Improving BGP Convergence Delay for Large-Scale Failures

Amit Sahoo
Dept. of Computer Science
UC Davis
Davis, CA 95616, USA
Telephone: (530) 304–7375
Email: asahoo@ucdavis.edu
Contact Author

Dr. Krishna Kant
Intel Corporation
Hillsboro, OR 97124, USA
Telephone: (503) 712–3997
Email: krishna.kant@intel.com

Dr. Prasant Mohapatra
Dept. of Computer Science
UC Davis
Davis, CA 95616, USA
Telephone: (530) 754–8380
Email: pmohapatra@ucdavis.edu

Abstract

Border Gateway Protocol(BGP) is the standard routing protocol used in the Internet for routing packets between the Autonomous Systems(AS'es). 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 impact on the BGP recovery time. We propose a novel batching scheme to reduce the number of BGP updates, and thereby decrease route update flapping. We show that this batching scheme and 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.

Index Terms

BGP, convergence delay, large-scale failures, MRAI, batch processing

I. 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 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. When this primary path fails, BGP withdraws this path and 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 In case the backup route has also failed, it will be withdrawn only after a withdrawal is sent by the neighbor which advertised it.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 valid one. This can result in a considerable delay before the cycle of withdrawals/advertisements ends and all BGP nodes have a valid path to the destination.

Numerous studies [4], [5], [6], [7], [8], [9] 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 > 3 min in 30% of the cases, and could be as high as 15 minutes. Models [6], [7] 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.

The primary reasons why large scale/multiple failures in the Internet have not been studied are 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 thereby significantly increase the convergence time, as reported in our previous work on the subject [10]. We also observed that delay increased as the network grew in size, which means that the failure of 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. Therefore 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.

II. BGP Convergence Delay

Previous works [5], [7] 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 least MRAI seconds 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 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 that 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 [6] studied the effect of MRAI on the convergence time after a fault was initiated 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 equal 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.

We believe that the behavior observed by Griffin and Premore [6] can be explained in terms of processing overhead of BGP updates. Let’s assume that a node A sends an update to a neighbor B at time t. Then, if the MRAI is set to the optimal value, then there is a high probability that A has processed all the update messages in its queue by the time $t+Optimal\ MRAI$. If the MRAI is increased further, it means that the nodes have to wait longer before sending the update message and this increases the convergence delay. Thus, in this phase, the number of sent update messages remains roughly constant

and the delay increases linearly. If we decrease the MRAI below the optimal value, the rate at which updates are processed increases and more nodes to get overloaded. 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. So not only does the neighbor have to process an extra update message, the potential forwarding of the earlier update increases workload on the other downstream nodes as well. 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 of the failure, the average degree of the nodes, the degree distribution, the processing overhead, etc. [1], [3] In our previous study [10], we looked at numerous 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. We also observed that the convergence delay rose sharply in the beginning when the size of the failure was increased. Finally we saw that the convergence delay was decreased if the degree distribution was non-uniform(combination of high and low degree nodes).

A. Other Related Work

III. METHODOLOGY

We used a number of inter-AS network topologies for our studies. We used a modified version of BRITE [11] for topology generation and the SSFNet [12] simulator for the BGP simulations.

A. 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 [13], Albert-Barabasi [14], and GLP [15]. We modified BRITE to allow more flexible degree distributions. For 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 used two types of degree distributions to generate the topologies. The first was a constant distribution in which all the ASes had exactly the same number of inter-AS links. The second was 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. We used this distribution because it is closer to the heavy tailed degree distribution found in reality. 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 [16], about 70% of ASes were connected to less than 4 other ASes.

Although large scale failures could be scattered throughout the network, many failure scenarios (e.g., those caused by natural disasters, terrorism, etc.) are expected to be geographically concentrated. Unfortunately, BRITE does not have geographical location capability. We randomly placed all the routers on a 1000x1000 grid and then consider failures in contiguous areas of the grid (usually the center of the grid to avoid edge effects). In our previous work, we have examined 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).

