Inferring the Origin of Routing Changes using Link Weights

Mohit Lad*, Ricardo Oliveira*, Dan Massey†, and Lixia Zhang*

*Computer Science Department, University of California, Los Angeles CA 90095
Email: {mohit,rveloso,lixia}@cs.ucla.edu
†Computer Science Department, Colorado State University, Fort Collins, CO 80523
Email: massey@cs.colostate.edu

***Abstract*— The global Internet routing infrastructure is a large and complex distributed system where routing changes occur constantly. Our objective in this paper is to develop a simple and effective inference solution that can identify the AS or inter-AS link failures that trigger large scale routing changes in *near realtime*. We achieve this goal through a novel approach based on *link weights*. We measure the weight of each inter-AS link by the number of routes carried over that link, and keep track of its expected value and variance. We then correlate the weight changes of adjacent links and use a min-cut heuristic to find candidates for the origin of change. This work makes three contributions. First, we keep track of link weights rather than the routes of individual prefixes and thus our analysis is based on an aggregate view. Second, we use expected value and mean deviation of the link weights to identify routing events and distinguish route changes caused by failures from those by recoveries. Finally we use a min-cut heuristic based on the classification of routing events to accurately identify the AS or inter-AS link most likely responsible for the observed route changes. We verified our design using BGP data collected from operational Internet. Our efficient and accurate routing diagnosis solution can greatly help us gain better understanding of the dynamics in the operational Internet.**

I. Introduction

The global Internet routing infrastructure is a large and complex system. Although the inter-domain routing protocol, BGP, seems simple by its specification, its behavior can be very complex in this fully distributed and highly dynamic environment where tens of thousands of routers interact through BGP message exchanges. Yet it is crucially important, both practically and theoretically, to be able to identify the origins of routing changes. This ability enables network operators to quickly locate the sources of significant routing events, and enables researchers to gain further understanding of the routing protocol's reactions to those events. In this paper, we present a technique to efficiently detect significant events and identify the Autonomous Systems (ASes) or inter-AS links that caused the events.

We begin by considering the view from an arbitrary vantage point in the Internet, a BGP router in an ISP or enterprise network. One can hope to diagnose events by observing the stream of BGP updates received by the router. The paths to some prefixes change, some other prefixes become unreachable, and perhaps some new prefixes also appear. Given the sheer scale and dense connectivity of the global Internet, a typical BGP router receives a constant stream of update messages. For example, one BGP router monitored by the Oregon RouteViews received over 15 million routing updates during January 2007. Furthermore, the dense connectivity and routing policies lead to the fact that routers in different locations have different views regarding the routes and route changes to individual prefixes. These two factors suggest that previous approaches to the origin inference by tracking route changes to specific prefixes (e.g. the work in [4]) would require routing data collected from a large number of vantage points and thus incur a heavy load of data processing. To develop a simple and effective inference solution, we must start from a different point.

We design a new inference scheme using the abstract measures of *link weight* and *weight changes* developed in our previous work, the Link-Rank tool [9]. Link-Rank extracts the total number of routes carried over each inter-AS link in the Internet topology, called link weight, and measures the changes in the number of routes on each link to capture aggregate routing changes. This provides a concise representation of the view from a particular BGP router. We further leverage our previous observations that, among multiple alternative paths to a given destination, the most preferred path is used most of the time [12]. As a direct corollary from this observation, each AS link is expected to have a stable weight, and deviations from this expected value can serve as indications of significant routing changes.

Once a significant deviation is detected, our objective is to identify the origin of this deviation. Given the view from a single router, we can use a min-cut heuristic to identify the most likely faulty AS node or AS-AS link, enabling an isolated BGP router with only its own update stream to identify significant events and infer the origin of these events. By correlating changes observed from different monitors, we can achieve a very high degree of accuracy identifying the AS node or AS-AS link responsible for triggering the event. This can all be done in near real-time and provides useful tool for network operations and for understanding BGP protocol behavior in the aggregate. We validated our heuristic by accurately identifying session problems reported by Abilene with its peers. Our evaluation on events where problem area is adjacent to the origin AS shows that we could achieve an accuracy of close to 95%. On applying our heuristic over one month of BGP data, we found various interesting routing instabilities, some recurring again and again, clearly highlighting the need to be able to identify origin of changes on a regular basis.

The remainder of the paper is organized as follows. Section II provides background on Internet routing, BGP data collection, and our LinkRank approach of assigning weights. Section III presents our scheme for identifying events and Section IV describes the min-cut heuristic. Section V validates our scheme and presents the results of using our approach on BGP data from the Internet. Section VI discusses some open issues. Section VII presents related work and Section VIII concludes the paper.

II. Background

A. Internet Routing and BGP

The Internet consists of a large number of networks called autonomous systems (AS). Each AS is assigned an AS number and contains one or multiple destination networks. Each destination network is represented by an IP address prefix. For example, the prefix 131.179.96.0/24 represents a network at UCLA and is part of AS 52 (UCLA's AS number). As of July 2007, the Internet consists of over 25,000 autonomous systems and over 220,000 prefixes.

Border Gateway Protocol (BGP) [13] is the de-facto routing protocol used between autonomous systems in the Internet today. BGP is a path vector protocol and routing information in BGP is propagated by the sending BGP update messages. A BGP update message contains information about the destination prefix and the AS path used to reach that prefix. Figure 1 shows how BGP updates propagate routing information in the Internet. In this figure, AS 22 owns a prefix P1 and sends a BGP update message $\{P1 : 22\}$ to its neighbor AS 33. AS 22 is said to be the origin AS for prefix P1. On receiving this update, AS 33 now prepends its own AS number to the received path and sends the BGP update $\{P1 : 33, 22\}$ to its neighbors, AS 44 and AS 55. Note, AS 44 receives two paths to reach $P1$ and it chooses one of them as the primary path based on its routing policies.

In Figure 1, a router in AS 44 is attached to a collection box that receives and logs BGP updates. The collection box could be some private log system setup by AS 44 or could be a public log of BGP updates such as those provided by RouteViews [11] and RIPE [15]. In the figure, router R in AS 44 serves as our observation point and the update data provides a *view from operational router* R *in AS 44*. An observation point is always a specific router and different routers in the same AS may provide different views. But if there is only one router R being monitored in an AS A, we refer to view from router R in AS A as simply the *view from AS A*.

