Quantifying Path Exploration in the Internet

Ricardo Oliveira* Beichuan Zhang† Dan Pei‡ Lixia Zhang*

*{rveloso,lixia}@cs.ucla.edu †bzhang@cs.arizona.edu ‡peidan@research.att.com
University of California, Los Angeles University of Arizona AT&T Labs – Research

Abstract— **Previous measurement studies have shown the existence of path exploration and slow convergence in the global Internet routing system, and a number of protocol enhancements have been proposed to remedy the problem. However, existing measurements were conducted only over a small number of testing prefixes. There has been no systematic study to quantify the pervasiveness of BGP slow convergence in the operational Internet, nor any known effort to deploy any of the proposed solutions.**

In this paper we present our measurement results that identify BGP slow convergence events across the entire global routing table. Our data shows that the severity of path exploration and slow convergence varies depending on where prefixes are originated and where the observations are made in the Internet routing hierarchy. In general, routers in tier-1 ISPs observe less path exploration, hence they experience shorter convergence delays than routers in edge ASes; prefixes originated from tier-1 ISPs also experience less path exploration than those originated from edge ASes. Furthermore, our data shows that the convergence time of route fail-over events is similar to that of new route announcements, and is significantly shorter than that of route failures. This observation is in contrary to the widely held view from previous experiments but confirms our earlier analytical results. Our effort also led to the development of a path preference inference method based on the path usage time, which can be used by future studies of BGP dynamics.

I. Introduction

The Border Gateway Protocol (BGP) is the routing protocol used in the global Internet. A number of previous analytical and measurement studies ([1], [2], [3]) have shown the existence of BGP path exploration and slow convergence in the operational Internet routing system, which can potentially lead to severe performance problems in data delivery. Path exploration suggests that, in response to path failures or routing policy changes, some BGP routers may try a number of transient paths before selecting a new best path or declaring unreachability to a destination. Consequently, a long time period may elapse before the whole network eventually converges to the final decision, resulting in slow routing convergence. An example of path exploration is depicted in Figure 1, where node C's original path to node E (path 1) fails due to the failure of link D-E. C reacts to the failure by attempting two alternative paths (paths 2 and 3) before it finally gives up. The experiments in [1], [2], [3] show that some BGP routers can spend up to several minutes exploring a large number of alternate paths before declaring a destination unreachable.

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No N66001-04-1-8926 and by National Science Foundation (NSF) under Contract No ANI-0221453. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the DARPA or NSF.

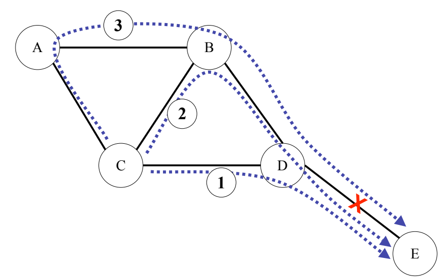


Fig. 1. Path exploration triggered by a fail-down event.

The analytical models used in the previous studies tend to represent worst case scenarios of path exploration [1], [2], and the measurement studies have all been based on controlled experiments with a small number of beacon prefixes. In the Internet operational community there exist various different views regarding whether BGP path exploration and slow convergence represent a significant threat to the network performance, or whether the severity of the problem, as shown in simulations and controlled experiments, would be rather rare in practice. A systematic study is needed to quantify the pervasiveness and significance of BGP slow convergence in the operational routing system, which is the goal of this paper.

In this paper we provide measurement results from the BGP log data collected by RouteViews [4] and RIPE [5]. For all the destination prefixes announced in the Internet, we cluster their BGP updates into routing events and classify the events into different convergence classes. We then characterize path exploration and convergence time of each class of events. The results reported in this paper are obtained from BGP logs of January and February 2006, which are representative of data we have examined during other time periods. The main contributions of this paper are summarized as follows.

- We provide the first quantitative assessment on path explorations for the entire Internet destination prefixes. Our results confirmed the wide existence of path exploration and slow convergence in the Internet, but also revealed that the extent of the problem depends on where a prefix is originated and where the observation is made in the Internet routing hierarchy. When observed from a top tier Internet service provider (ISP), there is relatively little path exploration, and this is especially true when the prefixes being observed are also originated from some other top tier ISPs. On the other hand, an observer in an edge network is likely to notice a much higher degree of path exploration and slow convergence, especially when the prefixes being observed are originated from other edge networks. In other words, the existing different opinions on the extent of path exploration and slow convergence may be a reflection of where one takes measurement and which prefixes are being examined.

- We provide the first measurement and analysis on the convergence times of *route change* events in the entire operational Internet. Our results show that route fail-over events, where the paths move from shorter or more preferred ones to longer or less preferred ones, has much shorter convergence time than route failure events, where the destinations become unreachable. Moreover, we find that, on average, the durations of various route convergence events take the following order: among all routing events, those moving from longer or less preferred to shorter or more preferred paths, symbolically denoted as T_{short} events, have the shortest convergence delay, which are closely followed by new prefix announcements (denoted as T_{up} event), which in turn have similar convergence delay as the routing events of moving from shorter to longer paths (denoted as T_{long}). Finally, route failure events, denoted as T_{down}, have a substantially longer delay than all the above events. In short, we have $T_{short} < T_{up} \approx T_{long} \ll T_{down}$ regarding their convergence delays. Note that T_{long} is significantly shorter than T_{down}, which is a noticeable departure from widely accepted views based on the previous "worst-case" experiments [1], but is in accordance to our previous theoretical analysis results presented in [6].
- A major challenge in our data analysis is how to differentiate T_{long} and T_{short} events, which requires knowing routers' path preferences. We have developed a new path ranking algorithm to infer relative preference of each path among all the alternative paths to the same destination prefix. We believe that our path ranking algorithm can be of useful in many other BGP data analysis studies.

The rest of the paper is organized as follows. Section II describes our general methodology and data set where we develop a path ranking algorithm to classify events into different types. We analyze the extent of path exploration and slow convergence for each type of events in Sections III and IV. Section V discuss related work and Section VI concludes the paper.

II. Methodology and Data Set

Previous measurement results on BGP slow convergence were obtained through controlled experiments. In these experiments, a small number of "beacon" prefixes are periodically announced and withdrawn by their origin ASes at fixed time intervals [7], [8], and the resulting routing updates are collected at remote monitoring routers and analyzed. In addition to generate announcements and withdrawals (T_{up} and T_{down} events), one can also use a beacon prefix to generate T_{long} events by doing AS prepending [1]. For a given beacon prefix, because one knows exactly what, when, and where is the root cause of each routing update, one can easily measure the routing convergence time by calculating the difference between when the root cause is triggered and when the last update due to the same root cause is observed. Although routing updates for beacon prefixes may also be generated by unexpected path changes in the network, those updates can be clearly identified through the use of *anchor prefixes* as explained later in this section. Unfortunately one cannot assess the overall Internet routing performance from observing the small number of existing beacon prefixes.

Our observation of routing dynamics is based on a set of routers termed *monitors*, that propagate their routing table updates to *collector* boxes, which store them in disk (e.g. RouteViews[4]). To obtain a comprehensive understanding of BGP path explorations in the operational Internet, we first cluster routing updates from the same monitor and for the same prefix into events, sort all the routing events into several classes, and then measure the duration and number of paths explored for each class of events. Our task is significantly more difficult than measuring the convergence delay of beacon prefixes for the following reasons. First, there is no easy way to tell whether a sequence of routing updates is due to the same, or different root causes in order to properly group them into events. Second, upon receiving an update for a prefix, one cannot tell what is the root cause of the update, as is the case with beacon prefixes. Furthermore, when the path to a given destination prefix changes, it is difficult to determine whether the new path is a more, or less, preferred path compared to the previous one, i.e. whether the prefix experiences a T_{short} or a T_{long} event in our event classification.

To address the above problems, we take advantage of beacon updates to develop and calibrate effective heuristics and then apply them to all the prefixes. In the rest of this section, we first describe our data set, then discuss how we use beacon updates to validate a timer-based mechanism for grouping routing updates into events, and how we use beacon updates to develop a usage-based path ranking method which is then used in our routing event classifications.