B. 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 supported 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 a per-destination basis, as is commonly done in the Internet.

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 perhaps unlikely and not considered here. Since our networks had only one BGP router per AS, the failures are really AS failures. We plan to examine more complex scenarios in

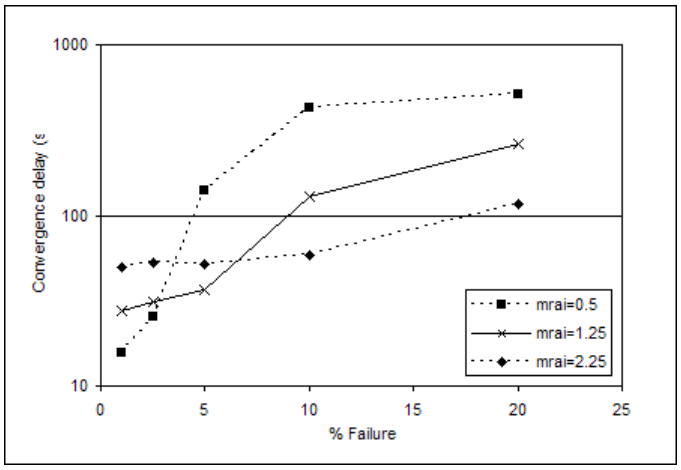

Fig. 1. Convergence delay for different sized failures

future work.

IV. 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/Nodes but we did run a smaller number of experiments with topologies of 60 and 240 nodes to verify the results.

A. Effect of MRAI

We had already found in our previous study [10] 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[III-A] distribution. 70% of the ASes/nodes had a degree in the range 1-3 while the 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 perhaps 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.

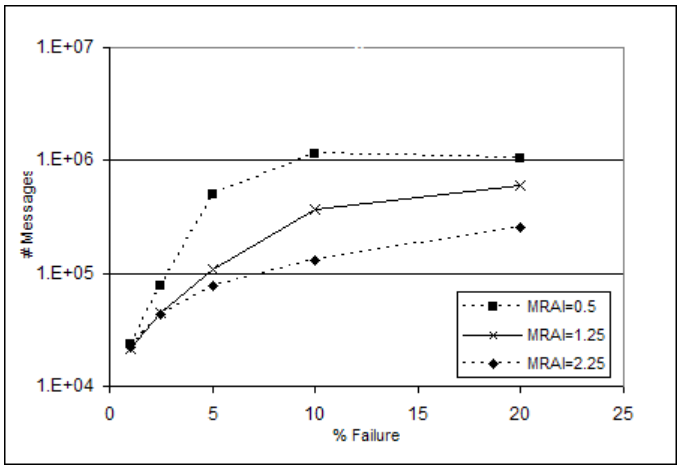

Fig. 2. No. of generated messages for different MRAI values

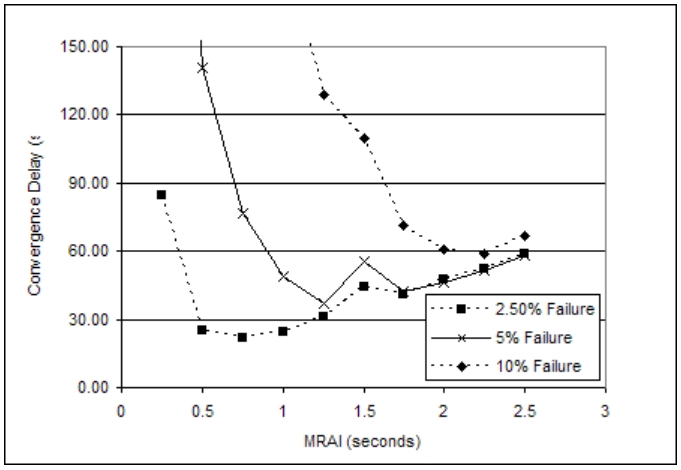

Fig. 3. Variation in convergence delay with MRAI

Fig 2 shows the number of generated messages for 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 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