B. Inferring origin of routing changes

The origin of a routing change is the AS that first sends out an update message indicating a change in BGP route. This change then propagates through the Internet and is observed at different BGP routers. The origin can also be defined as

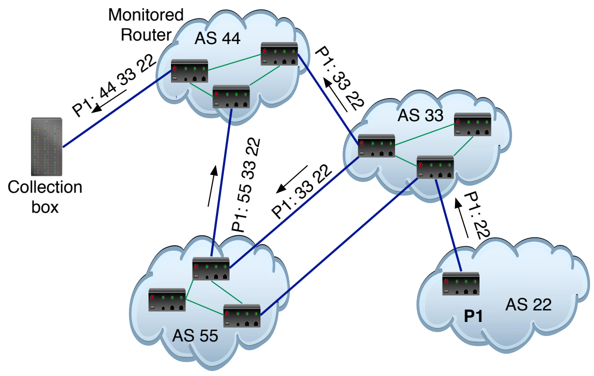


Fig. 1. Internet routing and BGP monitoring

an AS-AS link if the reason the update was sent was due to an issue with that particular link. For example, suppose the link between AS 44 and AS 33 failed in Figure 1. Depending on how quickly and efficiently BGP converges, our collection point may receive a few updates. Eventually our collection point will receive a BGP update reporting the new path to $P1$ as $\{P1 : 44, 55, 33, 22\}$. The origin of this change is the link between AS 44 and AS 33. The problem of inferring origin of change involves examining the observed BGP route changes and inferring the AS that sent originated the BGP route change.

However our objective is not to investigate every single AS path change. Internet routing is very dynamic and routes are constantly changing with a typical observation point logging over 15 million updates in a month. In this work, we focus on aggregate routing changes that stand out from day to day activity. To aggregate changes together and identify events, we begin with some of our previous work from the Link-Rank representation and visualization toolset.

C. Link-Rank graphs

Link-Rank graphs are used to visualize routing events involving multiple prefixes. Link-Rank takes as input the updates received from an observation point (BGP router) and uses the AS path information in updates to construct that observation point's view of the logical AS connectivity. In particular, Link-Rank assigns a weight to each logical AS-AS link that reflects the number of BGP routes carried over that link. Due to differences in BGP polices, the same link can have a different weight when viewed from different observation points. Unless otherwise specified, in the rest of the paper, link weight is tied to a specific observation point. Each update from the observation point may report some change in path for some prefixes and thus may change the link weight for one or more logical links. Overall, an observation point can provide a logical topology where each link in the topology has a weight that reflects the number of prefixes carried on the link and each update potentially changes the weights.

These changes can be captured using a rank change graph such as the one shown in Figure 2. The figure shows an event observed from a router in AS 11686. Red edges (with

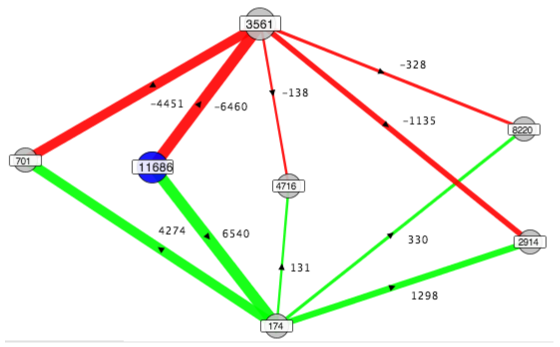


Fig. 2. Example of a Link-Rank graph from AS 11686

negative weights) indicate that a link has lost of routes and the weight equals the number of routes lost, while green edges indicate a gain of routes (and thus gain of weight) on that link. Arrow direction indicates how traffic would flow from the observation point to the destinations with the observation point colored blue. This simple Link-Rank graph shows the main link weight changes involved in a routing event. In this case, the origin of the event was a problem on the link between 11686 and 3561. Link-Rank is meant to be a tool to aid network operators understand changes and relies on user insight in making inferences about event. Systematically identifying events and finding their origin is non-trivial and is the focus of this paper.

While Link-Rank graphs and concepts provide an aggregated view of routing from an observation point, the link weights are in constant flux. BGP routing is dynamic and there are constant changes going on in the Internet. These changes result in changes in link weights. However, there is reason to believe one could establish a normal value for a given logical link. Prior work in understanding BGP dynamics indicated that majority of routes in the Internet do not change frequently, and majority of the routing changes can be narrowed down to a small percentage of prefixes [14], [6]. Furthermore, our previous measurements showed that among multiple alternative paths to a given destination, the most preferred path is used most of the time [12]. The combined results from the above observations lead to the expectation that the number of routes carried over most AS-AS link could be captured as some sort of an expected normal value.

III. Event Detection

In this section, we explain how we characterize link weights and identify *routing events*. Our first objective is to identify the *expected* value of a link over a period of time. In addition to this expected value, we are also interested in knowing how *stable* the link weight is over a period of time. Using these two values we try to identify events by looking for irregular behavior.

We sample the link weight every T time units and define an exponential moving average of link weight to get the *expected Weight*. The expected weight $\bar{w}_t(l)$ at time t takes into account the past history of the link weight as well as present and is computed as

$$\bar{w}_t(l) = (1-\alpha)\cdot \bar{w}_{t-1}(l) + \alpha \cdot w_t(l)$$

where $w_t(l)$ is the current weight of the link l, and α decides how much importance is assigned to the current value compared to the past values.