A. Data Set and Preprocessing

To develop and calibrate our update grouping and path ranking heuristics, we used eight BGP beacons, one from PSG [7] (*psg01*), the other seven from RIPE [8] (*rrc01*, *rrc03*, *rrc05*, *rrc07*,*rrc10*, *rrc11* and *rrc12*). All the eight beacon prefixes are announced and withdrawn alternately every 2 hours. We preprocessed the beacon updates following the methods developed in [3]. First, we removed from the update stream all the duplicate updates, as well as the updates that differ only in COMMUNITY or MED attribute values, because these updates are usually caused by internal dynamics inside the last-hop AS. Second, we used the *anchor prefix* of each beacon to detect routing changes other than those generated by the beacon origins. An anchor prefix is a separate prefix announced by a beacon prefix's origin AS, and is never withdrawn after its announcement. Thus it serves as a calibration point to identify routing events that are not originated by the beacon injection/removal mechanism. Because the anchor prefix shares the same origin AS, and hopefully the same routing path, with the beacon prefix, any routing changes that are not associated with the beacon mechanism will trigger routing updates for both the anchor and the beacon prefixes. To remove all beacon updates triggered by such unexpected routing events, for each anchor prefix update at time t, we ignore all beacon updates during the time window $[t-W, t+W]$. We set W's value to 5 minutes, as the results reported in [3] show that the number of beacon updates remains more or less constant for $W > 5$ minutes. After the above two steps of preprocessing, beacon updates are mainly comprised of those triggered by the scheduled beacon activity at the origin ASes.

To assess the degree of path exploration for all the prefixes in the global routing table, we used the public BGP data collected from 50 full-table monitoring points by RIPE [5] and RouteViews [4] collectors during the months of January and February 2006. We used the data from January to evaluate the different path comparison metrics and we later analyzed the events in both months. We removed from the data all the updates that were caused by BGP session resets between the collectors and the monitors, using the minimum collection time method described in [9]. Those updates correspond to BGP routing table transfers between the collectors and the monitors, and therefore should not be accounted in our study of the convergence process.

The 50 monitors were chosen based on the fact that each of them provided full routing tables and continuous routing data during our measurement period. One month was chosen as our measurement period based on the assumption that ISPs are unlikely to make many changes of their interconnectivity within one month period, so that we can assume the AS level topology did not change much over our measurement time period, an assumption that is used in our AS path comparison later in the paper.

B. Clustering Updates into Events

Some of the previous BGP data analysis studies [10], [11], [12] developed a timer-based approach to cluster routing updates into events. Based on the observation that BGP updates come in bursts, two adjacent updates for the same prefix are assumed to be due to the same routing event if they are separated by a time interval less than a threshold T. A critical step in taking this approach is to find an appropriate value for T. A value that is too high can incorrectly group multiple events into one. On the other hand, a value that is too low may divide a single event into multiple ones. Since the root causes of beacon routing events are known, and the beacon update streams contain little noise after the preprocessing, we use beacon prefixes to find an appropriate value for T.

Figure 2 shows the distribution of update inter-arrival times of the eight beacon prefixes as observed from the 50 monitors. All the curves start flattening out either before or around 4 minutes (the vertical line in the figure). If we use 4 minutes as the threshold value to separate updates into different events, i.e. $T = 4$ minutes, in the worst case (*rrc01* beacon) we incorrectly group about 8% of messages of the same event into different events; this corresponds to the inter-arrival time difference between the cutting point of the *rrc01* curve at 4 minutes and the horizontal tail of the curve. The tail drop of all the curves at 7200 seconds corresponds to the 2-hour interval between the scheduled beacon prefix activities [1].

Although the data for the beacon updates suggests that a threshold of $T = 4$ minutes may work well for grouping updates into events, no single value of T would be a perfect fit for all the prefixes and all the monitors. Thus we need to assess how sensitive our results may be with the choice of $T = 4$ minutes. Figure 3 compares the result of using $T = 4$ minutes with that of $T = 2$ minutes and $T = 8$ minutes for clustering the updates of all the prefixes collected from all the 50 monitors during our one-month measurement period. Let $E(m, p, 4)$ be the number of events identified by monitor m for prefix p using $T = 4$ minutes; $E(m, p, 2)$ and $E(m, p, 8)$ are similarly defined but with $T = 2$ minutes and $T = 8$ minutes respectively. Figure 3 shows the distribution of $|E(m, p, 8) - E(m, p, 4)|$ and $|E(m, p, 2) - E(m, p, 4)|$, which reflects the impact of using a higher or lower timeout value, respectively. As one can see from the figure, in about 50% of the cases the three different T values result in the same number of events, and in more than 80% of the cases the results from using the different T values differ by at most 2 events. Based on the data we can conclude that the result of event clustering is insensitive to the choice of $T = 4$ minutes. This observation is also consistent with previous work. For example [12] experimented with various timeout threshold values between 2 minutes and 16 minutes, and found no significant difference in the clustering results. In the rest of the paper, we use $T = 4$ minutes.

C. Classifying Routing Events

After the routing updates are grouped into events, we classify the events into different types based on the effect that each event has on the routing path. Let us consider two consecutive events n and $n + 1$ for the same prefix observed by the same monitor. We define the path in the last update of event n as the *ending path* of event n, which is also the *starting path* for event $n + 1$. Let p_{start} and p_{end} denote an event's starting and ending paths, respectively, and ε denote the path in a withdrawal message (representing an empty path). If the last update in an event is a withdrawal, we have $p_{end} = \varepsilon$. Based on the relation between p_{start} and p_{end} of each event, we classify all the routing events into one of the following

[1]The *psg01* curve reaches a plateau earlier than the other curves, indicating that it suffers less from slow routing convergence. However one may note its absence of update inter-arrivals between 100 seconds and 3600 seconds, followed by a high number of inter-arrivals around 3600 seconds. As hinted in [3], this behavior could be explained by BGP's route flat damping, and one hour is the default maximum suppression time applied to an unstable prefix when its announcement goes through a router which enforces BGP damping.

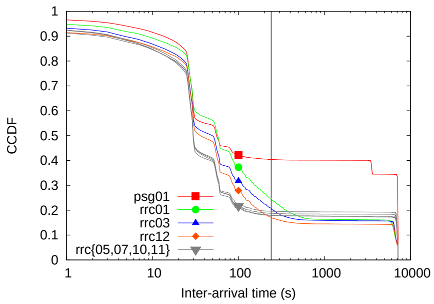


Fig. 2. CCDF of inter-arrival times of BGP updates for the 8 beacon prefixes as observed from the 50 monitors.

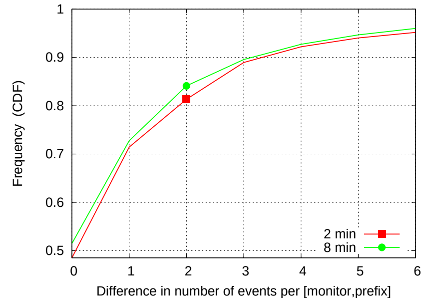


Fig. 3. Difference in number of events per [monitor,prefix] for T=2 and 8 minutes, relatively to T=4 minutes, during one month period.

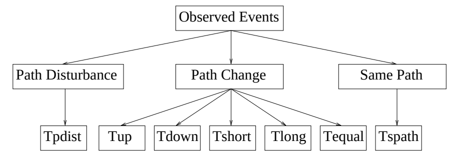


Fig. 4. Event taxonomy.

categories as shown in Figure 4 [2].

1) *Same Path* (T_{spath}): A routing event is classified as a T_{spath} if its $p_{start} = p_{end}$, and every update in the event reports the same AS path as p_{start}, although they may differ in some other BGP attribute such as MED or COMMUNITY value. T_{spath} events typically reflect the routing dynamics inside the monitor's AS.
2) *Path Disturbance* (T_{pdist}): A routing event is classified as T_{pdist} if its $p_{start} = p_{end}$, and at least one update in the event carries a different AS path. In other words, the AS path is the same before and after the event, with some transient change(s) during the event. T_{pdist} events are likely resulted from multiple root causes, such as a transient failure closely followed by a quick recovery, hence the name of the event type. When multiple root causes occur closely in time, the updates they produce also follow each other very closely, and no timeout value would be able to accurately separate them out by the root causes. In our study we identify these T_{pdist} events but do not include them in the convergence analysis.
3) *Path Change*: A routing event is classified as a path change if its $p_{start} \neq p_{end}$. In other words, the paths before and after the event are different. Path change events are further classified into five categories, based on whether the destination becomes available or unavailable, or changed to a more preferred or less preferred path, at the end of the event. Let $pref(p)$ represent a router's preference of path p, with a higher value representing a higher preference.
   - T_{up}: A routing event is classified as a T_{up} if its $p_{start} = \varepsilon$. A previously unreachable destination becomes reachable through path p_{end} by the end of the event.
   - T_{down}: A routing event is classified as T_{down} if its $p_{end} = \varepsilon$. That is, a previously reachable destination becomes unreachable by the end of the event.
   - T_{short}: A routing event is classified as T_{short} if its $p_{start} \neq \varepsilon$, $p_{end} \neq \varepsilon$ and $pref(p_{end}) > pref(p_{start})$, indicating a reachable destination has changed the path to a more preferred one by the end of the event.
   - T_{long}: A routing event is classified as a T_{long} event if its $p_{start} \neq \varepsilon$, $p_{end} \neq \varepsilon$ and $pref(p_{end}) < pref(p_{start})$, indicating a reachable destination has changed the path to a less preferred one by the end of the event.
   - T_{equal}: A routing event is classified as T_{equal} if its $p_{start} \neq \varepsilon$, $p_{end} \neq \varepsilon$ and $pref(p_{end}) = pref(p_{start})$. That is, a reachable destination has changed the path by the end of the event, but the starting and ending paths have the same preference.

