Detect and Analyze Large-scale BGP Events by Bi-clustering Update Visibility Matrix

Meng Chen[1], Mingwei Xu[1], Qing Li[2], Xirui Song[3] and Yuan Yang[1]

[1]Dept. of Computer Science and Technology, Tsinghua University, China
[2]Graduate School at Shenzhen, Tsinghua University, China
[3]Beijing University of Posts and Telecommunications, China
chenm11@mails.tsinghua.edu.cn xumw@tsinghua.edu.cn li.qing@sz.tsinghua.edu.cn
songxirui@bupt.edu.cn yyang@csnet1.cs.tsinghua.edu.cn

***Abstract*—Many attempts have been made to detect and analyze anomalous Internet events through dissecting BGP updates and tables, and substantial progress has been made in detecting and quantifying the impact of major Internet disruptions. However, we notice that most works in this realm either deploy/use a limited quantity of monitors or analyze aggregated statistics, and such practice may result in overestimating the impact of monitor-local events, which can be viewed only by a rather small portion of the Internet. To eliminate the impact of such local events on the detection of Internet-level anomalies, we raise the concept of Large-scale BGP Event (LBE), which affects a large amount of IP prefixes (high impact) and is widely observable (non-local). To detect LBE, we record update data in the Update Visibility Matrix (UVM) according to the prefix and monitor related to each update. At first, we formulate the problem of identifying LBE in UVM as a bi-clustering problem; after proving it is NP-hard, we describe our heuristic algorithm. Next, we apply our scheme to more than 2 TB of historical data. We find that LBE is highly correlated with many well-known disruptive incidents. Furthermore, we also identify some abnormal events that have never been investigated. We believe our work can assist in network operation tasks such as problem prevention, diagnosis, and recovery.**

I. Introduction

The de facto inter-domain routing protocol, Border Gateway Protocol (BGP), connects tens of thousands of Autonomous Systems (AS) that constitute the Internet. Route updating information is propagated by the means of BGP updates, and the content/statistics of updates imply rich information about the stability and healthiness of networks. Therefore, a wealth of measurement works on BGP dynamics have been proposed, many of which are dedicated to identifying abnormal Internet events, such as severe attacks, bad operations, and mis-configurations (e.g., [1]–[6]). These works play significant roles in network operation/management and problem diagnosis/recovery/dissection.

However, most prior efforts are prone to events local to specific route monitors (a route monitor is a BGP-speaking router that is configured to cooperate), mainly because they focus on the aggregated data from multiple monitors instead of investigating the data from each monitor separately. Examples of aggregated data are total update quantity [6], overall update patterns [1], and the feature traces extracted from all updates [5]. As a result, events local to only one/few monitors have the potential to strongly affect the analysis results, e.g., local events with limited impact being incorrectly interpreted as high-impact events.

Primarily due to the incremental characteristic of BGP, local events are prevalent in the Internet. Specifically, after the initial exchange of complete routing information, a pair of BGP routers exchange only the changes to that information; this characteristic largely restricts the propagation scale of updating information: it does not propagate far unless it changes the best routes at most of the BGP routers it traverses. In addition, route export policy also plays an important role in restraining this propagation scale. For example, a transit Internet Service Provider (ISP) usually does not export routes learnt from peers and providers to other peers and providers, hence it does not send the updates for these routes in these directions.

On the other hand, widely observable events do not necessarily mean high impact. For instance, individual new prefix announcement, or individual prefix withdrawal by its origin ASes. Note that these events affect a small number of prefixes.

In this paper, we define a *Large-scale BGP Event (LBE)* as follows: within a given time period, an LBE is a BGP event that a) impacts a large quantity of prefixes and b) can be observed by a large portion of (many) route monitors. Intuitively, an LBE indicates extraordinarily active changes in either the origin ASes of these prefixes or some critical transit ASes. Actually, we believe that an LBE implies anomalous even disruptive event(s) in the Internet. The reasons are a) The widely-observable characteristic suggests Internet-level inter-domain instability. b) The large numbers of monitors and prefixes imply an extremely high quantity of updates being propagated in the Internet, which is a potential threat to the processing capability of the routing facilities in the Internet. c) Due to the inter-domain instability, the performance and connectivity for the prefixes being updated may deteriorate, affecting world-wide attempts of trying to access these prefixes.