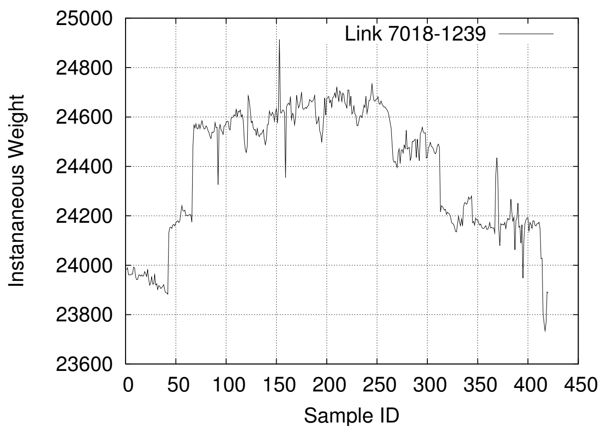
It is not enough to just capture the expected weight and one also needs to take into account the typical variance of this link. This variance will be an important factor in identifying events. Figure 3(a) shows the instantaneous weight of link 7018-1239 as seen from a BGP router in AS 7018. Note that the Y-range in this graph starts from 23,600. We can see a few spikes in the instantaneous weight as well as some longer lasting changes (plateaus). Our approach is to characterize how much link weights fluctuate and look for deviations that exceed some normal value. We capture this deviation using *mean deviation* $\delta_t(l) = |\bar{w}_t(l) - w_t(l)|$, where $\bar{w}_t(l)$ is the expected weight of link l, and $w_t(l)$ is the current observed weight of the link. To take into account the history of deviation, we define an exponential moving average of the mean deviation as

$$\bar{\delta}_t(l) = (1-\beta)\cdot \bar{\delta}_{t-1}(l) + \beta \cdot |\bar{w}_t(l) - w_t(l)|$$

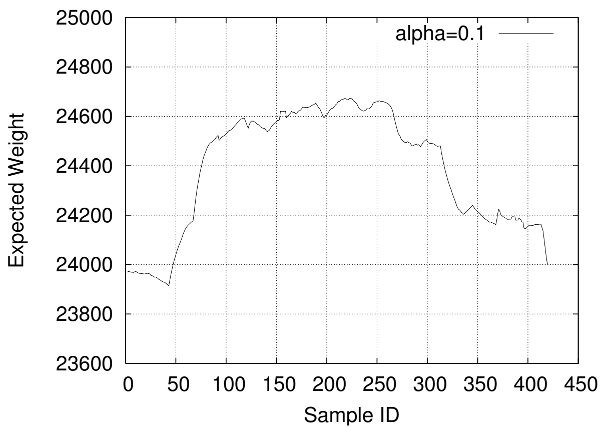
Our objective in assigning values to the sampling interval T, α and β is to get an expected behavior that is not influenced by very short lived changes (especially convergence events), while at the same time being able to adapt to longer lasting changes quickly enough. We want the value of T to be greater than the convergence time, which has been found to be 2 minutes on average [12] and sometimes be as long as 5 minutes. Based on this, we conservatively set the sampling interval $T = 10$ minutes. We wanted the expected weight to be resilient against short lived changes in weights due to failures. We picked a few random links from the set of heavy weight links, medium weight links and low weight links. We plotted the instantaneous samples for all these links over a 10 day period. We then plotted the expected weight $\bar{w}_t(l)$ for different values of α ranging from 0.1 to 0.9. From our observations, $\alpha = 0.1$ provided a close approximation of the link weights while not being affected by the short lived spikes. Figure 3(b) shows the expected weight for (7018,1239) with $\alpha = 0.1$, and we can see that the curve closely follows the instantaneous values except for the short lived spikes. Following similar studies for deviation, we picked $\beta = 0.25$. With an accurate knowledge of mean failure and recovery times, one may be able to better tune the value of α and β.

A. Link Events

We define a *link event* as a significant deviation from the expected weight of a link. The mean deviation $\bar{\delta}_t(l)$ provides an estimate of how much a link weight fluctuates. We define a change as significant when the weight of a link changes by more than twice the expected mean deviation. In other words, we associate the start of the event as the time when the current weight changes by more $2 * \bar{\delta}_t(l)$. At this point, we need to identify the end of the event when the link stabilizes again. We mark the end of the event by using a fixed timeout



(a) Instantaneous Link Weight



(b) Expected Weight

Fig. 3. Example showing choice of $\alpha = 0.1$

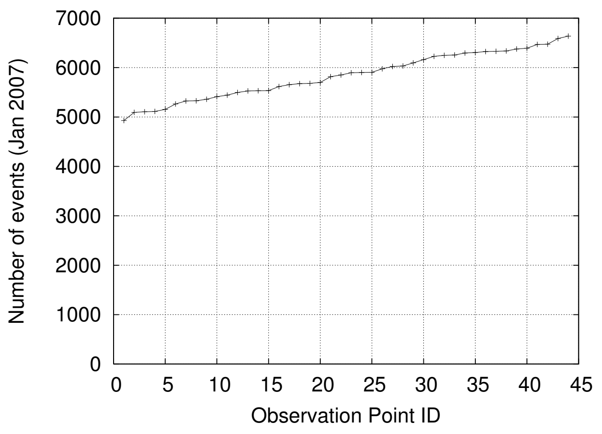


Fig. 4. Events in January 2007

of $t = 3$ minutes, being slightly more conservative than the average convergence time for events of around 2 minutes [12]. When viewing all links together, an event starts when any link changes by more than the deviation range, and ends $t = 3$ minutes later. At the end of the event, we have two weights for each link, the weight before and after the event.

Note it is possible that links with very small weights (e.g. $\bar{w}_t(l) = 5$) can result in a link event even when 1 route changes. To account for this, we log all link weight changes and filter out very small weight changes (e.g < 25 routes). Also due to our choice of fixed timeout of $t = 3$ minutes, it is possible for a single big event (e.g. 50,000 route changes) to be broken into more than one event. In Figure 4 shows the number of events identified by different observation points from RouteViews' Oregon collector for the month of January 2007.

Note that our scheme is very different from previous approaches on event detection that work with update clustering. Update messages can result from a variety of causes and not all of them affect data delivery. Our approach abstracts out the AS path changes using the aggregate measure of link weights. The reader must keep in mind that cases where one observes significantly large amount of updates being generated by a very small set of prefixes could be missed by our scheme. Further note that our current scheme defines aggregate in terms of a simple count of number of routes, and one might view a smaller number of important prefixes being affected as more significant than a larger set of affected prefixes seeing lesser traffic. Nevertheless, we feel that this scheme is well suited for identifying large scale routing changes.

IV. The Inference Scheme

We now present our heuristic to identify the origin of routing events. We first present an overview of our approach and then go into the details.

A. *Overview of approach*

After we apply the event identification scheme described in Section III, we get a set of routing events. Each event contains the time of the event, the set of links along with the weight of the link at the end of the event, and the change in weight. An event may be a *failure* event where a bunch of preferred routes are lost, or may be a *recovery* where previously lost preferred routes are available again. If the event is a failure then the origin of change is contained in the set of links that have lost routes, while if its a recovery event, then the origin of change is contained in the set of links that gained routes. At the first stage we do not make any classification of the event as a failure or recovery and identify origin of change for both possibilities.

