 12. Effect of batching with different MRAIs**

(1-2.5%) failures was 0.5 seconds and the optimal MRAI for large (10%) failures was 3.5 seconds. We show the effect of batching and the dynamic MRAI scheme on these topologies in Fig 13. We observe that the behavior is similar to what we saw earlier in Fig 10 . In conclusion, we have established that the batching scheme as well as the dynamic MRAI scheme can minimize the impact of large scale failures substantially by reducing the processing load at the BGP routers, without increasing the convergence delays for small failures.

On a separate note, another form of "batching" is carried out in BGP routers today. This is done to mitigate the speed mismatch between the rate at which BGP updates can be processed (fast) and the rate at which the new routes can be transferred to the line cards (slow). Typically one data buffer (TCP) is read from each peer connection and all the collected BGP updates are processed sequentially in a batch, after which the route changes are transmitted to the line cards. During periods of overload this scheme can provide some of the benefits as our scheme, if two updates for the same destination are present in the same batch. If the size of the failure is large however, the number of destinations for which updates are sent will be high, and the probability of having two updates for the same destination in a batch will progressively decrease. Thus our scheme should perform much better for large scale failures.

## 5 Discussion and Conclusions

In this paper, we have shown the effect of BGP's MRAI (Minimum Route Advertisement Interval) on the convergence delay for large scale failures. We found that the MRAI significantly affects the variation in the convergence delay as a function of the size of the failures. We discovered that the "optimal" MRAI, or the MRAI value at which we incur the least convergence delay, is dependent on the size of the failure and actually increases with the size. Thus, there is no single MRAI value which will provide the best convergence delay for different types of failures in a network. We also found that the "optimal" MRAI is dependent on the degree distribution of the network. We investigated the effect of having different MRAIs at different nodes and we saw that the convergence delay for larger ($\geq 5\%$) failures is heavily dependent on the MRAI of the higher degree nodes.

We presented a dynamic scheme to vary the MRAI at a node, which automatically tries to select the "optimal" MRAI for a failure, based on the current processing load at the router. We found that the dynamic scheme worked very well, and the convergence delay was always close to the minimum for failures of different sizes. The dynamic scheme reduced the convergence delays for large scale failures while keeping the delays low for smaller, more probable failures. The parameters for this scheme were the three different MRAI values and the two thresholds, all selected based on experimental results. In order to use this type of scheme in real networks, it is necessary to develop a suitable theory for choosing various parameters. This work is currently ongoing. We also examined a batching scheme, designed to reduce the generation of invalid route advertisements and to remove stale update messages, during periods of overload. We find that the batching scheme can substantially cut down the convergence delays (by a factor of 3 or more). The convergence delays were reduced even further if we combined the batching and the dynamic MRAI schemes. Another advantage of the batching scheme is that it does not use any configuration parameters.

Both of our proposed schemes are designed to improve the convergence delay in situations where the update processing load at BGP routers is high. If the processing delays are so small that the BGP routers do not get overloaded, then the convergence delays will be unchanged. The

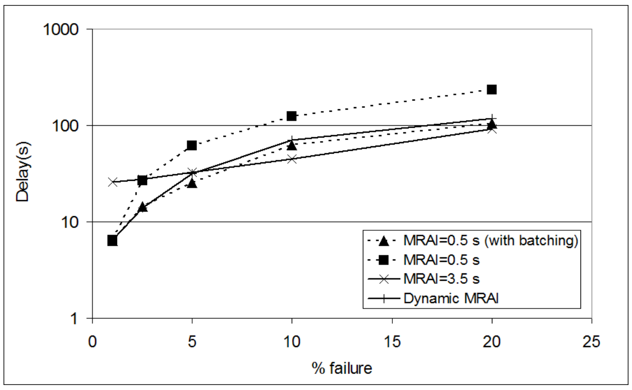

**Figure 13. Convergence delay of realistic topologies**

processing load is however also dependent on the number of update messages which in turn depends on the number of destinations affected by the failure. Despite the advances in router processor speeds, a large scale failure in the Internet, which contains nearly 200,000 destinations, will generate a huge number of updates that is likely to overload a large number of routers. Hence our schemes will be effective in such a scenario.