In this papar, we first raise the concept of *Update Visibility Matrix (UVM)*, a binary matrix in which data from every

978-1-4673-8590-9/15/$31.00 ©2015 IEEE

monitor and for every prefix are recorded separately; using UVM, we can effectively eliminate monitor-local events. Next, we formulate the problem of identifying LBE as a bi-clustering problem; after proving it is NP-hard, we propose an efficient heuristic algorithm to solve it.

Next, we apply the method to two sets of data: a) updates around nine famous disruptive incidents, and b) updates from Jan. to Oct., 2013. The total monitors exceed 400 and the total data add up to about 2.06 TB. The major observations are concluded as follows.

1) We notice a strong correlation between the well-known incidents and the identified LBEs: the quantity and sizes of the identified LBEs increase soon after the occurrence of the corresponding incidents.
2) LBEs do exist even during 'innocent period': we detect 101 LBEs in the ten months in 2013; that is one LBE per two to three days in average. we find that these LBEs are quite unevenly distributed.
3) The (prefix, monitor) pairs within an LBE involves averagely more updates than those outside the LBE, indicate higher instability.
4) Through clustering, we find that a single underlying event could cause multiple LBEs (as many as 18) widely scattered in time.
5) Through the case study of a large cluster, we reveal a persistent high-impact incident and attribute it to some configuration error in a national ISP.

II. UVM-BASED LBE IDENTIFICATION

In this paper, we denote scalars by lower-case letters (a), sets by upper-case letters (I), vectors by lower-case bold-face letters ($\mathbf{v}$), and matrices by upper-case bold-face letters ($\mathbf{X}$).

A. Problem Formulation

Definition 1. ***(Update Visibility Matrix)*** *Let P be the prefix set and M be the monitor set. Let $\mathbf{x}_{ij}$ be the element of the UVM $\mathbf{X}$; $\mathbf{x}_{ij} = 1$ if the jth monitor observes any BGP update for the ith prefix; otherwise $\mathbf{x}_{ij} = 0$. Let $I \subseteq P$ and $J \subseteq M$ be subsets of prefixes and monitors. The pair (I, J) specifies a submatrix $\mathbf{X}_{IJ}$ with the following attributes.*

The size of $\mathbf{X}_{IJ}$ is

$$Size(\mathbf{X}_{IJ}) = |I| \times |J| \tag{1}$$

The height and width of $\mathbf{X}_{IJ}$ are

$$Hei(\mathbf{X}_{IJ}) = |I| \tag{2}$$

$$Wid(\mathbf{X}_{IJ}) = |J| \tag{3}$$

The weight of $\mathbf{X}_{IJ}$ is the quantity of element '1':

$$Wei(\mathbf{X}_{IJ}) = \sum\nolimits_{i,j} x_{ij} \tag{4}$$

The density of $\mathbf{X}_{IJ}$ is the proportion of element '1':

$$Den(\mathbf{X}_{IJ}) = \frac{Wei(\mathbf{X}_{IJ})}{Size(\mathbf{X}_{IJ})} \tag{5}$$

For simplicity, we assume an UVM does not contain any empty (i.e., all '0') row or column. Note that the rows (and columns) of a submatrix are not sequenced hence are interchangeable. Also note that the attributes also apply to a single row (i.e., $|I| = 1$) or a single column (i.e., $|J| = 1$).

Definition 2. ***($(\theta_s, \theta_w, \theta_h, \theta_d)$-Event)*** *Given non-negative integers θ_s, θ_w, θ_h and a real θ_d ($0 \leq \theta_d \leq 1$), A $(\theta_s, \theta_w, \theta_h, \theta_d)$-Event $\mathbf{X}_{IJ}$ is a submatrix in the UVM $\mathbf{X}$ such that $Size(\mathbf{X}_{IJ}) \geq \theta_s$, $Wid(\mathbf{X}_{IJ}) \geq \theta_w$, $Hei(\mathbf{X}_{IJ}) \geq \theta_h$, and $Den(\mathbf{X}_{IJ}) \geq \theta_d$.*

For brevity, we denote $(\theta_s, \theta_w, \theta_h, \theta_d)$-Event as θ-Event. Given proper values of the thresholds θ_s, θ_w, θ_h, and θ_d, a θ-Event is considered as an LBE. In this paper we use the two terms interchangeably. The four thresholds restrict four aspects of an LBE. θ_s decides the minimum size an LBE should have. θ_w and θ_h prevent the detection of local or low-impact events. θ_d determines how much noise can be tolerated.