For the set of edges that lost routes, we correlate the changes across different links involved and construct a flow graph called *fault graph* with an artificial source S and an artificial sink T. The idea behind constructing a fault graph is that any cut (set of edges) disconnecting S and T represents a possible explanation for *origin of change*. Further, if each edge were equally likely to fail, a cut involving the least number of edges, or min-cut is most likely to be the origin of the change. By augmenting the fault graph with information from other available observation points, we argue that the min-cut on the fault graph is the most likely explanation. We repeat

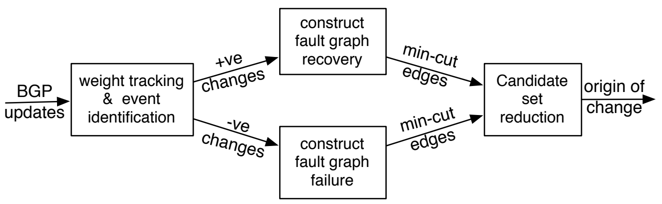


Fig. 5. Main steps in inference

the procedure of constructing a fault graph and finding the min-cut for the set of edges that gained routes as well. Finally, given two possible explanations, one with edges that lost routes and one with edges that gained routes, we use information about expected weights and variance to understand which explanation is more likely. Figure 5 shows the main steps in our process.

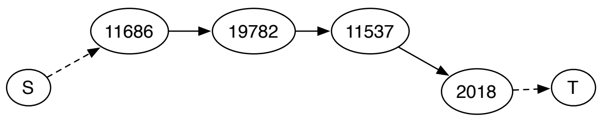
B. Fault graph

We now go into details of the construction of a fault graph. For each event, we construct two graphs, one is the loss graph involving all links that lost weight, and the other is the gain graph involving all links that gained weight. At this stage we do not know whether the event is a *failure* or a *recovery* and there is greater benefit in adding information from other monitoring points first. Algorithm 1 details the construction of a fault graph for positive and negative weight edges for each event.

Algorithm 1: Fault-Graph(E)

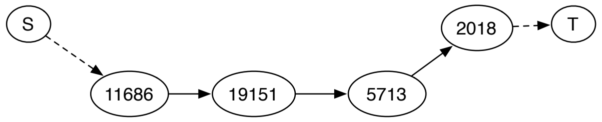
```
Construct $G_{st}$ as union of all negative (or positive) edges in $E$;
Assign $w(e) = 1$ for all $e \in E$;
/* Find nodes with no incoming changes */
for each node n such that $(x, n) \notin E$ for any $x$ do
    add edge $(s, n)$ to $G_{st}$ with $w(s, n) = \infty$ ;
/* Find nodes with no outgoing changes */
for each node n such that $(n, x) \notin E$ for any $x$ do
    add edge $(n, t)$ to $G_{st}$ with $w(n, t) = \infty$ ;
```

The main idea in constructing the fault graph is to connect the source node S to all the nodes that have only outgoing edges, and the sink node T to all the nodes that have only incoming changes. Figure 6 shows a fault graph constructed from a single observation point for an event that occured on March 9, 2007 at 18:05. One can see here that node 11537 is not connected to either S or T, since it has both incoming and outgoing edges. In reality, our implementation takes into account the total incoming weight change and the total outgoing weight change in making a decision on whether a node connects to S or T. For example, if some node B has total incoming change of -500 i.e. $w(A, B) = -500$ but an outgoing change of only -10, i.e. $w(B, C) = -10$, then the *sink* of the flow has to be at node B, due to the discrepancy in the incoming and outgoing changes. The enclosed table shows



Link	Weight	Expected	Change	Dev
11686-19782	12387	12342	-135	21
19782-11537	8495	8658	-135	23
11537-2018	0	137	-137	0

a. Negative weight change



Link	Weight	Expected	Change	Dev
11686-19151	16484	18346	135	3356
19151-5713	243	120	129	0
5713-2018	136	0	136	0

b. Positive weight change

Fig. 6. A fault graph from single observation point

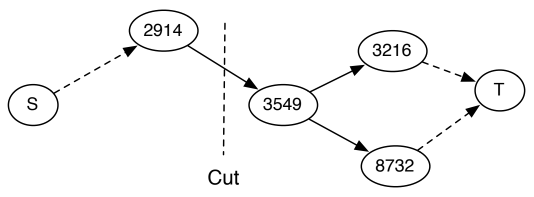
the link weights and changes for the three links included in the fault graph. In the fault graph, any of the three edges could be cut to obtain a possible explanation, but generally, as you go farther away from the source S, the edge weight decreases, and with comparable change on a single path, we try to remove an edge as far away from S as possible. In Figure 6, a min-cut on each graph results in two edges 11537-2018 and 5713-2018 as possible candidates

C. Augmenting fault graph with views from additional observation points

One may have access to events generated from other observation points in addition to their own BGP data. In practice, one can achieve this by processing events from a few peers of public data collectors like RouteViews and RIPE RIS. We are interested in adding information from other observation points to aid the inference from our primary observation point. Specifically, we build on the fault graph from the primary observation point in Algorithm 1.

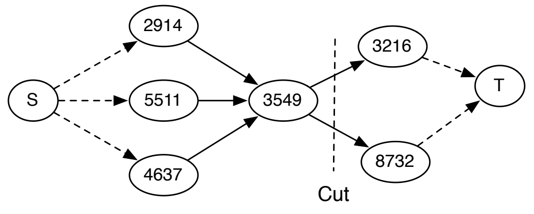
We first identify links from other observation points that are common to the fault graph from the primary point. We then identify the links that connect to and from these common links and add these set of links to the fault graph. Finally, we connect nodes that have only outgoing edges to S and nodes that have only incoming edges to T. Algorithm 2 presents the details.