A major challenge in event classification is how to differentiate between T_{long} and T_{short} events, a task that requires judging the relative preference between two given paths. Individual routers use locally configured routing policies to choose the most preferred path among available ones. Because we do not have precise knowledge of the routing policies, we must derive effective heuristics to infer a routers' path preference. It is possible that our heuristics label two paths with equal preference, in which case the event will be classified as T_{equal}. However, a good path ranking heuristic should minimize such ambiguity.

D. *Comparing AS Paths*

If a routing event has non-empty p_{start} and p_{end}, then the relative preference between p_{start} and p_{end} determines whether the event is a T_{long} or T_{short}. In the controlled experiments using beacon prefixes, one can create such events by manipulating AS paths. For example in [1], AS paths with length up to 30 AS hops were used to simulate T_{long} events.

However in general there has been no good way to infer routers' preferences among multiple available AS paths to the same destination. Given a set of available paths, a BGP router chooses the most preferred one through a decision process. During this process, the router usually considers several factors in the following order: local preference (which reflects the local routing policy configuration), AS path length, the MED attribute value, IGP cost, and tie-breaking rules. Some of the previous efforts in estimating path preference tried to emulate a BGP router's decision process to various degrees. For example, [1], [2], [12] used path length only. Because BGP is not a

[2]To establish a valid starting state, we initialize p_{start} for each (monitor,prefix) pair with the path extracted from the routing table of the corresponding monitor.

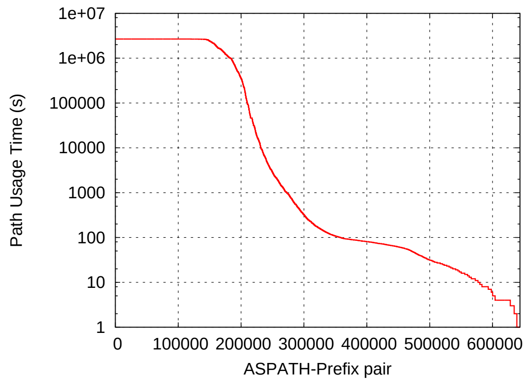


Fig. 5. Usage time per ASPATH-Prefix for router 12.0.1.63, Jan 2006.

shortest-path routing protocol, however, it is known that the most preferred BGP paths are not always the shortest paths. In addition, there often exist multiple shortest paths with equal AS hop lengths. There are also a number of other efforts in inferring AS relationship and routing policies. However as we will show later in this section, none of the existing approaches significantly improves the inference accuracy.

To infer path preference with a high accuracy for our event classification, we took a different approach from all the previous studies. Instead of emulating the router's decision process, we propose to look at the end result of the router's decision: the *usage time* of each path. The usage time is defined as the cumulative duration of time that a path remains in the router's routing table for each destination (or prefix). Assuming that the Internet routing is relatively stable most of the time and failures are recovered promptly, then most preferred paths should be used most and thus remain in the routing table for the longest time. Given our study period is only one month, during this time period it is unlikely that significant changes happened to routing policies and/or ISP peering connections in the Internet. Thus we conjecture that relative preferences of routing paths remained stable for most, if not all, the destinations during our study period. Figure 5 shows the path usage time distribution for the monitor with IP address 12.0.1.63 (AT&T). The total number of distinct ASPATH-prefix pairs that appeared in this router's routing table during the month is slightly less then 650,000 (corresponding to about 190,000 prefixes). About 23% of the ASPATH-prefix pairs (the 150,000 on the left side of the curve) stayed in the table for the entire measurement period, and about 500,000 ASPATH-prefix pairs appeared in the routing table for only a fraction of the period, ranging from a few days to some small number of seconds.

We compare this new *Usage Time* based approach with three other existing methods for inferring path preference: *Length*, *Policy*, and *Policy+Length*. *Usage Time* uses the usage time to rank paths. *Length* infers path preference according the AS path length. *Policy* is derives path preference based on inferred inter-AS relationships. We used the algorithm developed in [13] to classify the relationships between ASes into customer, provider, peer, and sibling. A path that goes through a *customer* is preferred over a path that goes through a *peer*, which is preferred over a path that goes through a *provider* [3]. *Policy+Length* infers path preference by using the policies first, and then using AS length for those paths that have the same AS relationship.

One challenge in conducting this comparison is how to verify the path ranking results without knowing the router's routing policy configurations. We tackle this problem by leveraging our understanding about T_{down} and T_{up} events. During T_{down} events, routers explore multiple paths in the order of decreasing preference; during T_{up} events, routers explore paths in the order of increasing preference. Since we can identify T_{down} and T_{up} events fairly accurately, we can use the information learned from these events to verify the results from different path ranking methods.

In an ideal scenario where paths explored during a T_{down} (or T_{up}) event follow a monotonically decreasing (or increasing) preference order, we can take samples of every consecutive pair of routing updates and rank order the paths they carried. However due to the difference in update timing and propagation delays along different paths, the monotonicity does not hold true all the time. For example, we observed path withdrawals appearing in the middle of update sequences *during* T_{down} events. Therefore, instead of comparing the AS paths carried in adjacent updates during a routing event, we compare the paths occurred during an event with the stable path used either before or after the event. Figure 6 shows our procedure in detail. All the updates in the figure are for the same prefix P. Before the T_{up} event occurs, the router does not have any route to reach P. The first four updates are clustered into a T_{up} event that stabilizes with path p_4. After p_4 is in use for some period of time, the prefix P becomes unreachable. During the T_{down} event, paths p_5 and p_6 are tried before the final withdrawal update. From this example, we can extract the following pairs of path preference: $pref(p_1) < pref(p_4)$, $pref(p_2) < pref(p_4)$, $pref(p_3) < pref(p_4)$, $pref(p_5) < pref(p_4)$, and $pref(p_6) < pref(p_4)$.

After extracting path preference pairs from T_{down} and T_{up} events, we apply the four path ranking methods in comparison to the same set of routing updates and see whether they produce the same path ranking results as we derived from T_{down} and T_{up} events. We keep three counters $C_{correct}$, C_{equal} and C_{wrong} for each method. For instance, in the example of Figure 6, if a method results in p_1 and p_2 being *worse* than p_4, and p_3 having the same preference of p_4 (*equal*), then for the T_{up} event we have $C_{correct} = 2$, $C_{equal} = 1$ and $C_{wrong} = 0$. Likewise, for the T_{down} event, if a method results in p_5 being *better* than p_4 and p_6 being *equal* to p_4, then we have $C_{correct} = 0$, $C_{equal} = 1$ and $C_{wrong} = 1$. To quantify the accuracy of different inference methods, we define $P_{correct} = \frac{C_{correct}}{C_{correct}+C_{equal}+C_{wrong}}$. We use $P_{correct}$ as a measure of accuracy in our comparison.

To compare the four different path ranking methods, we first applied them to our beacon data set which contains updates generated by T_{up} and T_{down} events, and computed the values of $C_{correct}$, C_{equal} and C_{wrong} for each of the

[3] We ignore those cases in which we could not establish the policy relation between two ASes. Such cases happened in less than 1% of the total paths.

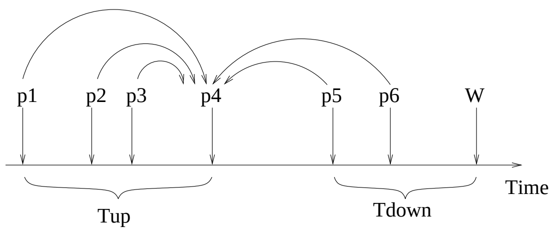


Fig. 6. Validation of path preference metric.