Definition 3. ***(The Large-scale BGP Event Identification Problem)*** *Given an UVM $\mathbf{X}$, and non-negative integers θ_s, θ_h, θ_w, and a real θ_d ($0 \leq \theta_d \leq 1$), find the submatrix $\mathbf{X}_{IJ}$ such that:*

i) $\mathbf{X}_{IJ}$ is a $(\theta_s, \theta_d, \theta_w, \theta_h)$-Event; and
ii) $size(\mathbf{X}_{IJ})$ is maximized.

Note that although it is possible that multiple $(\theta_s, \theta_d, \theta_w, \theta_h)$-Events are contained in $\mathbf{X}$, we identify only the largest one. This is because we do not assume a one-to-one mapping from an identified LBE to one underlying incidents. In other words, an LBE is the superposition of multiple underlying incidents, so we care more about the existence (and the properties) rather than the quantity of LBE within a period.

Theorem 1. *The Large-scale BGP Event Identification Problem is NP-hard.*

Proof: After setting the thresholds θ_s, θ_w, θ_h to 0, and θ_d to 1, we get a more-specific problem. Then, we expand $\mathbf{X}$ into a square matrix by adding 0s. Because $\theta_d = 1$ means a θ-Event contains only element '1', the identified θ-Events before and after the expansion are identical (even though $\mathbf{X}$ has become larger). Now the more-specific problem is equivalent to a more-general problem of the Maximum Clique Problem [7] (the original MCP requires a square identified submatrix). Since the MCP is NP-hard, our original problem is also NP-hard. ■

Actually, because of high complexity, it is difficult even to approximate the Maximum Clique Problem. But we want a quite efficient algorithm to solve our problem, considering that timely alarm of anomaly is critical for operation purposes. To this end, we devise the following heuristic algorithm.

B. The Heuristic Algorithm

The heuristic algorithm is shown in Algorithm 1; its basic idea is as follows. Given an UVM $\mathbf{X}$ with prefix set P (so $Hei(X) = |P|$) and monitor set M (so $Wid(X) = |M|$), and

Algorithm 1 The heuristic algorithm

Input: An UVM $\mathbf{X}$; the thresholds θ_h, θ_w, θ_s, and θ_d
1: $\mathbf{M} \leftarrow \mathbf{X}$, $den \leftarrow$ GET-DENSITY($\mathbf{M}$)
2: **repeat**
3: $\quad \mathbf{r}_a \leftarrow$ GET-CANDI-RA($\mathbf{X}, \mathbf{M}$), $\mathbf{c}_a \leftarrow$ GET-CANDI-CA($\mathbf{X}, \mathbf{M}$)
4: $\quad$ **if** not ADDITION($\mathbf{M}$,$\mathbf{r}_a$,$\mathbf{c}_a$) **then**
5: $\quad\quad \mathbf{r}_d \leftarrow$ GET-CANDI-RD($\mathbf{M}$,θ_h), $\mathbf{c}_d \leftarrow$ GET-CANDI-CD($\mathbf{M}$,θ_w)
6: $\quad\quad$ **if** $\mathbf{r}_d \neq NULL$ and $\mathbf{c}_d == NULL$ **then**
7: $\quad\quad\quad M \leftarrow M - \mathbf{r}_d$
8: $\quad\quad$ **else if** $\mathbf{c}_d \neq NULL$ and $\mathbf{r}_d == NULL$ **then**
9: $\quad\quad\quad M \leftarrow M - \mathbf{c}_d$
10: $\quad\quad$ **else if** $\mathbf{r}_d \neq NULL$ and $\mathbf{c}_d \neq NULL$ **then**
11: $\quad\quad\quad M \leftarrow M-$ LARGER-INCDEN-PER-DELSIZE($\mathbf{r}_d$, $\mathbf{c}_d$)
12: $\quad\quad$ **else**
13: $\quad\quad\quad$ Return $NULL$
14: $\quad den \leftarrow$ GET-DENSITY($\mathbf{M}$)
15: **until** $den \geq \theta_d$
16:
17: $\mathbf{r}_a \leftarrow$ GET-CANDI-RA($\mathbf{X}, \mathbf{M}$), $\mathbf{c}_a \leftarrow$ GET-CANDI-CA($\mathbf{X}, \mathbf{M}$)
18: **while** $\mathbf{r}_a \neq NULL$ and $\mathbf{c}_a \neq NULL$ **do**
19: $\quad$ ADDITION($\mathbf{M}$,$\mathbf{r}_a$,$\mathbf{c}_a$)
20: $\quad \mathbf{r}_a \leftarrow$ GET-CANDI-RA($\mathbf{X}, \mathbf{M}$), $\mathbf{c}_a \leftarrow$ GET-CANDI-CA($\mathbf{X}, \mathbf{M}$)
21:
22: $den \leftarrow$ GET-DENSITY($\mathbf{M}$)
23: **while** $den \geq \theta_d$ **do**
24: $\quad \mathbf{r}_a \leftarrow$ GET-CANDI-RA($\mathbf{X}, \mathbf{M}$), $\mathbf{c}_a \leftarrow$ GET-CANDI-CA($\mathbf{X}, \mathbf{M}$)
25: $\quad M \leftarrow M+$ LARGER-INCSIZE-PER-DECDEN($\mathbf{r}_a$, $\mathbf{c}_a$)
26: $\quad den \leftarrow$ GET-DENSITY($\mathbf{M}$)
27:
28: **if** $Size(\mathbf{M}) \geq \theta_s$ **then**
29: $\quad$ Return $Size(\mathbf{M})$
30: **else**
31: $\quad$ Return $NULL$