Figure 7(a) shows the fault graph for an event observed from AS 2914. Here, the min-cut results in edge 2914-3549 as the possible origin of change. Now, adding information from 2 other monitoring points as in Algorithm 2 results in the fault graph in Figure 7(b), and one can see that the



Link	Weight	Change
2914-3549	5870	-172
3549-3216	0	-61
3549-8732	0	-85

(a) Min-cut on fault graph from 2914 identifies incorrect edge



(b) Adding information from other observation points to fault graph increases accuracy

Fig. 7. Augmenting a fault graph with additional information

Algorithm 2: Augmenting the fault graph

```
Input: $G_{st}$: The fault graph from primary observation
       point for time $t$
Output: $G'_{st}$: Augmented fault graph for time $t$
for each monitor $M_i$ do
    for event in time interval $(t - \delta, t + \delta)$ do
        for edge $(a, b)$ common to $G_{st}$ and $E_i$ do
            $E_i$=edges connecting to and from $(a, b)$;
            addToGraph($G_{st}, E_i$);
```

edges 3549-3216 and 3549-8732 are now identified as the origin of the change. Adding more observation points further strengthens this explanation. By observing the weights of the edges involved, it seems most likely that this is the correct explanation.

D. Candidate set reduction

In this final stage, we infer whether the change was a failure or a recovery based on the candidate sets we have and end up with either the positive change candidate set or the negative change candidate set as the most likely root cause.

1) Using link state information: We define the following states a link (l) can be based on its current weight $w(l)$, expected weight $\bar{w}(l)$ and deviation $\delta(l)$.

1) Normal: when $w(l)$ is within $\bar{w}(l) \pm 2 * \delta(l)$
2) Loss: when $w(l) < \bar{w}(l) - 2 * \delta(l)$
3) Gain: when $w(l) > \bar{w}(l) + 2 * \delta(l)$

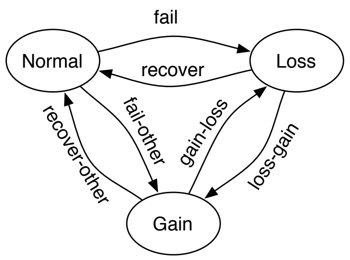


Fig. 8. State transition diagram

The state transitions between these states are shown in Figure 8. When a link fails, its weight drops and as a result it transitions from normal state to loss state, defined as *fail*. At the same time, the affected routes would take an alternate path and the weight for such a link on the alternate path will increase, thus moving it from normal state to gain state. We define this transition of state from normal to gain as a *fail-other*. This increase in rank however will not be due to its own recovery, but due to some other failure that results in routes using the link as an alternative. Similarly, transition from loss to normal and gain to normal can be classified as *recover* and *recover-other*.

Figure 8 also shows transitions between states of *loss* and *gain*. Such transitions (called *loss-gain* and *gain-loss*) are not frequent, but are difficult to classify and hence we do not associate them with recovery or failure. While we find a vast majority of the transitions to occur between normal and loss, and normal and gain states, a more detailed study needs to be done in understanding conflicting cases in order to make the best use of the knowledge of expected link weights. From the example in Figure 6, using the states defined in Figure 8, we classify the edge 11537-2018 as *fail*, and the edge 5713-2018 as *fail other*. Thus, the origin lies on the link 11537-2018, and actually the correct explanation was verified by logs obtained from AS 11537 confirming a BGP session failure with AS 2018 at that exact time.

E. Identifying node problems

Even though our scheme uses link weights to capture changes, we can also identify potential node problems where the origin is an AS instead of particular links. This can happen due to i-BGP problems within an AS causing instability or due to a BGP router connected to multiple ASes going down. To identify potential node problems, we first rank each node by the number of edges adjacent to it in the set of links from candidate set. The nodes with high rank (e.g. lots of adjacent edges in candidate set) are likely candidates for node problems. In our analysis, we found a few cases where a lot of links adjacent to a particular node were identified as origins of change.

V. Evaluation

In this section we discuss how we validate the results from our scheme. A fundamental challenge in the validation of a scheme to infer origin of changes is the dearth of publicly

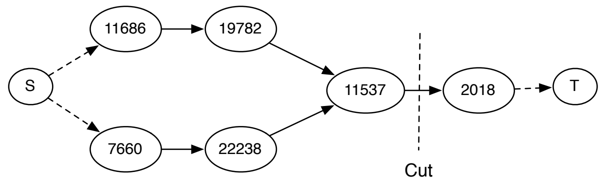


Fig. 9. BGP peer TENET (AS 2018) of Abilene (AS 11537) was unreachable. Event observed from primary view of AS 11686.

available documentation of BGP session or router failures. We perform two kinds of validation, one using publicly available information of outages from Abilene, and the other using BGP updates to identify a class of events where the origin of the event is almost surely adjacent to the AS announcing the prefix.

A. Validation using Abilene Data

The Abilene network in United States is a high-performance backbone network created by the Internet2 community. Abilene maintains a publicly accessible mailing list and archive containing descriptions of problems inside Abilene, outages, as well as BGP session problems with its peers. A typical *peer unavailable* message describes which BGP router inside Abilene lost connectivity to which AS, and the start and end time of the outage. However, most of Abilene's peering with an AS occurs through multiple physical locations, e.g. Abilene peers with KreoNet through Chicago as well as Seattle. Hence, when one of the BGP sessions, (say the Chicago peering with KreoNet) goes down, the affected AS (KreoNet) can still be reached using other internal BGP routers (KreoNet through Seattle). In such cases, Abilene may not announce an AS path change to its neighbors, and hence such events cannot be used for our validation. Using peering maps available from Abilene, we identified BGP peers of Abilene that connected at exactly one location and carried more than 25 prefixes, big enough to generate a link weight disturbance. If such a peering session were to go down, then Abilene would have to send an AS path change to its neighbors and this event could be seen at other observation points. We extracted events related to these peers over a 3 month period from January 1, 2007 to March 31, 2007. We found a total of 7 BGP peering session failures in this period from Abilene's email archives. Out of these events, 6 of them caused link weight disturbances observed at one or more observation points in our data, and all 6 of them were identified by our heuristic. Figure 6 on page 5 presents the fault graph from one of these 6 cases that was used for validation. Adding one additional view from AS 7660 gives the fault graph with weight losses shown in Figure 9. Our heuristic puts the min-cut at edge {11537,2018}. From the mailing list message, Abilene's peer TENET (AS 2018) lost connectivity to the Abilene network (AS 11537) at that time, as accurately identified by our scheme.