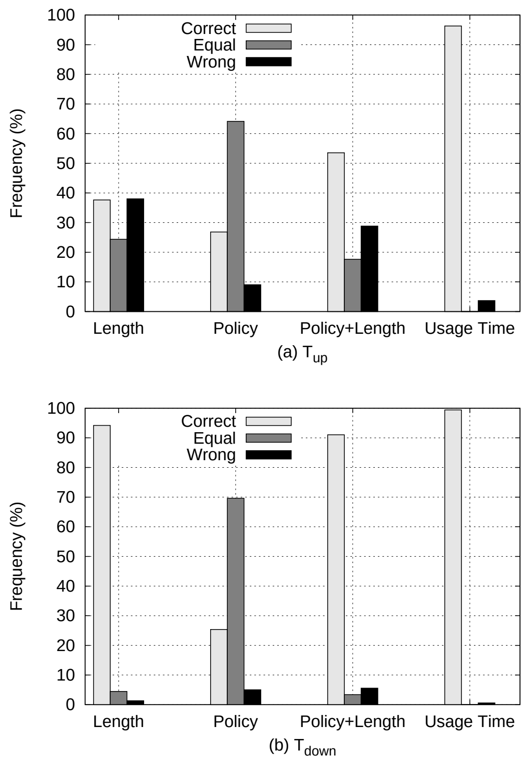


Fig. 7. Comparison between $C_{correct}$,C_{equal} and C_{wrong} of *length* , *policy* and *usage time* metrics for (a) T_{up} and (b) T_{down} events of beacon prefixes.

four methods. Figure 7 shows the result. As one can see from the figure, *Length* works very well in ranking paths explored during T_{down} events, giving 93% correct cases and 5% equal cases. However, it performs much worse in ranking the paths explored during T_{up} events, producing 40% correct cases and 40% wrong cases. During T_{down} events, many "invalid" paths are explored and they are very likely to be longer than the stable path. However during T_{up} events, only "valid" paths are explored and their preferences are not necessarily based on their path lengths.

Policy performs roughly equally for ranking paths during T_{down} and T_{up} events. It does not make many wrong choices, but produces a large number of equal cases (around 70% of the total). This demonstrates that the inferred AS relationship and routing policies provide *insufficient* information for path ranking. They do not take into account many details, such as traffic engineering, AS internal routing metric, etc., that affect actual routes being used. Compared with *Length*, *Policy+Length*

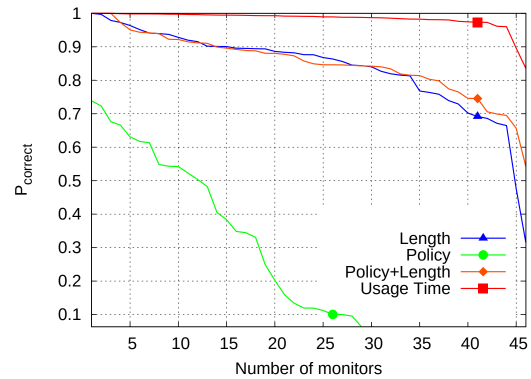


Fig. 8. Comparison between accuracy of *length*, *policy* and *usage time* metrics.

has a slightly worse performance with T_{down} events, and a moderate improvement with T_{up} events. Our observations are consistent with a recent study that concludes that per-AS relationships is not fine-grained enough to compute routing paths correctly [14].

Usage Time works surprisingly well and outperforms the other three in both T_{down} and T_{up} events. Its $P_{correct}$ is about 96.3% in T_{up} and 99.4% in T_{down} events. Its C_{equal} value is 0 in both T_{up} and T_{down} events. This is because we are measuring the path usage time using the unit of *second*, which effectively puts all the paths in strict rank order. We also notice that for T_{up} events, about 3.7% of the comparisons are *wrong*, whereas for T_{down} events this number is as low as 0.6%. We believe this noticeably percentage of *wrong* comparisons in T_{up} events is due to path changes caused by topological changes, such as a new link established between two ASes as a result of a customer switching to a new provider. Because the new paths have low usage time, our *Usage Time* based inference will give them a low rank, although these paths are actually the preferred ones. Nevertheless, the data confirmed our earlier assumption that, during our 1-month measurement period, there were no significant changes in Internet topology or routing polices, otherwise we would have seen a much higher percentage of *wrong* cases produced by *Usage Time*.

We now examine how the value of $P_{correct}$ varies between different monitors under each of the four path ranking methods. Figure 8 shows the distribution of $P_{correct}$ for different methods, with X-axis representing the monitors sorted in decreasing order of their $P_{correct}$ value. The value of $P_{correct}$ for each monitor is calculated over all the T_{down} and T_{up} events in our beacon data set. When using the path usage time for path ranking, we observe an accuracy between 84% and 100% across all the monitors, whereas with using path length for ranking, we observe the $P_{correct}$ value can be as low as 31% for some monitor. Using policy for path ranking leads to even lower $P_{correct}$ values.

After we developed and calibrated the usage time based path ranking method using beacon updates, we applied the method, together with the other three, to the BGP updates for *all* the prefixes collected from all the 50 monitors, and we obtained the results that are similar to that from the beacon update set. Considering the aggregate of all monitors

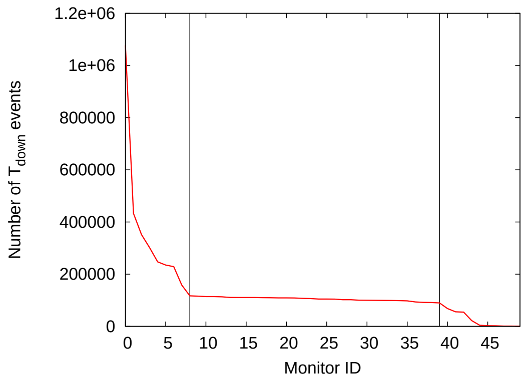


Fig. 9. Number of T_{down} events per monitor.

and all prefixes, $P_{correct}$ is 17% for *Policy*, 65% for *Length*, 73% for Policy+Length, and 96.5% for *Usage Time*. Thus we believe usage time works very well for our purpose and use it throughout our study.

To the best of our knowledge, we are the first to propose the method of using usage time to infer relative path preference. We believe this new method can be used for many other studies on BGP routing dynamics. For example, [12] pointed out that if after a routing event, the stable path is switched from P1 to P2, the root cause of the event should lie on the better path of the two. The study used length-only in their path ranking and the root cause inference algorithm produced a mixed result. Our result shows that using length for path ranking gives only about 65% accuracy, and usage time can give more than 96% accuracy. Using usage time to rank path can potentially improve the results of the root cause inference scheme proposed in [12].

III. Characterizing Events

After applying the classification algorithm to BGP data, we count the number of T_{down} events observed by each monitor as a sanity check. A T_{down} event means that a previously reachable prefix becomes unreachable, suggesting that the root cause of the failure is very likely at the AS that originates the prefix, and should be observed by all the monitors. Therefore, we expect every monitor to observe roughly the same number of T_{down} events. Figure 9 shows the number of T_{down} events seen by each monitor. Most monitors observe similar number of T_{down} events, but there are also a few outliers that observe either too many or too few T_{down} events. Too many T_{down} events can be due to failures that are close to monitors and partition the monitors from the rest of the Internet, or underestimation of the relative timeout T used to cluster updates. Too few T_{down} events can be due to missing data during monitor downtime, or overestimation of the relative timeout T. In order to keep consistency among all monitors, we decided to exclude the head and tail of the distribution, reducing the data set to 32 monitors.

Now we examine the results of event classification. Tables I and II show the statistics for January and February respectively for each event class, including the total number of events, the average event duration, the average number of updates per

	No. of Events ($\times 10^6$)	Duration (second)	No. of Updates	No. of Paths
T_{up}	3.39	45.26	2.30	1.59
T_{down}	3.35	116.34	4.10	1.95
T_{short}	7.37	31.32	1.71	1.27
T_{long}	8.04	69.93	2.52	1.62
T_{pdist}	15.51	174.19	4.66	2.33
T_{spath}	23.24	38.91	1.52	1.00

TABLE I

Event Statistics for Jan 2006 (31 days)

	No. of Events ($\times 10^6$)	Duration (second)	No. of Updates	No. of Paths
T_{up}	2.88	42.54	2.20	1.54
T_{down}	2.85	118.98	4.00	1.90
T_{short}	8.09	39.68	2.46	1.51
T_{long}	8.94	67.26	2.51	1.70
T_{pdist}	16.01	190.79	4.80	2.31
T_{spath}	20.44	30.42	1.44	1

TABLE II

Event Statistics for Feb 2006 (28 days)

event, and the average number of unique paths explored per event. We exclude T_{equal} events from the table since their percentage is negligible. Comparing the results from the two months, we note that the values are very close, as can also be observed by comparing the distribution of event duration on Figures 10 and 11. Given this similarity, we will base our following analysis on January data, although the same observations apply to February.