Algorithm 2 Function: ADDITION($\mathbf{M}$,$\mathbf{r}_a$,$\mathbf{c}_a$)

Input: Current submatrix $\mathbf{M}$; Candidate row $\mathbf{r}_a$ and column $\mathbf{c}_a$ for addition
1: **if** $\mathbf{r}_a \neq NULL$ and $\mathbf{c}_a == NULL$ **then**
2: $\quad M \leftarrow M + \mathbf{r}_a$
3: **else if** $\mathbf{c}_a \neq NULL$ and $\mathbf{r}_a == NULL$ **then**
4: $\quad M \leftarrow M + \mathbf{c}_a$
5: **else if** $\mathbf{r}_a \neq NULL$ and $\mathbf{c}_a \neq NULL$ **then**
6: $\quad M \leftarrow M+$ LARGER-WEIGHT($\mathbf{r}_a$, $\mathbf{c}_a$)
7: **else**
8: $\quad$ Return False ▷ No addition is conducted
9: Return True

the parameters θ_h, θ_w, θ_s, and θ_d, we conduct multiple rounds of row/column deletion/addition actions to get a submatrix $\mathbf{X}_{IJ}$ with a density greater than θ_d. If $Size(\mathbf{X}_{IJ}) \geq \theta_s$, we record the LBE and its size; otherwise no detection is recorded.

1) Main iteration: the main iteration is from line 2 to 15. In each iteration, the action is either row/column addition or row/column deletion. Since our major target is to increase the density of the remaining submatrix, deletion is usually much more frequent than addition. However, whenever addition does not decrease density, we prefer addition to deletion because the optimization objective of the original problem is maximized size. The details of the addition/deletion actions are described below.

2) Addition: among all the rows $\mathbf{x}_{kJ}$ ($k \in K, K = P \setminus I$) outside the submatrix $\mathbf{X}_{IJ}$, $Den(\mathbf{x}_{\mu J})$ is the maximum; if $Den(\mathbf{x}_{\mu J}) \geq Den(\mathbf{X}_{IJ})$, $\mathbf{x}_{\mu J}$ is the candidate addition row $\mathbf{r}_a$. Using similar method, we can get the candidate addition column $\mathbf{c}_a$ if it exits. These tasks are accomplished by functions GET-CANDI-RA() and GET-CANDI-CA() (line 3). They return $NULL$ if no candidate is found. If both $\mathbf{r}_a$ and $\mathbf{c}_a$ exist, we add to $\mathbf{X}_{IJ}$ whichever has greater weight, accomplished by function LARGER-WEIGHT() in ADDITION() (The pseudo-code is in Algorithm 2). We use the metric weight here because we want a) the size of the submatrix after addition is maximized, and b) the density of the submatrix after addition is maximized (so that less deletion is conducted in future). Weight is the product of the two metrics.