The future work on the subject includes more thorough experimentation and analytic modeling of the advantages of the various schemes discussed here. At the same time, we continue to look for ways of improving the proposed schemes. For example, a scheme that can accurately and quickly set the MRAI consistent with the extent of failure without significant overhead is highly desirable. Similarly, the batching scheme can be improved further to remove conflicting/superfluous updates. We are also looking at factors other than the processing overhead that are responsible for the V-shaped delay vs. MRAI curve, so that we can find new approaches to reduce the convergence delay for large scale failures. The ultimate aim of these efforts is to intelligently manage BGP update handling to a point where large failures no longer lead to significant down times, packet losses or packet delays.

## References

[1] Y. Rekhter and T. Li, "Border Gateway Protocol 4," RFC 1771, Mar. 1995.

[2] "The Border Gateway Protocol". [Online]. Available: http://www.cisco.com/univercd/cc/td/doc/cisintwk/-ito_doc/bgp.htm

[3] Bassam Halabi, *Internet Routing Architectures*. Cisco Press, 1997.

[4] C. Labovitz, G. R. Malan, and F. Jahanian, "Internet Routing Instability," *IEEE/ACM Transactions on Networking*, vol. 6, no. 5, pp. 515–528, Oct. 1998.

[5] Labovitz, C., Ahuja, et al., "Delayed internet routing convergence," in *Proc. ACM SIGCOMM 2000*, Stockholm, Sweden, Aug. 28–Sep. 1, 2000, pp. 175–187.

[6] C. Labovitz, A. Ahuja, et al., "The Impact of Internet Policy and Topology on Delayed Routing Convergence," in *Proc. IEEE INFOCOM 2001*, vol. 1, Anchorage, Alaska, Apr. 22—26, 2001, pp. 537–546.

[7] T.G. Griffin and B.J. Premore, "An experimental analysis of BGP convergence time," in *Proc. ICNP 2001*, Riverside, California, Nov. 11–14, 2001, pp. 53–61.

[8] Dan Pei, B. Zhang, et al., "An analysis of convergence delay in path vector routing protocols," *Computer Networks*, vol. 30, no. 3, Feb. 2006, pp. 398–421.

[9] D. Obradovic, "Real-time Model and Convergence Time of BGP," in *Proc. IEEE INFOCOM 2002*, vol. 2, New York, Jun. 23–27, 2002, pp. 893–901.

[10] G. Siganos and M. Faloutsos, "Analyzing BGP Policies: Methodology and Tool," in *Proc. IEEE INFOCOM 2004*, vol. 3, Hong Kong, Mar. 7–11, 2004, pp. 1640-1651.

[11] A. Sahoo, K. Kant, and P. Mohapatra, "Characterization of BGP recovery under Large-scale Failures," in *Proc. ICC 2006*, Istanbul, Turkey, Jun. 11–15, 2006.

[12] S. Deshpande and B. Sikdar,"On the Impact of Route Processing and MRAI Timers on BGP Convergence Times," in *Proc. GLOBECOM 2004*, Vol. 2, pp 1147- 1151.

[13] A. Medina, A. Lakhina, et al., "Brite: Universal topology generation from a user's perspective," in *Proc. MASCOTS 2001*, Cincinnati, Ohio, August 15–18, 2001, pp. 346-353.

[14] "SSFNet: Scalable Simulation Framework". [Online]. Available: http://www.ssfnet.org/

[15] B. Waxman, "Routing of Multipoint Connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, Dec. 1988.

[16] A.L. Barabasi and R. Albert, "Emergence of Scaling in Random Networks," *Science*, pp. 509–512, Oct. 1999.

[17] T. Bu and D. Towsley, "On Distinguishing between Internet Power Law Topology Generators," in *Proc. IEEE INFOCOM 2002*, vol. 2, New York, Jun. 23–27, 2002, pp. 638–647.

[18] B. Zhang, R. Liu, et al., "Measuring the internet's vital statistics: Collecting the internet AS-level topology ," *ACM SIGCOMM Computer Communication Review*, vol. 35, issue 1, pp. 53–61, Jan. 2005.

[19] A. Lakhina, J.W. Byers, et al., "On the Geographic Location of Internet Resources," *IEEE Journal on Selected Areas in Communications*, vol. 21 , no. 6, pp. 934–948, Aug. 2003.

[20] H. Tangmunarunkit, J. Doyle, et al, "Does Size Determine Degree in AS Topology?," *ACM SIGCOMM Computer Communication Review*, vol. 31, issue 5, pp. 7–10, Oct. 2001.