There are three observations. First, the three high-level event categories in Figure 4 have approximately the same number of events: *Path-Change* events are about 36% of all the events, *Same-Path* 34% and *Path-Disturbance* 30%. Breaking down *Path-Change* events, we see that the number of T_{down} balances that of T_{up}, and the number of T_{long} balances that of T_{short}. This makes sense since T_{down} failures are recovered with T_{up} events, and T_{long} failures are recovered with T_{short} events.

Second, the average duration of different types of events can be ordered as follows: $T_{short} < T_{spath} \simeq T_{up} < T_{long} \ll T_{down} < T_{pdist}$[4]. Figure 10 shows the distributions of event durations, [5] which also follow the same order. Note that the shape of the curves is stepwise with jumps at multiples of around 26.5 seconds. The next section will explain that this is due to the *MinRouteAdvertisementInterval* (MRAI) timer, which controls the interval between consecutive updates sent by a router. The default range of MRAI timer has the average value of 26.5 seconds, making events last for multiples of this value. Table I also shows that T_{pdist} events have the longest duration, the most updates and explore the most unique paths. This suggests that T_{pdist} likely contains two events very close in time, e.g., a link failure followed shortly by its recovery. A study [15] on network failures inside a tier-1 provider revealed that about 90% of the failures on high-failure

[4]The order of T_{spath} and T_{short} average durations invert on February 2006, even though the values remain very close to each other.

[5]The T_{spath} curve is omitted from the figure for clarity.

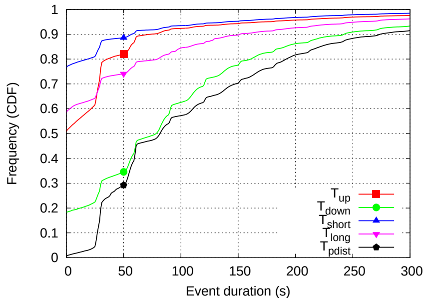


Fig. 10. Duration of events for January 2006.

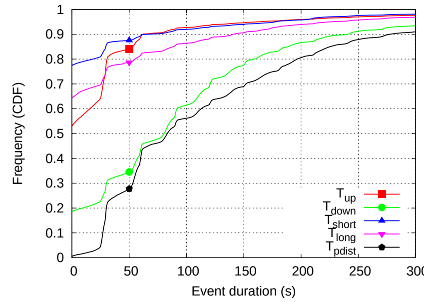


Fig. 11. Duration of events for February 2006.

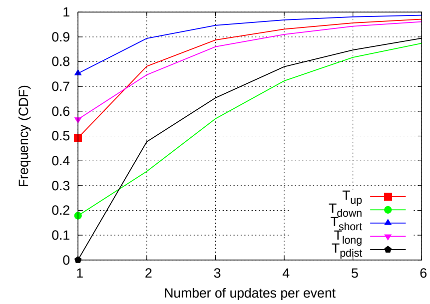


Fig. 12. Number of Updates per Event, January 2006.

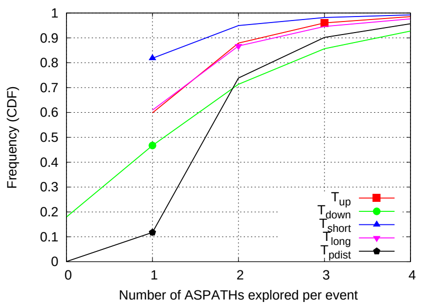


Fig. 13. Number of Unique Paths Explored per Event, January 2006.

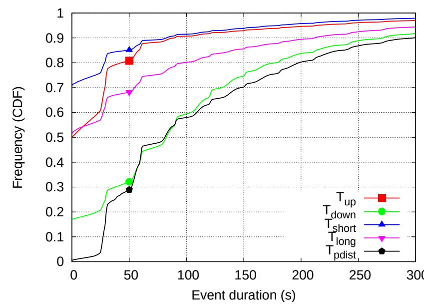


Fig. 14. Duration of events for unstable prefixes, January 2006.

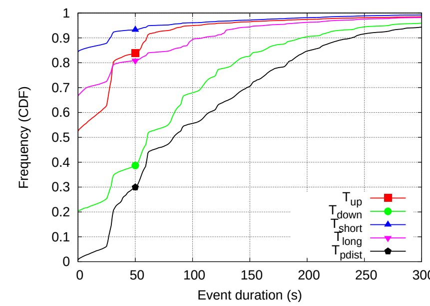


Fig. 15. Duration of events for stable prefixes, January 2006.

links take less that 3 minutes to recover, while 50% of optical-related failures take less than 3.5 minutes to recover. Therefore there are many short-lived network failures and they can very well generate routing events like T_{pdist}. On the other hand, T_{spath} events are much shorter and have less updates. It is because that T_{spath} is likely due to routing changes inside the AS hosting the monitor, and thus does not involve inter-domain path exploration.

Third, among the path changing events, T_{down} events last the longest, have the most updates, and explore the most unique paths. Figures 10, 12 and 13 show the distributions of event duration, number of updates per event, and number unique paths explored per event respectively. The results show that route fail-down events (T_{down}) last considerably longer than route fail-over events (T_{long}). In fact, Figure 10 shows that about 60% of T_{long} events have duration of zero, while 50% of T_{down} events last more than 80 seconds. In addition, Figure 12 shows that about 60% of T_{long} events have only 1 update, while about 70% of T_{down} events have 3 or more updates. Figure 13 shows that T_{down} explore more unique paths than T_{long}. These results are in accordance with our previous analytical results in [6], but contrary to the results of previous measurement work [2], which concluded that the duration of T_{long} events is similar to that of T_{down} and longer than that of T_{up} and T_{short}. In [6] we showed that the upper bound of T_{long} convergence time is proportional to $M(P-J)$, where M is the MRAI timer value, P is the path length of to the destination *after* the event, and J is the distance from the failure location to the destination. Since P is typically small for most Internet paths, and J could be anywhere between 0 and P, the duration of most T_{long} events should be short. We believe that the main reason [2] reached a different conclusion is because they conducted measurements by artificially increasing P to 30 AS hops using AS prepending. The analysis in [6] shows that an overestimate of P would result in a longer T_{long} convergence time, which would explain why they observed longer durations for beacon prefixes than what we observed for operational prefixes.

A. The Impact of Unstable Prefixes

So far we have been treating all destination prefixes in the same way by aggregating them in a single set in our measurements. However, previous work[10] showed that most of routing instabilities affect a small number of unstable prefixes, and popular destinations (with high traffic volume) are usually stable. Therefore, it might be the case that the results we just described are biased towards those unstable prefixes, since these prefixes are associated with more events. In order to verify if this is the case, we classify each prefix p into one of two classes, based on the number of events associated with it. If we let $\hat{E}$ be the median of the distribution of the number of events per prefix $E(p)$, then we can classify each prefix p in: (1) *unstable* if $E(p) \geq \hat{E}$, or (2) *stable* if $E(p) < \hat{E}$. From the 205,980 prefixes in our set, only 28,954 (or 14%) were classified as unstable, i.e. 14% of prefixes were responsible for 50% of events. In Figures 14 and 15 we show the distribution of event duration for unstable and stable prefixes respectively. Note that not only are these two distributions very similar, but they are also very close to the original distribution of the aggregate in Figure 10. Based on

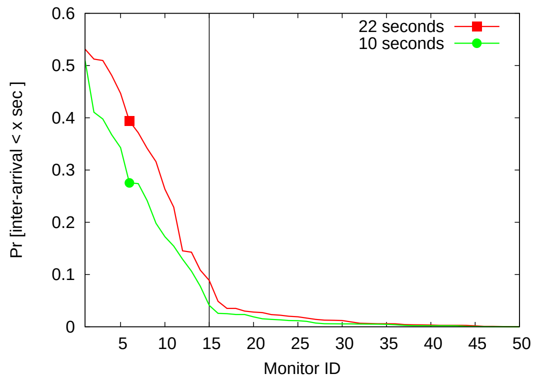


Fig. 16. Determining MRAI configuration.

these observations, we believe there is no sensitive bias in the aggregated results shown before.

IV. Policies, Topology and Routing Convergence

In this section we compare the extent of slow convergence across different prefixes and different monitors to examine the impacts of routing polices and topology on slow convergence.