3) Deletion: if no addition is conducted (ADDITION() returns False), we conduct deletion. We first identify the candidate deletion row and column ($\mathbf{r}_d$ and $\mathbf{c}_d$) by functions GET-CANDI-RD() and GET-CANDI-CD() respectively (line 5). The scheme is simple: among all the rows $\mathbf{x}_{iJ}$ ($i \in I$) in the submatrix $\mathbf{X}_{IJ}$, $Den(\mathbf{x}_{\gamma J})$ is the minimum; if $Den(\mathbf{x}_{\gamma J}) < Den(\mathbf{X}_{IJ})$, $\mathbf{x}_{\gamma J}$ is the $\mathbf{r}_d$. Using the same method, we can get $\mathbf{c}_d$ if it exits. If both $\mathbf{r}_d$ and $\mathbf{c}_d$ exist, we delete whichever makes the increased density per deleted size maximized; this is accomplished by function LARGER-INCDEN-PER-DELSIZE() (line 11).

4) Addition in the end: the main iteration ends when $Den(\mathbf{X}_{IJ}) \geq \theta_d$. Straight after that, we conduct two rounds of additions by different methods to maximize $Size(\mathbf{X}_{IJ})$. a) Round one (line 17 to 20): if further addition does not decrease $Den(\mathbf{X}_{IJ})$, we apply the same addition scheme as that within the main iteration. b) Round two (line 22 to 26): from now on we allow $Den(\mathbf{X}_{IJ})$ to decrease, as long as $Den(\mathbf{X}_{IJ}) \geq \theta_d$. The major difference with round one is that if both candidates exist, we select the candidate that maximize the increased size per decreased density, accomplished by function LARGER-INCSIZE-PER-DECDEN() (line 25).

5) No-action conditions: note that an action cannot be conducted if either of the conditions occur: a) no candidate can be found; b) further deletion violates the width or height threshold (note the arguments of GET-CANDI-RD() and GET-CANDI-CD()). If both addition and deletion are prohibited, the algorithm terminates (line 13).

6) Complexity: the quantity of row and column manipulations are $O(|P|)$ and $O(|M|)$ respectively. We assume that the number of prefix is much larger than that of monitor. Within each iteration, it takes $O(|M|)$ time (i.e., simply scanning the columns) to get the candidate addition and deletion columns. By deploying a 'value to index mapping' data structure, it also takes $O(|M|)$ time to get the row candidates because the rows in and out of the current submatrix have $O(|M|)$ different weight values. So the total time complexity of the algorithm is $O(|M|(|P| + |M|))$.

III. MEASUREMENT SETUP

A. Data Collection and Preprocessing

We obtain BGP updates from RouteViews [8] and RIPE RIS [9], both of which operate multiple route collectors that have BGP sessions with BGP speaking routers (work as monitors) in the Internet. We use 17 collectors that provide data from 452 monitors as of January 2013. The number of available monitors is not constant at different time points because firstly, collectors/monitors start working at different time points, and secondly, sometimes collectors/monitors may stop working temporarily or permanently.

1) Monitor winnowing: some of the 452 monitors are unsuitable for our experiment: a) If a monitor has only a partial view of the Internet, it may miss some Internet-level BGP events; so we select only the monitors with global view. To this end, we set a benchmark as the quantity of prefixes in the routing table of a core router [10], and a global-view monitor should see at least 90% of the benchmark. This operation removes 269 monitors. b) Some BGP routers have BGP sessions with more than one collectors, hence are interpreted as multiple monitors. As a result, the impact observed by such a router may be incorrectly amplified. To deal with it, we select only one among the sessions with the same router. This step removes 52 monitors. c) Some ASes own multiple monitors, which may lead to events local to these ASes being overstated. So we select only one monitor in each AS. This step further removes 8 monitors. In summary, among the 452 monitors, 123 are suitable for our measurement.

2) Distribution of the monitor ASes: we get the tiers of the monitors' ASes according to [11], which uses the size of customer cone as the metric. An AS's customer cone is the ASes that can be reached through AS-level provider-to-customer links, and the size of customer cone indicates the influence of the AS. The method is: the ASes with less than 5 downstream customers are stubs; the ASes with between 5 and 50 downstream customers are small ISPs (tier-3); the remaining non-tier-1 ASes are large ISPs (tier-2); the list of tier-1 ISPs are directly obtained from the website [12]. We get customer cone data from a public repository [13], maintained by RIPE RIS. The result for the 123 monitors is: tier-1: 8, tier-2: 37, tier-3: 43, stub: 35. Next, we map the ASes to countries/regions by using the geographic data from the site [14]. For tier-1 ASes, we record their region as 'Global'. We find that the 123 monitors are spread across 25 countries/regions in the world. To summary, the monitors are widely spread in the Internet; so we believe they are capable of detecting Internet-scale events.