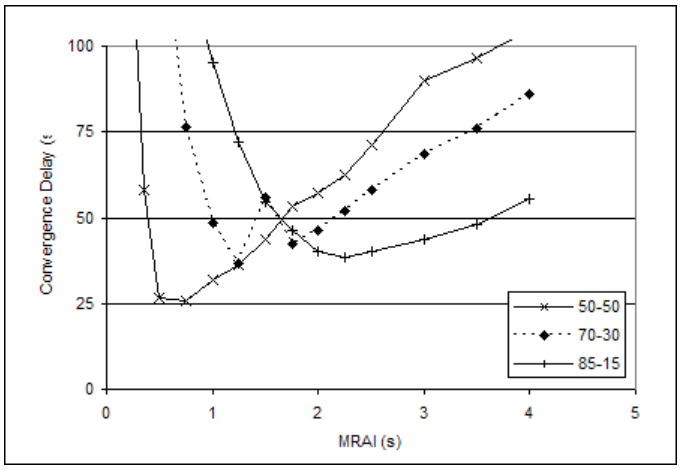

Fig. 4. Convergence Delay for Different Topologies

increases. This is similar to the results observed by Griffin and Premore [6]. We can see that increasing the MRAI beyond the optimal value (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 (1.25 seconds for for 5% failure) increases the number of messages and the convergence delay [Figs 1 and 2].

These observations explain why the optimal MRAI increases with the size of the failure. Basically, larger failures result in more update messages and hence higher processing load. For 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 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 four different topologies. The minimum convergence delay for the "50-50", "70-30" and "85-15" topologies

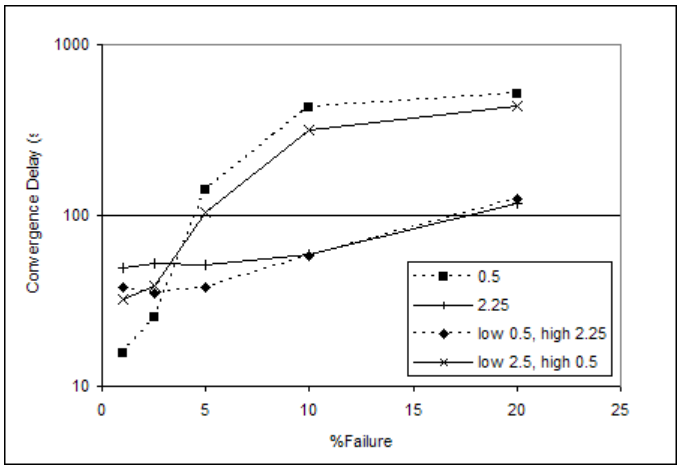

Fig. 5. Effect of variable MRAI

is achieved with MRAI roughly equal to 0.75, 1.25 and 2.25 seconds respectively. There is a distinct trend here, and is related to the maximum degree of a node in each of the topologies. Nodes close to the maximum degree are most likely to receive the largest number of messages and hence they are the most likely to get overloaded. Thus, for MRAI equal to 0.75 second, very few, if any, nodes in the ”50-50” topology (maximum degree=6) seem to be overloaded, leading to a small convergence delay. But with the same MRAI, a significant number of nodes in the ”85-15” topology (maximum degree=14) seem to be overloaded, causing the high convergence delay, and we have to increase the MRAI considerably to remove the overload.

B. Variable 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 for higher degree nodes. This issue is explored in this section. We again used 120 node topology 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 secs for them. The remaining 30% of the nodes have a degree of 8, and we use MRAI=2.25 secs. This case is marked as (low 0.5, high 2.25) in Fig 5. For comparison, we also examined the reversed situation, i.e., MRAI=2.25 secs for low degree nodes and MRAI=0.5 secs for high degree nodes. This situation is marked as (low 2.25, high 0.5) in Fig 5. Two other cases were also considered, namely where MRAI is fixed at 0.5 and 2.25 secs respectively.

From the figure we can see that for the ”low 0.5, high 2.25” case, 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. However for small failures, the convergence delays for ”variable MRAI” cases was in between the delays with MRAI equal to 0.5 and 2.25 seconds. This deficiency can be addressed by changing the MRAI dynamically, as discussed next.