A. MRAI Timer

In order to make fair comparisons of slow convergence observed by different monitors, we need to be able to tell whether a monitor enables MRAI timer or not. The BGP specification (RFC 4271 [16]) defines the *MinRouteAdvertisementInterval* (MRAI) as the minimum amount of time that must elapse between two consecutive updates sent by a router regarding the same destination prefix. Lacking MRAI timer may lead to significantly more update messages and longer global convergence time [17]. Even though it is a recommended practice to enable the MRAI timer, not all routers are configured this way. Since MRAI timer will affect observed event duration and number of updates, for the purpose of studying impacts of policies and topology, we should only make comparisons among MRAI monitors, or among non-MRAI monitors, but not between MRAI and non-MRAI monitors.

By default the MRAI timer is set to 30 seconds plus a jitter to avoid unwanted synchronization. The amount of jitter is determined by multiplying the base value (e.g., 30 seconds) by a random factor which is uniformly distributed in the range [0.75, 1]. Assuming routers are configured with the default MRAI values, we should (1) not observe consecutive updates spaced by less than $30 \times 0.75 = 22.5$ seconds for the same destination prefix, and (2) observe a considerable amount of inter-arrival times between 22.5 and 30 seconds, centered around the expected value, $30 \times \frac{0.75+1}{2} = 26.5$ seconds.

For each monitor, we define a *Non-MRAI Likelihood*, $L_{\overline{M}}$, as the probability of finding consecutive updates for the same prefix spaced by less than 22 seconds. Figure 16 shows $L_{\overline{M}}$ for all the 50 monitors in our initial set. Clearly, there are monitors with very high $L_{\overline{M}}$ and monitors with very small $L_{\overline{M}}$. The curve has a sharp turn, hinting a major configuration change. Based on this, we decided to set $L_{\overline{M}} = 0.05$ as a threshold to differentiate MRAI and non-MRAI monitors. Those with $L_{\overline{M}} < 0.05$ are classified as MRAI monitors, and those with $L_{\overline{M}} \geq 0.05$ are classified as non-MRAI monitors. However, there could still be cases of non-MRAI monitors with MRAI timer configuration just slightly bellow the RFC recommendation, and would therefore be excluded using our method. In order to assure this was not the case, we show in Figure 16 the curve corresponding to the probability of finding consecutive updates spaced by less than 10 seconds. We note that the 10-second curve is very close to the 22-second curve, and therefore we are effectively only excluding monitors that depart significantly from the 30-second base value of the RFC.

Using this technique, we detect that 15 routers from the initial set of 50 are non-MRAI (see the vertical line in Figure 16), and 10 of them are part of the set of 32 routers we used in previous section. We will use this set of 32-10=22 monitors for the next subsection to compare the extent of slow convergence across monitors.

B. The Impact of Policy and Topology on Routing Convergence

Internet routing is policy-based. The “no-valley” policy [13], which is based on inter-AS relationships, is the most prevalent one in practice. Generally, most ASes have relationships with their neighbors as provider-customer or peer-peer. In provider-customer relationship, the customer AS pays the provider AS to get access service to the rest of the Internet; in peer-peer relationship, the two ASes freely exchange traffic between their respective customers. As a result, a customer AS does not forward packets between its two providers, and a peer-peer link can only be used for traffic between the two incident ASes’ customers. For example, in Figure 19, paths [C E D], [C E F] and [C B D] all violate the “no-valley” policy and generally are not allowed in the Internet.

Based on AS connectivity and relationships, the Internet routing infrastructure can be viewed as a hierarchy.

- *Core*: consisting of a dozen or so tier-1 providers forming the top level of the hierarchy.
- *Middle*: ASes that provide transit service but are not part of the core.
- *Edge*: stub ASes that do not provide transit service (they are customers only).

We collect an Internet AS topology [18], infer inter-AS relationships using the algorithm from [19], and then classify all ASes into these three tiers. *Core* ASes are manually selected based on their connectivity and relationships with other ASes [18]; *Edge* ASes are those that only appear at the end of AS paths; and the rest are *middle* ASes. With this classification, we can locate monitors and prefix origins with regard to the routing hierarchy.

Our set of 22 monitors consists of 4 monitors in the *core*, 15 in the *middle* and 3 at the *edge*. We would like to have a more representative set of monitors at the *edge*, but we only found these many monitors in this class with consistent data from the RouteViews and RIPE data archive. The results presented in this subsection might not be quantitatively accurate due to the limitation of monitor set, but we believe they still illustrate qualitatively the impact of monitor location on slow convergence.

In the previous section we showed that T_{down} events have both the longest convergence time and the most path exploration from all path change events. Furthermore, in a T_{down} event, the root cause of the failure is most likely inside the destination AS, and thus all monitors should observe the same set of events. Therefore, the T_{down} events provide a common base for comparison across monitors and prefixes, and the difference between convergence time and the number of updates should be most pronounced. In this subsection we examine how the location of prefix origins and monitors impact the extent of slow convergence.

Figure 17 shows the duration of T_{down} events seen by monitors in each tier. The order of convergence time is $core < middle < edge$, and the medians of convergence times are 60, 84 and 84 seconds for *core*, *middle* and *edge* respectively. Taking into account that our *edge* monitor ASes are well connected: one has 3 providers in the *core* and the other two reach the *core* within two AS hops, we believe that in reality *edge* will generally experience even longer convergence times than the values we measured. Figure 18 shows that monitors in the *middle* and at the *edge* explore 2 or more paths in about 60% of the cases, whereas monitors in the *core* explore at most one path in about 65% of the cases.

In a T_{down} event, the monitor will not finish the convergence process until it has explored all alternative paths. Therefore, the event duration depends on the number of alternative paths between the event origin and the monitor. In general, due to no-valley policy [13], tier-1 ASes have fewer paths to explore than lower tier ASes. For example, in Figure 19, node D (representing a tier-1 AS) has only one no-valley path to reach node G (path 4), while node E has three paths to reach the same destination: paths 1,2 and 3. In order to reach a destination, tier-1 ASes can only utilize provider-customer links and peer-peer links to other tier-1s, but a lower tier AS can also use customer-provider links and peer-peer links in the *middle* tier, which leads to more alternative paths to explore during T_{down} events.

We have studied how T_{down} events are experienced by monitors in different tiers. Now we study how the *origin* of the event impacts the convergence process. Note that we must divide again the results according to the monitor location, otherwise we may introduce bias caused by the fact that most of our monitors are in the *middle* tier. We use the notation $x \rightarrow y$, where x is the tier where the T_{down} event is originated from and y is the tier of the monitor that observe the event. In our measurements, we observed that the convergence times of $x \rightarrow y$ case were close to the $y \rightarrow x$ case. Therefore, from these two cases we will only show the case where we have a higher percentage of monitors. For instance, between $core \rightarrow edge$ and $edge \rightarrow core$ cases we will only show the later since our monitor set covers about 27% of the *core* but only a tiny percentage of the *edge*. Figure 20 shows the duration of T_{down} events for prefixes originated and observed at different tiers. We omit the cases $middle \rightarrow core$ and $middle \rightarrow middle$ for clarity of the figure, since they almost overlap with curves $edge \rightarrow core$ and $edge \rightarrow middle$ respectively. The figure shows that the $core \rightarrow core$ case is the fastest, and the $edge \rightarrow middle$, $edge \rightarrow edge$ cases are the slowest. This observation is also confirmed by Figure 21, which shows the number of paths explored during T_{down}. Table 22 lists the median durations of T_{down} events originated and observed at different tiers. Events observed by the *core* have shortest durations, which confirms our previous observation (Figure 17). Note that the $edge \rightarrow edge$ convergence is slightly faster than the $edge \rightarrow middle$ convergence. We believe this happens because, as mentioned before, our set of *edge* monitors are very close to the *core*. Therefore, they may not observe so much path exploration as the *middle* monitors, which may have a number of additional peer links to reach other *edge* nodes without going through the *core*.

Note that we expect that the $edge \rightarrow edge$ case reflects most of the *slow* routing convergence observed in the Internet because about 80% of the autonomous systems in the Internet are at the *edge*, and about 68% of the T_{down} events are originated at the *edge*, as will be shown in the next subsection.

C. Origin of Fail-down Events

We will now examine *where* the T_{down} events are originated in the Internet hierarchy. Since we expect the set of T_{down} events be common to all the 32 monitors of our data set (section III), we will use in this subsection a single monitor, the router 144.228.241.81 from Sprint. Note that similar results are obtained from other monitors.

Because our data set spans over 1 month period, we do not know if during this time there was any high-impact event that triggered an abnormal number of T_{down} failures, which could bias our results if we simply use daily count or hourly count. Instead, Figure 23 plots the cumulative number of T_{down} events as observed by the monitor during January 2006, and the time granularity is second. The cumulative number of events grows linearly, with an approximate constant number of 3,600 T_{down} events per day. This uniform distribution along the time dimension seems also to suggest that most fail-down events have a random nature.