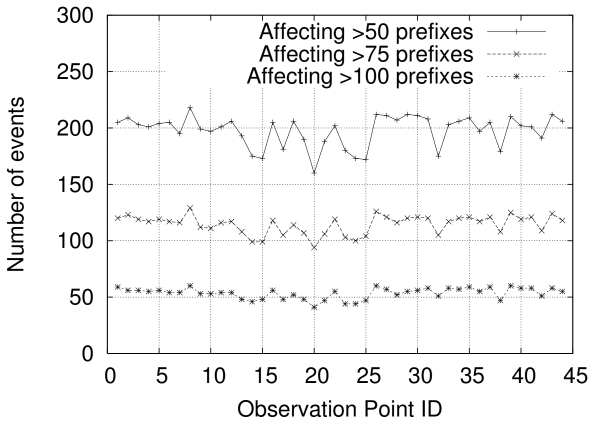


Fig. 10. Number of origin-adjacent events affecting each observation point

B. Validation using Origin-adjacent events

Prior works in inference of origin of change [4] validate results using BGP beacons. However, we do not use BGP beacons for validation since beacon events are per prefix and hence are not suited for our link weight based scheme which is tailored for larger scale disturbances. Instead, we identify large scale events where a set of prefixes originated by the same AS were unreachable as observed from a majority of observation points. Given the topological mesh-ness, the most likely reason for unreachability from diverse set of observation points is that the problem lies on or directly adjacent to the AS originating the prefixes. We call such events *origin-adjacent* events.

1) Collecting origin-adjacent events: To collect a set of origin-adjacent events, we used the clustering based event classification scheme [12] based on initial and final paths for each prefix. In particular, we were interested in the events classified by [12] as T_{down} events where a prefix is unreachable from multiple observation points. We extended this scheme and further correlated the T_{down} events across different prefixes using a fixed sliding window of 30 minutes. All events happening in the same time window affecting prefixes originated by the same origin AS were aggregated into a single event. We further removed those events that were affecting less than 50 prefixes, and from the remaining events, we only considered those that were observed by more than 50% of the monitors. We applied this classification scheme on the BGP data collected from RouteViews Oregon collector over the month of January 2007 to give us the set of origin-adjacent events. Note, that different observation points might observe slightly different events but any event must have been observed by at least 50% of the observation points. Figure 10 shows the number of events involving different minimum number of prefixes for each observation point [1].

2) Applying link weight based min-cut heuristic: To apply our scheme, we identified events based on our link weight

[1]We removed two observation points that saw a very small number of events.

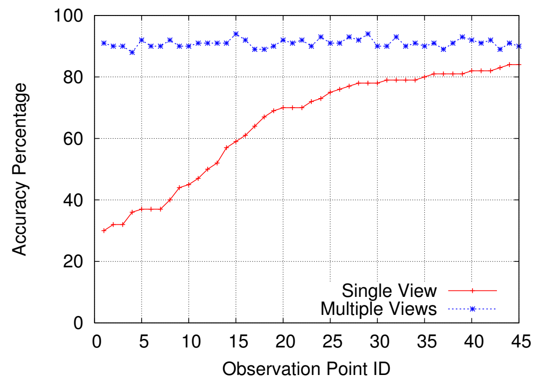


Fig. 11. Accuracy of F_{single} and F_{mult} for origin events involving more than 50 prefixes

based event identification scheme described in Section III. We then constructed a fault graph for each observation point individually using Algorithm 1. The min-cut on this fault graph, called F_{single}, indicated the likely origin of change taking into account only the primary observation point. We then augmented the fault graph with the view from the other observation points as in Algorithm 2. The min-cut on this fault graph, called F_{mult}, indicated the likely origin of change by augmenting information from other observation points to the primary observation point.

3) Results: Figure 11 shows the percentage of prefix origin-events-50 accurately identified by each observation point using its primary view only, i.e. F_{single}, as well as with information from other views, i.e. F_{mult}. The accuracy of event detection using just the primary view varies over the different observation points. However, we can see that the accuracy for all observation points is consistently above 90% for F_{mult} shown in Figure 11. To understand why accuracy for some monitors was low when using just the primary view, consider the example fault graph constructed in Figure 12. This fault graph was constructed as viewed from monitor AS 2493, and included withdrawn prefixes from AS 14117 and AS 306 around the same time. As can be seen, the first hop in the withdrawn path (i.e. 2493-3602) is common to these unrelated events, and this hop is returned by the min-cut scheme as the problem link, instead of the two links 6762-14117 and 575-306. However, when information from other monitors is added to this fault graph, other diverse paths force the cut on the last hop links, thus resulting in more accurate inference. Similarly high accuracy was also obtained with multiple monitors for prefix events involving more than 75 and 100 prefixes respectively. This shows that adding information from other observation points does help in increasing the accuracy of identifying the cause of the change.

C. Application to BGP data

We applied our scheme to BGP data over the month of January 2007. For each observation point, we used the additional views from the other points in diagnosing its own events. In particular, we analyzed the events seen by AS 2914

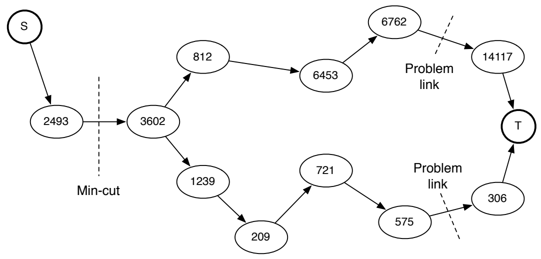


Fig. 12. Example showing why accuracy may be low with only primary view

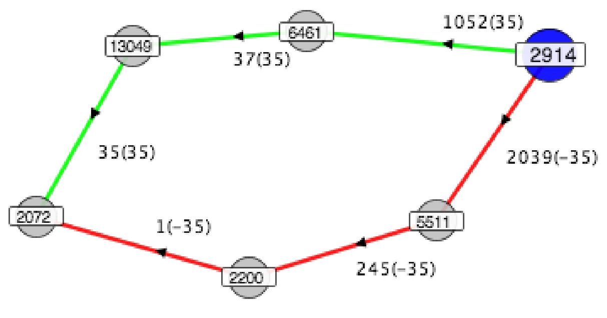


Fig. 13. Repeated instability involving AS 2072 as viewed from AS 2914.

in detail and present some results here. From the 5000 plus events seen by this monitor during this period, we observed that about 25% of the links are responsible for over 75% of the events. We examined the top 10 links most active links and found that most of the top links are links adjacent to AS 2914 or involving big ISPs. Interestingly, the top two most active links were 2200-2072 and 13049-2072. These links were two hops away from AS 2914, yet were responsible for over 400 events each. Figure 13 shows the Link-Rank representation for one such event involving these two links. In this particular case, as viewed from AS 2914, routes were lost along 2200-2072 and gained along 13049-2072. Over the one month period, many such events were observed where routes repeatedly switched between the two paths shown, indicating the existence of link instabilities over long periods of time. Clearly, cases like these can be avoided if detected immediately, instead of recurring over extended periods of time. Next we present another interesting large scale event we discovered using our scheme.