C. 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 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 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 values of 1.25 and 2.25 seconds were best for 5 and 10% failures respectively. 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

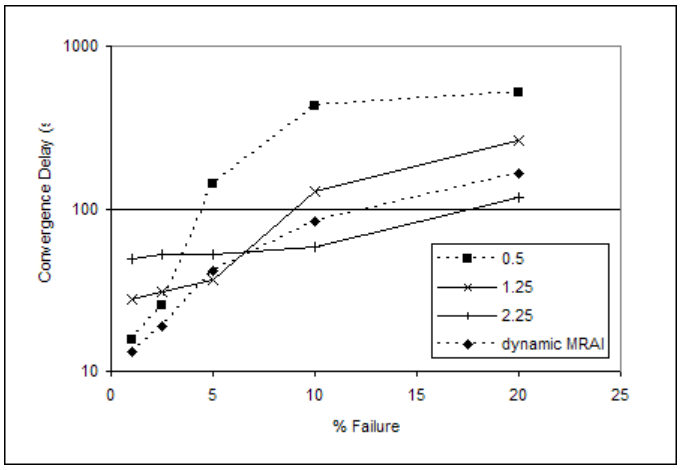

Fig. 6. Dynamic MRAI

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 show the effects of using this dynamic MRAI scheme in Fig 6. For this set of results we set the *downTh* to 0.05 seconds and the *upTh* to 0.65 seconds. From Fig 6 we can see that the dynamic MRAI scheme performs quite well. The convergence delay for small (1-2.5%) failures is actually lower than that with MRAI=0.5. 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.

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 7. 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

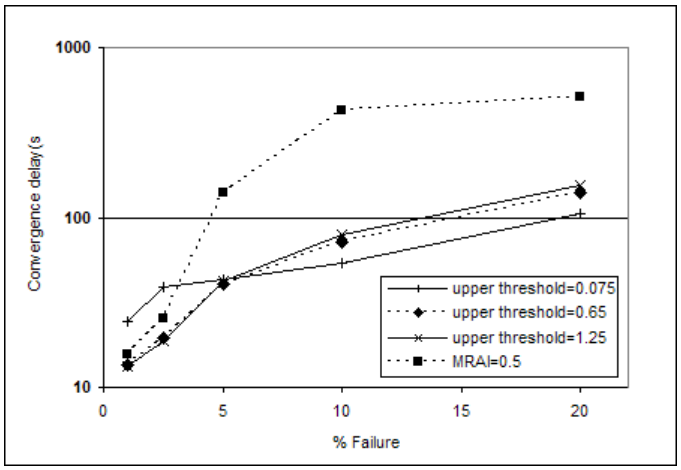

Fig. 7. Effect of upper threshold value

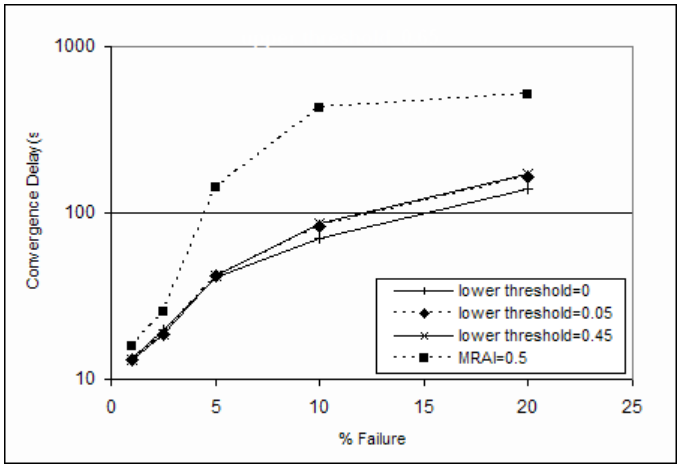

Fig. 8. Effect of lower threshold value

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 were able to get good results for a range of *upTh* values.