Table III shows the break down of T_{down} events by the tier that they are originated from. We observe that about 68% of the events are originated at the *edge*. However, the *edge* also announces a chunk of 56% of the prefixes. Therefore, in order to assess the stability of each tier, and since our identification of events is based on prefix, a simple event count is not enough. A better measure is to divide the number of events originated at each tier by the total number of prefixes originated from that tier. The row "No. events per prefix" in Table III shows that if the *core* originates n events per prefix, the *middle* originates $2 \times n$ and the *edge* originates $3 \times n$ such events, yielding the interesting proportion 1:2:3. This seems to indicate that generally, prefixes in the *middle* are twice as unstable as prefixes in the *core*, and prefixes at the *edge* are three times as unstable as prefixes in the *core*.

D. Impact of Fail-down Convergence

The ultimate goal of routing is to deliver data packets. One may argue that although T_{down} events have the longest convergence time, they do not make the performance of data delivery

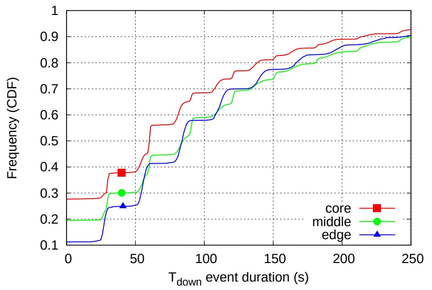


Fig. 17. Duration of T_{down} events as seen by monitors at different tiers.

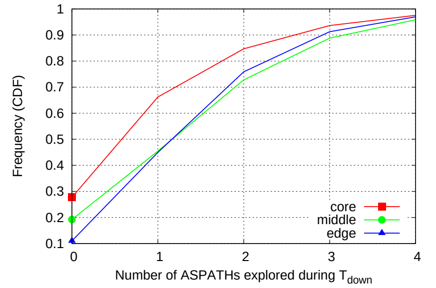


Fig. 18. Number of unique paths explored during T_{down} as seen by monitors at different tiers.

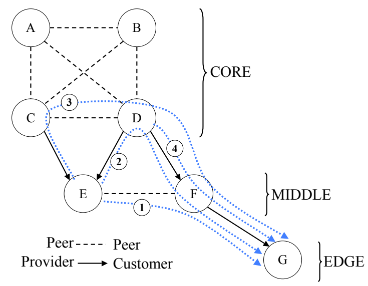


Fig. 19. Topology example.

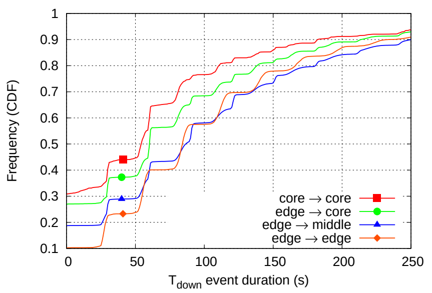


Fig. 20. Duration of T_{down} events observed and originated in different tiers.

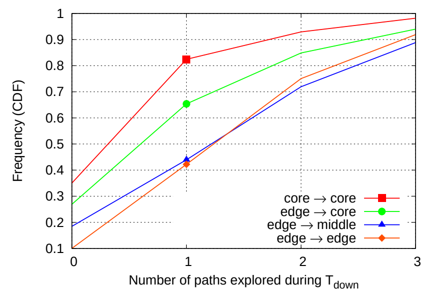


Fig. 21. Number of paths explored during T_{down} events observed and originated in different tiers.

T_{down} duration (s)	
core→core	54
middle→core	60
edge→core	61
middle→middle	83
edge→edge	85
edge→middle	87

Fig. 22. Median of duration of T_{down} events observed and originated in different tiers.

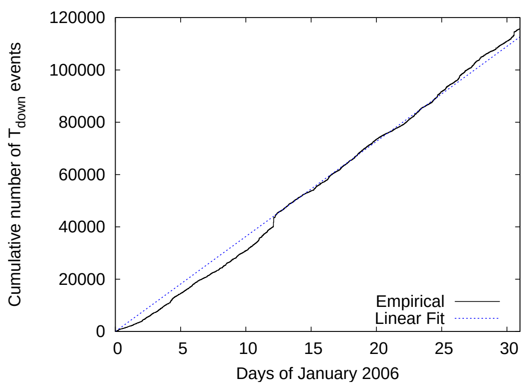


Fig. 23. Number of T_{down} events over time.

	Core	Middle	Edge
No. of events	3,011	34,514	78,149
No. of prefixes originated	14,367	81,988	122,877
No. of events per prefix	0.21	0.42	0.63

TABLE III
T_{down} EVENTS BY ORIGIN AS

worse because the data packets would be dropped anyway if the prefix is unreachable. However, this is not necessarily true. In the current Internet, sometimes the same destination network can be reached via multiple prefixes. Therefore, the failure to reach one prefix does not necessarily mean that the destination is unreachable, because the destination may be reachable via another prefix.

Figure 24 shows a typical example. Network A has two providers, B and C. To improve the availability of its Internet access, A announces prefix 131.179/16 via B and prefix 131.179.100/24 via C. In this case, 131.179/16 is called the "covering prefix" [20] of 131.179.100/24. As routing is done by longest prefix match, data traffic destined to 131.179.100/24 normally takes link A-C to enter network A. When link A-C fails, ideally, data traffic should switch to link A-B quickly with minimal damage to data delivery performance. However, the failure of link A-C will result in a T_{down} event for 131.179.100/24. Before the convergence process completes, routers will keep trying obsolete paths to 131.179.100/24, rather than switching to paths towards 131.179/16. This can result in packet lost and long delays, which probably will have serious negative impacts on data delivery performance.

We analyzed routing tables from RouteViews and RIPE monitors to see how frequent are the scenarios illustrated by Figure 24. The result shows that routing announcements like the one in Figure 24 are a common practice in the Internet. In the global routing table, 50% of prefixes have covering prefixes being announced through a different provider, and are therefore vulnerable to the negative impacts caused by fail-down convergence. A recent study [21] showed that about 50% of VOIP glitches as perceived by end users may be caused by BGP slow convergence.

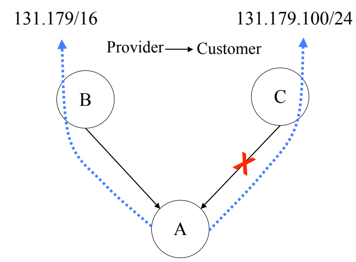


Fig. 24. Case where T_{down} convergence disrupts data delivery.

V. Related Work

There are two types of BGP update characterization work in the literature: passive measurements [22], [23], [24], [25], [26], [27], [10], [28], [12], and active measurements [1], [2], [3]. The work presented in this paper belongs to the first category. We conducted a systematic measurement to classify routing instability events and quantify path exploration for all the prefixes in the Internet. Our measurement also showed the impact of AS's tier level on the extent of path explorations.

Existing measurements of path exploration and slow convergence have all been based on active measurements [1], [2], [3], where controlled events were injected into the Internet from a small number of beacon sites. These measurement results demonstrated the existence of BGP path exploration and slow convergence, but did not show to what extent they exist on the Internet under real operational conditions. In contrast, in this paper we classify routing events of all prefixes, as opposed to a small number of beacon sites, into different categories, and for each category we provide measurement results on the updates per event and event durations. Given we examine the updates from multiple peers for all the prefixes in the global routing table, we are able to identify the impact of AS tier levels on path exploration. Regarding the relation between the tier levels of origin ASes, our results agree with previous active measurement work [2] (using a small number of beacon sites) that prefixes originated from tier-1 ASes tend to experience less slow convergence compared to prefixes originated from lower tier ASes. Moreover, our results also showed that, for the same prefix, routers of different AS tiers observe different degree of slow convergence, with tier-1 ASes seeing much less than lower tier ASes.

Existing passive measurements have studied the instability of all the prefixes. The focuses have been on update inter-arrival time, event duration, and location of instability, and characterization of individual updates [22], [23], [24], [25], [26], [27], [10], [28], [12]. There is no previous work on classifying routing events according to their effects (e.g. whether path becomes better or worse after the event). Our paper describe a novel path preference heuristic based on path usage time, and studied in detail the characteristics of different classes of instability events in the Internet.