3) Eliminating the effect of BGP session resets: due to connection outage by either intended manipulations or unintended faults, occasionally route collectors need to re-build BGP sessions with some route monitors. In the process, the complete BGP tables of these monitors are sent to the collector in the form of updates, which occupy notable computing and storage resources but provide little information about global Internet status. Therefore, we delete these updates by applying the method in [15].

B. Parameter Settings

Since the quantities of monitors and total routable prefixes change with time, we set θ_w and θ_s according to the quantity of monitors $|M|$ and that of total prefix at the time of measurement, i.e., $|P_t|$ (not $|P|$, which is just the number of prefixes in UVM). Specifically, we set θ_w to a ratio of $|M|$, and θ_s to a ratio of $|M| \times |P_t|$. For brevity, we directly use the ratio values to represent width and size, and call them *relative width* and *relative size* respectively.

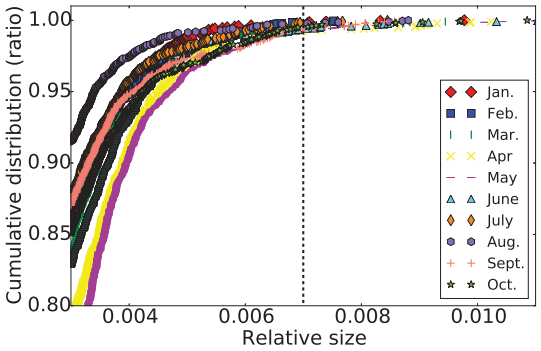


Fig. 1: Cumulative Distribution of LBE size for the 10 months when setting θ_s to 0.

In order to set θ_s, we plot the cumulative distribution of the sizes of identified LBEs when setting θ_s to 0, shown in Fig. 1. We assume the Internet is stable and healthy most of the time, so θ_s should be large enough in order that all the months contain small quantities of LBEs. That is to say, we are interested in capturing LBEs in the tails of the curves, and θ_s accommodates all the distribution curves. Fig. 1 illustrates that setting θ_s to 0.007 can catch about the top 0.5% 'LBE's in a month: the vertical line at size 0.007 intersects the curves at ratios between 99.2% to 99.8%. We set θ_s to 0.007 in this paper.

We set θ_w to 40% (i.e., $\theta_w = 40\% \times |M|$). This value effectively avoids that all the monitors in an LBE are of the same tier or in the same country/region. Our further analysis shows that the LBE identification result is not sensitive to this threshold, i.e., the vast majority of LBEs have quite large width.

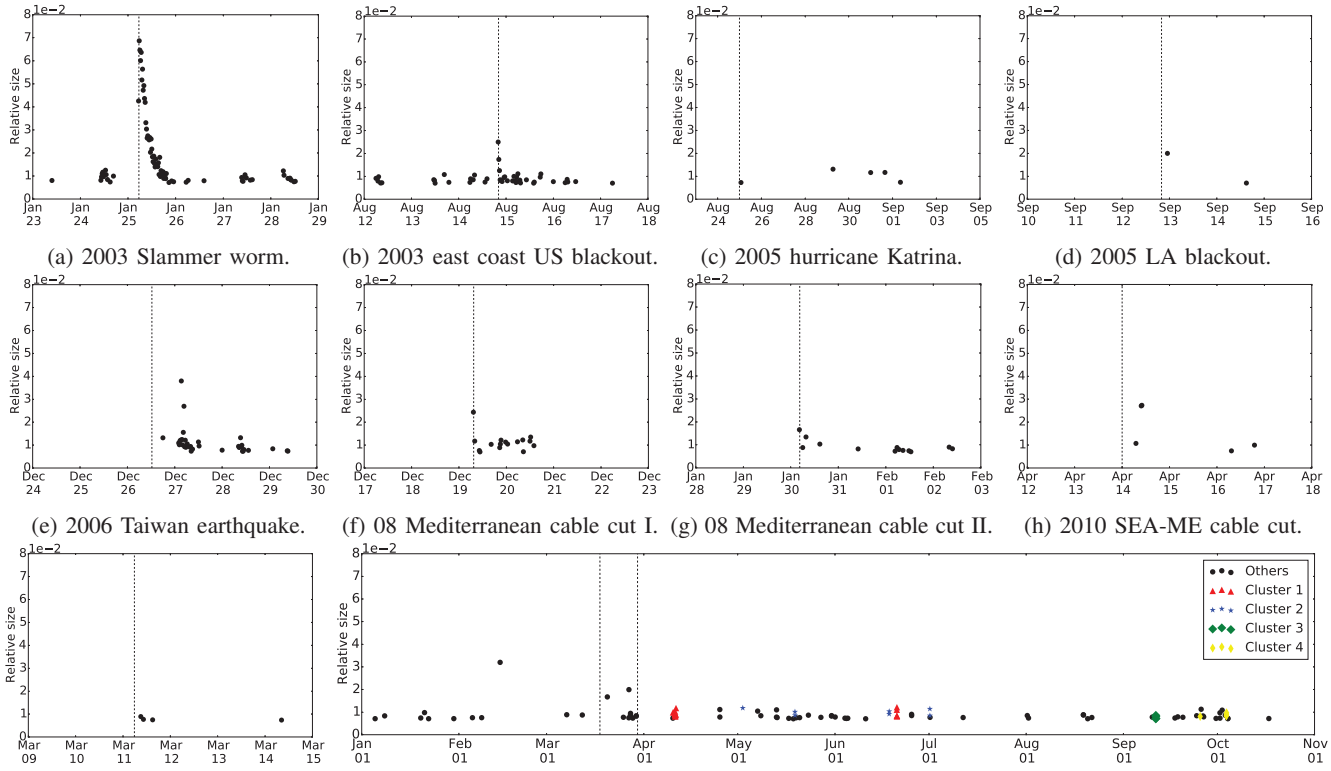
In terms of θ_h, since the quantity of prefixes is much larger than that of monitors in our experiment, we omit the height threshold; the height of LBE $\mathbf{X}_{IJ}$ is at least $\frac{\theta_s}{Wid(\mathbf{X}_{IJ})}$.