Next we have a look at the effect of the *downTh* value. We show the results for those experiments in Fig 8. 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. However we see that the sensitivity of the results

to the value of *downTh* is low and 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 only shortcoming of the dynamic MRAI scheme has to do with the selection of the different MRAI values. 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 might have to estimate the MRAI values. We should be able to do that by observing the trends for smaller networks, but one would obviously like to have a scheme with as few parameters as possible.

D. Batching of Update Processing

The default implementation of BGP processes all messages in the FIFO order and may result in unnecessary processing and sending out of conflicting messages. As an example, suppose that node *A* sends an update to neighbor *B* at time *t* and at that time there are four 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

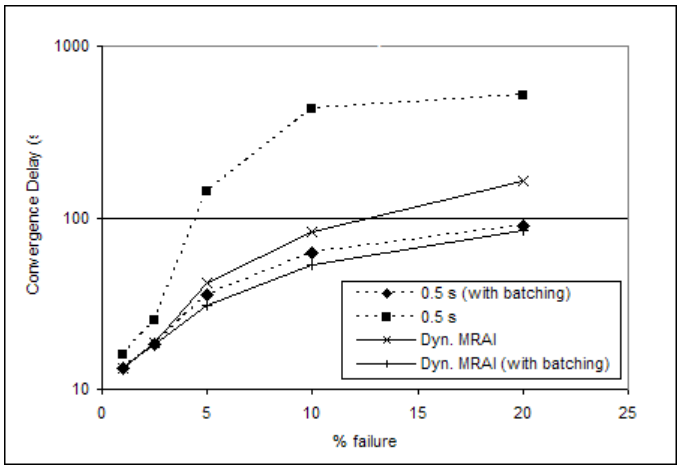

Fig. 9. Batching Scheme

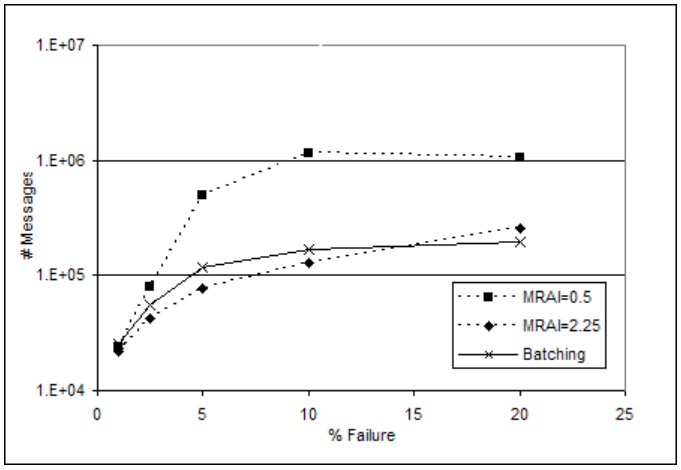

Fig. 10. Messages with batching scheme

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, 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.

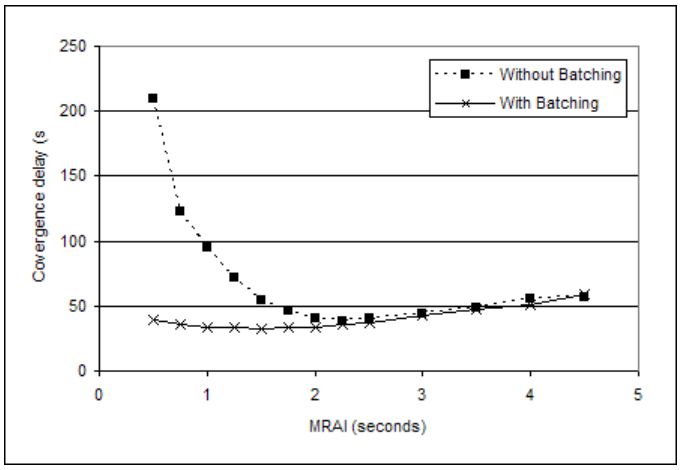

Fig. 11. Effect of batching for different MRAIs

Furthermore, we can cull messages with duplicate, invalid and conflicting updates. Such an intelligent processing reduces the processing overhead and minimizes route flapping. The price of destination based queuing is generally quite small as compared to the benefits achieved.