Our approach shares certain similarity with [10], [28], [12] in that we all use a timeout-based approach to group updates into events. Such an approach can mistakingly group updates of multiple root causes that happened close to each other or overlapped in time into a single event. As we discussed earlier, the events in our *Path-Disturbance* category can be examples of grouping updates of overlap root causes, because the path to a prefix changed at least twice, and often more times, during one event. We moved a step forward by detecting and separating these overlapping events into a different category. It is most likely that those *Path-Change* events with very long durations are also overlapping events, and one possible way to identify them is to set a time threshold on the event duration, which we plan to do in the future.

VI. Conclusions

We conducted the first systematic measurement study to quantify the existence of path exploration and slow convergence in the global Internet routing system. We first developed a new path ranking method based on the usage time of each path and validated its effectiveness using data from controlled experiments with beacon prefixes. we then applied our path ranking method to BGP updates of all the prefixes in the global routing table and classified each observed routing event into three classes: Path Change, Path Disturbance, and Same Path. For Path Change events, we further classified them into 4 sub-categories: T_{down}, T_{up}, T_{long}, and T_{short}. We measured the path exploration, convergence duration, and update count for each type of events.

Our work shows several significant results. First, although there is a wide existence of path exploration and slow convergence in the global routing system, the significance of the problem can vary considerably depending on the locations of both the origin ASes and the observation routers in the routing system hierarchy. In general, routers in tier-1 ISPs observe less path exploration and shorter convergence delays than routers in edge ASes, and prefixes originated from tier-1 ISPs also experience much less slow convergence than those originated from edge ASes.

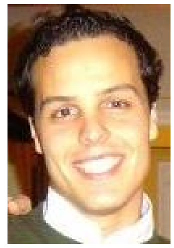
Second, T_{long} events have short duration in general that are comparable to that of T_{up} and T_{short} events. This is in accordance to our previous theoretical analysis results presented in [6], and is a noticeable departure from widely accepted views based on the previous experiments [1].

Furthermore, our data shows that the Same Path events account for about 34% of the total routing events, which seems an alarmingly high value. Since this class of events is most likely caused by internal routing changes within individual ASes, most of them probably should not have existed in the first place. Further investigations are needed to better understand the causes of the Same Path events. We also observed that about 30% of the routing events are due to *transient route changes* (which are captured as path disturbance events in our measurement), and are responsible for close to half of all the routing updates (47%). It would be interesting to identify the causes of these transient routing changes in order to further stabilize the global routing system.

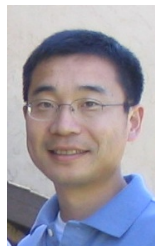
References

[1] C. Labovitz, A. Ahuja, A. Abose, and F. Jahanian, "Delayed Internet routing convergence," *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, pp. 293 – 306, June 2001.

[2] C. Labovitz, A. Ahuja, R. Wattenhofer, and S. Venkatachary, “The impact of Internet policy and topology on delayed routing convergence,” in *Proc. of IEEE INFOCOM*, April 2001.
[3] Z. M. Mao, R. Bush, T. Griffin, and M. Roughan, “BGP beacons,” in *ACM SIGCOMM Internet Measurement Conference (IMC)*, 2003.
[4] “The RouteViews project,” http://www.routeviews.org/.
[5] “The RIPE Routing Information Services,” http://www.ris.ripe.net.
[6] D. Pei, , B. Zhang, D. Massey, and L. Zhang, “An analysis of path-vector routing protocol convergence algorithms,” *Computer Networks*, vol. 50, no. 3, pp. 398 – 421, 2006.
[7] “PSG Beacon List.” [Online]. Available: http://www.psg.com/~zmao/BGPBeacon.html
[8] “RIPE Beacon List.” [Online]. Available: http://www.ripe.net/ris/docs/beaconlist.html
[9] B. Zhang, V. Kambhampati, M. Lad, D. Massey, and L. Zhang, “Identifying BGP Routing Table Transfers,” in *ACM SIGCOMM Mining the Network Data (MineNet) Workshop*, August 2005.
[10] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang, “BGP routing stability of popular destinations,” in *ACM SIGCOMM Internet Measurement Workshop (IMW)*, 2002.
[11] D. Chang, R. Govindan, and J. Heidemann, “The temporal and topological characteristics of BGP path changes,” in *Proc. of the Int’l Conf. on Network Protocols (ICNP)*, November 2003.
[12] A. Feldmann, O. Maennel, Z. M. Mao, A. Berger, and B. Maggs, “Locating internet routing instabilities,” in *Proc. of ACM SIGCOMM*, 2004.
[13] L. Gao, “On inferring autonomous system relationships in the Internet,” *ACM/IEEE Transactions on Networking*, vol. 9, no. 6, pp. 733–745, 2001.
[14] W. Mühlbauer, S. Uhlig, B. Fu, M. Meulle, and O. Maennel, “In search for an appropriate granularity to model routing policies,” in *Proc. of ACM SIGCOMM*, 2007.
[15] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, and C. Diot, “Characterization of failures in an IP backbone,” in *IEEE Infocom*, Hong Kong, 2004, Sprint ATL Research Report.
[16] Y. Rekhter, T. Li, and S. Hares, “Border Gateway Protocol 4,” Internet Engineering Task Force, RFC 4271, January 2006.
[17] T. G. Griffin and B. J. Premore, “An experimental analysis of bgp convergence time,” in *Proc. of the Int’l Conf. on Network Protocols (ICNP)*, 2001.
[18] B. Zhang, R. Liu, D. Massey, and L. Zhang, “Collecting the internet as-level topology,” *ACM SIGCOMM Computer Communications Review (CCR)*, vol. 35, no. 1, pp. 53–62, January 2005.
[19] J. Xia and L. Gao, “On the evaluation of AS relationship inferences,” in *Proc. of IEEE GLOBECOM*, December 2004.
[20] X. Meng, Z. Xu, B. Zhang, G. Huston, S. Lu, and L. Zhang, “IPv4 address allocation and BGP routing table evolution,” in *ACM SIGCOMM Computer Communication Review (CCR) special issue on Internet Vital Statistics*, Janurary 2005.
[21] N. Kushman, S. Kandula, and D. Katabi, “Can you hear me now?!: it must be BGP,” *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 2, pp. 75–84, 2007.
[22] C. Labovitz, G. Malan, and F. Jahanian, “Internet Routing Instability,” in *Proc. of ACM SIGCOMM*, September 1997.
[23] C. Labovitz, R. Malan, and F. Jahanian, “Origins of Internet routing instability,” in *Proc. of IEEE INFOCOM*, 1999.
[24] C. Labovitz, A. Ahuja, and F. Jahanian, “Experimental study of Internet stability and wide-area network failures,” in *Proceedings of FTCS99*, June 1999.
[25] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, A. Mankin, S. F. Wu, and L. Zhang, “Observation and analysis of BGP behavior under stress,” in *ACM SIGCOMM Internet Measurement Workshop (IMW)*, 2002.
[26] D. Andersen, N. Feamster, S. Bauer, and H. Balakrishnan, “Topology inference from bgp routing dynamics,” in *ACM SIGCOMM Internet Measurement Workshop (IMW)*, 2002.
[27] O. Maennel and A. Feldmann, “Realistic BGP traffic for test labs,” in *Proc. of ACM SIGCOMM*, 2002.
[28] 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 *Symposium on Networked System Design and Implementation (NSDI)*, May 2005.



Ricardo Oliveira received the B.S. in Electrical Engineering from the Engineering Faculty of Porto University (FEUP), Portugal in 2001 and the M.S. degree in Computer Science from University of California, Los Angeles in 2005. He has been pursuing the Ph.D. in Computer Science at University of California, Los Angeles, since 2005. His research interests include Internet topology, next generation routing architectures and development of Internet monitoring and measurement platforms. He is a student member of the ACM and the IEEE.



Beichuan Zhang is an Assistant Professor in the Department of Computer Science at the University of Arizona. His research interests include Internet routing and topology, multicast, network measurement, and security. He received Ph.D. in Computer Science from the University of California, Los Angeles (2003) and B.S. from Beijing University, China (1995).

Dan Pei is a researcher at AT&T Research. He received his Ph.D. degree from UCLA in 2005, and his B.S. and M.S. degrees from Tsinghua University in 1997 and 2000. His current research interests are network measurement and security.

Lixia Zhang received her Ph.D in computer science from the Massachusetts Institute of Technology. She was a member of the research staff at the Xerox Palo Alto Research Center before joining the faculty of UCLA’s Computer Science Department in 1995. In the past she has served as the vice chair of ACM SIGCOMM, Co-Chair of IEEE Communication Society Internet Technical Committee, and on the editorial board for the IEEE/ACM Transactions on Networking. Zhang is currently serving on the Internet Architecture Board.