As for the density threshold θ_d, it should not be 100% or very close to that. On the other hand, it cannot be too small otherwise the correlation between updates inside an LBE is too weak. We set θ_d to 0.8 and show the effect of varying this threshold in Section V-C.

We set the length of time slot to 20 minutes basing on the following considerations. First, we seek to decrease the possibility that the impact of one underlying incident is divided into separate slots; this requires wide slot. Second, we seek to decrease the number of concurrent underlying incidents within a single slot; this requires short slot. Previous research has established that it usually takes several minutes or less time for a re-routing event to converge [16], and quite rarely could the convergence time be longer than 10 minutes [16], or one hour [17]. So we believe 20 minutes is a reasonable choice. We analyze the impact of this parameter in Section V-C.

IV. Results for Famous Internet Incidents

The identified LBEs for nine famous disruptive Internet incidents are demonstrated in Fig. 2 (a to i), including worm (Fig. 2a), blackouts (Fig. 2b, 2d), natural disasters (Fig. 2c, 2e,



(a) 2003 Slammer worm. (b) 2003 east coast US blackout. (c) 2005 hurricane Katrina. (d) 2005 LA blackout.

(e) 2006 Taiwan earthquake. (f) 08 Mediterranean cable cut I. (g) 08 Mediterranean cable cut II. (h) 2010 SEA-ME cable cut.

(i) 2011 Japan Tsunami. (j) Ten months in 2013. The dash lines mark the start and end of the Spamhaus DDoS attack.

Fig. 2: Identified LBEs when $\theta_d = 0.8$, $\theta_w = 40\%$, and $\theta_s = 0.7\%$. The dash lines mark the occurrence of the famous incidents. It is visible that both the quantity and sizes of the LBEs increase shortly after the disruptive incidents occur.

2i), and cable cuts (Fig. 2f, 2g, 2h). The x-axis denotes date and the y-axis denotes relative size. The vertical dash lines in the figures mark the occurrence of the underlying incidents. Note that although θ_s is obtained by analyzing the 2013 data, for simplicity, we directly apply it here instead of obtaining a θ_s for every year. We elaborate on its impact later.

Among these incidents, seven (excluding Fig. 2i and 2h) were investigated in [1], where the visible impact of these incidents on the inter-domain routing system is demonstrated. Note that we do not analyze the incidents earlier than 2003, because the number of available monitors is too small. As regard to the other two events, the impact of the 2011 Japan Tsunami is illustrated in [18], and the 2010 SEA-ME cable cut is investigated in [19].