1) Case Study: The case study we present involved lots of routing changes from AS 2914 starting on Jan 31, 2007 around 7:00 GMT and lasting for about an hour. During this period, our heuristic identified a lot of links adjacent to AS 2914 as origins of observed routing change. To understand the event better, we present the Link-Rank graph summarizing the major links involved in Figure 14. As an example, the link between AS 2914 and AS 3257 (towards the top right) lost 2790 routes and had a final weight of 0. One can see from the Figure 14, that a lot of peers of AS 2914 lost all their routes. After discussing this event with a network operator at AS 2914, we found out that during that period, AS 2914 was

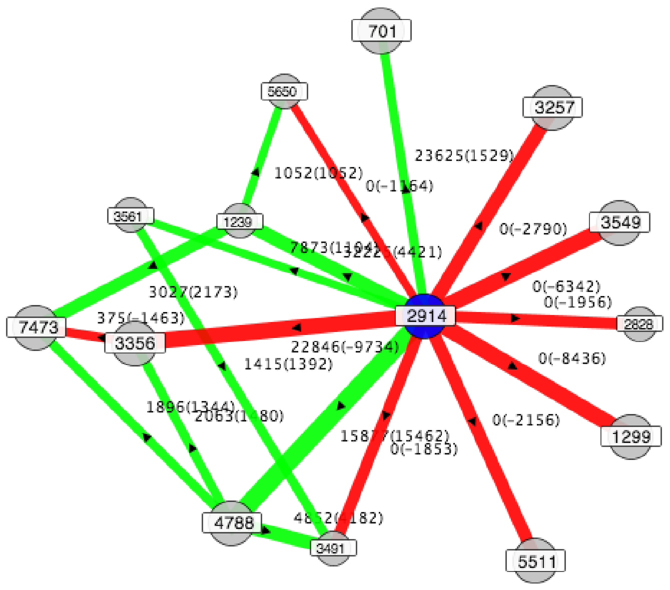


Fig. 14. Case study: Routing changes seen from AS 2914

providing temporary restoration for regional ISPs affected by the underwater cable cut due to the earthquakes off the coast of Taiwan. One of the downstream customers of AS 2914 in turn was providing temporary transit to a very large network and hence announced lots of prefixes to AS 2914. As a result, the max prefix filters of peers of AS 2914 were tripped causing the BGP peering sessions to go down. These session failures were accurately identified by our heuristic as origins of changes .

In this section we showed that our inference scheme can effectively identify the origin of large scale routing changes. We are currently working on applying our scheme to longer period of one year to get a better understanding of where problems occur repeatedly. In Section VI we look at our scheme at a higher level and discuss some potential limitations for inferring the origin of event.

VI. Discussion

In this section we examine our scheme at a higher level, discuss limitations of our inference technique and identify open issues that deserve further attention. Contrasting with the previous approach to origin inference by identifying the shared link or node among a large number of routes that failed at the same time, our link weight based approach does not require per-route information. Link weight changes reflect an aggregate measure of route changes, they not only bring an essential saving in the data processing but also help improve the inference accuracy in case of limited data, for example when only data from a single router is available. [2]

As the saying goes, every coin has two sides. While the use of link weights and weight changes makes the origin inference much faster, it can also introduce potential inference errors due to the lack of information about individual routes. One possibility is that some unrelated routing events may result in a link receiving both positive and negative weight changes simultaneously, and the combined results might lead to a negligible magnitude of changes, or even mask off the weight changes entirely. Another possibility concerns correlating observed link weight changes from different vantage points, there exists a non-zero probability that these changes could be due to simultaneous but different routing events, and again the lack of information about specific routes makes our scheme unable to tell whether that is, or is not, the case.

Nevertheless, our BGP routing measurement efforts over the last several years give us high confidence that the operational Internet exhibits strong system characteristics, which we can leverage to identify whether simultaneous routing events occurred. The expected weight and variance of the AS links represent such a system characteristic which we explored in our design. Given most routes are longer than one AS hop, the correlation of the weights and weight changes of *adjacent links* is another characteristic that we are yet to gain a full understanding, especially in cases where the affected links by different routing events partially overlap. As part of our future work, we expect to gain and utilize this understand to detect simultaneous routing events.

VII. Related work

Previous work in identifying source of BGP routing changes can be broadly sorted into three categories: automated analysis[7], [4], [3], [18], [2], [16], [19], [5], [17], visualization and human interaction based approaches [1], [9], and theoretical schemes in model settings[10].

In one of the seminal works about network instability, Labovitz *et. al.*[8] identified several causes of routing instabilities in the Internet, without however diagnosing their topological origin. Later efforts [3], [2], [4], [18] analyzed BGP updates and performed aggregation along three dimensions: time, monitors and prefixes, achieving a final output in the form of candidate sets of instability origins. Our *Mincut* scheme does a similar aggregation, except that we use link weight aggregates and focus on analyzing the BGP logs from the viewpoint of a specific monitor, augmented by some views of other monitors. [7] used a different approach, since it identified layers of links with shared risk and used membership information to isolate with accuracy failures in the optical hardware of a network backbone. Feldmann *et al.*[4] proposed a root cause inference system that aggregates BGP updates according to time, monitors and prefixes (by this order) and uses a greedy heuristic to identify the origin of change. Other class of works such as [16], [19], [5] diagnosed routing changes using anomaly detection techniques. Roughan *et. al.*[16] used an EWMA(exponential weighted moving average) technique in BGP data, and decomposition/Holt-Winters methods in SNMP data, showing that by increasing the number of monitors, the number of false alarms is also decreased. [19] used PCA (principal component analysis) techniques to correlate different updates into clusters, each cluster being

[2]Unfortunately, due to the unavailability of the data set used in [4], we are unable to quantitatively compare the processing saving and accuracy improvement of our scheme with the results presented in [4].

a set of prefixes or ASes which are affected by the same event. In similar flavor, Huang *et. al.*[5] used PCA to detect anomalies inside Abilene network using multiple Abilene BGP views and routers' configurations as input. Teixeira *et. al.*[17] describe a framework to detect the cause of a routing change using a coordinated diagnostic mechanism among several ISPs, requiring a special server in each ISP that replies to diagnose queries from other domains. In contrast, our scheme only requires the view from the own ISP and publicly available views from other ISPs that might be involved in the event.