We show the convergence delays for the batching scheme in Figs 9 and 10. 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 can see that the batching scheme is able to reduce the convergence delay for larger failures significantly while keeping the delays low for small failures. We 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 10, 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 ”85-15” topology in Fig 11. We can see that the convergence delay decreases significantly with batching if the MRAI is less than the optimal value; however batching does not have any 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 conclusion, the batching scheme can minimize the impact of large scale failures substantially without increasing the convergence delays for small failures.

V. Discussion and Conclusion

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 and the size 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 also considered a dynamic scheme to vary the MRAI at a node, which automatically tries to select the "optimal" MRAI for a failure. 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 to catch update messages that can be ignored because they are conflicting, invalid or duplicate. We find that the batching can substantially cut down the convergence delays (by a factor of 3!). Another advantage of the batching scheme is that it uses only one parameter. The convergence delays were reduced even further if we combined the batching and the dynamic MRAI schemes.

The future work on the subject includes more through experimentation and analytic modeling of the advantages of various schemes discussed here. At the same time, we continue to look for ways of improving proposed schemes. For example, a scheme that can accurately and quickly set MRAI consistent with the extent of failure w/o significant overhead or message is highly desirable. Similarly, the batching

scheme can be improved further to reduce route flapping or superfluous updates. The ultimate aim of these techniques is to intelligently manage BGP update handling to a point where BGP convergence delay no longer leads to significant down times, packet losses or packet delays under a wide variety of failure scenarios.

REFERENCES

[1] Y. Rekhter and T. Li, “Border Gateway Protocol 4”, RFC 1771, SRI Network Information Center, July 1995.

[2] “The Border Gateway Protocol”, Cisco Systems, at 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, 1998.

[5] Labovitz, C., Ahuja, et al., “Delayed internet routing convergence,” In Proc. of SIGCOMM (Aug 2000), ACM, pp. 175-187.

[6] T.G. Griffin, B.J. Premore, “An experimental analysis of BGP convergence time,” 2001 International Conference on Network Protocols ICNP, pp. 53-61, 2001.

[7] Dan Pei, B. Zhang, et al., “An Analysis of Path-Vector Routing Protocol Convergence Algorithms,” Computer Networks, 2005.

[8] D. Obradovic, “Real-time Model and Convergence Time of BGP,” in Proc. Of lEEE Infocom, 2002.

[9] G. Siganos, “Analyzing BGP Policies: Methodology and Tool,” IEEE INFOCOM 2002. http://citeseer.ist.psu.edu/652452.html

[10] Amit Sahoo, Krishna Kant, and Prasant Mohapatra, ”Characterization of BGP recovery under Large-scale Failures,” submitted to ICC 2006.

[11] A. Medina, A. Lakhina, et al., “Brite: Universal topology generation from a user’s perspective,” In Proc. of MASCOTS, October 2001.

[12] “SSFNet: Scalable Simulation Framework” Network Models. http://www.ssfnet.org/.

[13] B. Waxman, “Routing of Multipoint Connections,” IEEE J. Select. Areas Commun. SAC-6(9): 1617-1622, Dec 1988.

[14] A.L. Barabasi and R. Albert, “Emergence of Scaling in Random Networks,” Science, pp 509-512,Oct 1999.

[15] T. Bu and D. Towsley, “On Distinguishing between Internet Power Law Topology Generators,” in Proc. Infocom 2002, June 23–27, New York.

[16] B. Zhang, R. Liu, et al., “Collecting the Internet AS-level Topology,” ACM SIGCOMM Computer Communication Review (CCR), special issue on Internet Vital Statistics, January, 2005.

[17] X. Zhao, D. Pei, D. Massey, and L. Zhang, ”A Study on Routing Behavior of Latin America Networks,” IFIP/ACM SIGCOMM Latin America Networking Conference, La Paz, Bolivia, Oct 2003.