On the visualization front, Link-Rank[9] is a visualization tool that can show routing changes in the form of links that lost and gained routes. Link-Rank graphs are ideally suited for illustration in analyzing the cause of the event, since one can see both the initial paths as well as the final paths. BGPlay[1] is another visualization tool that shows how routes from various monitors to a single prefix change. BGPlay uses animation to show path changes, and is very useful for single prefix routing diagnostics, where one can visually identify common points of convergence or routes.

Lad *et al.*[10] uses a simple model of Internet routing with shortest path policies and present a greedy algorithm to identify the root cause link. In their model, since each node knows the preferred route for every node in the network, and hence identifying the origin is easier in their case than that in actual BGP routing.

VIII. Conclusion

Due to the sheer size of the global routing infrastructure as well as its dense connectivity and the resulting complex interactions among the large number of interconnected networks, inferring the origin of global routing changes presents a great research challenge. In this paper we developed a novel inference algorithm for origin identification of routing changes. Our solution builds upon two previous results. The first one is the link weight model developed by the Link-Rank tool. The measure of link weight changes enables us to effectively infer the origin of routing changes using data collected from a single monitor, and to execute efficiently in a near real time manner. We also demonstrated that the accuracy of our scheme can be significantly improved by using data collected from a dozen or so monitors. The second one is the observation that, among multiple alternative paths to a given destination, the most preferred path is used most of the time. This observation translates directly to the corollary that each AS link should have a stable expected weight, and the deviations from this expected value would be due to transient routing changes. We verified this conjecture using BGP log data and used this concept of expected weight to detect significant routing events, and to distinguish routing changes caused by failures and recoveries, respectively.

Our results show that the use of link weights and changes can be a promising direction towards routing problem diagnosis in large scale networks. We plan to incorporate the implementation of our inference algorithm into the Link-Rank tool release and apply it to monitor BGP routing changes in real time.

Acknowledgements

The authors would like to thank The National Science Foundation(NSF) (contract No ANI-0221453) and Cisco URP for funding this work. The authors would especially like to thank Bruce Davie for his insightful comments on this work. The authors are grateful to the anonymous reviewers for their detailed comments on how to improve the paper. The authors would like to thank Yiyi Huang and Nick Feamster for sharing their experience with the Abilene data. Finally, the authors would like to thank Eric Osterweil, Jerry Cheng and Starsky Wong for their comments and suggestions.

References

[1] Giuseppe Di Battista, Federico Mariani, Maurizio Patrignani, and Maurizio Pizzonia. BGPlay: A system for visualizing the interdomain routing evolution. In *Graph Drawing*, volume 2912 of Lecture Notes Computer Science, pages 295–306, 2003.

[2] M. Caesar, L. Subramanian, and R. Katz. Root cause analysis of Internet routing dynamics. Technical Report UCB/CSD-04-1302, U.C. Berkeley, november 2003.

[3] D. Chang, R. Govindan, and J. Hiedemann. The temporal and topological characterestics of BGP path changes. In *ICNP*, november 2003.

[4] A. FeldMann, Olaf Maennel, Z. Morley Mao, A. Berger, and B. Maggs. Locating Internet routing instabilities. In *Proceedings of Sigcomm*, September 2004.

[5] Y. Huang, N. Feamster, A. Lakhina, and J. Xu. Detecting Network Disruptions with Network-Wide Analysis . In *Proc. of ACM SIGMETRICS*, 2007.

[6] Geoff Huston. 2005 – A BGP Year in Review. APNIC 21, March 2006.

[7] Ramana Rao Kompella, Jennifer Yates, Albert Greenberg, and Alex Snoeren. IP fault localization via risk modeling. In *Proceedings of Second ACM/USENIX Symposium on Networked Systems Design and Implementation*, 2005.

[8] C. Labovitz, G. R. Malan, and F. Jahanian. Origins of internet routing instability. In *Proceedings of the IEEE INFOCOM '99*, pages 218–26, New York, NY, 1999.

[9] Mohit Lad, Daniel Massey, and Lixia Zhang. Visualizing Internet routing changes. In *IEEE Transactions on visualization and Computer Graphics, special issue on visual analytics*, to appear, 2006.

[10] Mohit Lad, Akash Nanavati, Dan Massey, and Lixia Zhang. An Algorithmic Approach to Identifying Link Failures. In *10th Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2004.

[11] University of Oregon. RouteViews Routing Table Archive.

[12] Ricardo Oliviera, Beichuan Zhang, Dan Pei, Rafit Itzak-Ratzin, and Lixia Zhang. Quantifying path exploration in the Internet. In *Proceedings of Internet Measurement Conference, to appear*, 2006.

[13] Y. Rekhter and T. Li. A Border Gateway Protocol (BGP-4). *Request for Comment (RFC): 1771*, 1995.

[14] Jennifer Rexford, Jia Wang, Zhen Xiao, and Yin Zhang. BGP routing stability of popular destinations. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop 2002*, 2002.

[15] RIPE. Routing Information Service Project.

[16] Matthew Roughan, Timothy G. Griffin, Z. Morley Mao, Albert Greenberg, and Brian Freeman. Forwarding anamolies and improving their detection using multiple data sources. In *SIGCOMM Workshop: Network Troubleshooting*, 2004.

[17] Renata Teixeira and Jennifer Rexford. A measurement framework for pin-pointing routing changes. In *Proceedings of the ACM SIGCOMM workshop on Network troubleshooting*, 2004.

[18] Jian Wu, Z. Morley Mao, and Jennifer Rexford. Finding a needle in a haystack: Pinpointing significant BGP routing changes in an IP network. In *Proceedings of 2nd symposium on Networked Systems Design and Implementation (NSDI)*, 2005.

[19] Kuai Xu, Jaideep Chandrashekar, and Zhi-Li Zhang. A First Step Towards Understanding Inter-domain Routing. In *Proc. of ACM SIGCOMM Workshop on Mining Network Data*, 2005.