As is evident from the figures, the quantity and sizes of the LBEs increase immediately after the occurrence of the incidents, indicating high correlation. Note that the delay of the LBEs in Fig. 2e and 2h is longer than that of the other figures. For the former, there were multiple earthquakes within several hours and we speculate the first strike was relatively weak and did not heavily impair the Internet. For the latter, the marked occurrence time is not accurate because we know only that the incident happened on April 14; but the exact time was never declared. So the exceptions do not violate our basic observation, i.e., a strong correlation between the number and sizes of LBEs and the famous disruptive incidents. These results highlight the effectiveness of our method and basic idea.

Except for Fig. 2a and 2b, no LBE exists before the occurrence of the incident. We believe the exceptions are due to the use of the θ_s learnt from the data in 2013, which is probably too small for early years thus lead to more LBEs being detected in 2013. Note that the sizes of these too-early LBEs are just slightly larger than θ_s, indicating that a slightly larger θ_s can easily eliminate them. On the other hand, compared with the too-early LBEs, the size increase of the other LBEs is significant, which validates the power of our method.

V. Results for the Ten Months in 2013

The results are shown in Fig. 2j: we detect 101 LBEs in the 10 months, i.e., 2.33 LBEs for each week in average. So generally speaking, LBEs are rare in the Internet, which complies with the common belief that the Internet is stable most of the time. Note that the distribution of the LBEs is quite uneven; for example, while the LBEs in late-March, mid-April, mid-May, and late-September are large in quantity, there are few LBEs in February and July.

We believe the LBEs in late-March can be explained by a well-known incident, i.e., a large-scale DDoS (Distributed Denial of Service) attack targeting the Spamhaus site, which lasted for more than 10 days and reached record-breaking flow

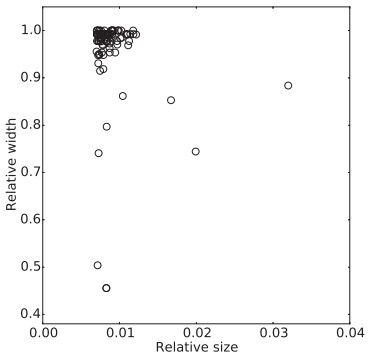


Fig. 3: Relative width as a function of relative size.

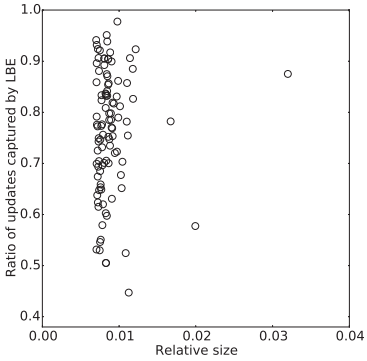


Fig. 4: The ratio of updates in a time slot captured by the LBE in the slot.

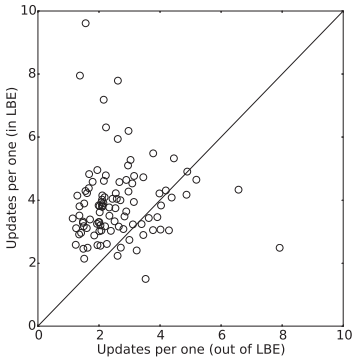


Fig. 5: The average number of updates captured by each '1' in and out of an LBE.

rate. The incident is marked by the two dash lines. We cannot map the other LBEs to well-known incidents.

A. Features of the LBEs

The widths of the LBEs as a function of their sizes are plotted in Fig. 3. The vast majority of LBEs have relative sizes between 0.007 and 0.015; among the three LBEs larger than 0.015, two are during the DDoS attack, implying the severity of the incident. On the other hand, most of the LBEs have rather large widths, significantly larger than θ_w. The result suggests that while LBE identification is not sensitive to the width threshold, it is quite sensitive to the size threshold.

Fig. 4 demonstrates the ratio of updates contained in an LBE as a function of the size of the LBE. We observe that all but one of the LBEs capture $> 50\%$ of the updates existed in the corresponding slot, and 72.3% of the LBEs capture $> 70\%$ of updates. The result indicates that an LBE is largely responsible for the high dynamics in the corresponding 'active period'.

In Fig. 5, we present the average number of updates each '1' in an LBE contains as a function of that number out of the LBE. It shows the ratio can be as small as 0.314, and as large as 7.962. Although the range of the ratio is wide, most LBEs lie in the 'large side', i.e., 83 LBEs are to the left of the line that crosses the figure. This observation indicates that instable (prefix, monitor) pairs, which involve more updates than other pairs, tend to be captured by LBE.

B. Clustering the LBEs

In this part, we cluster the LBEs that are probably caused by the same reason, and the clustering is mainly according to the prefix sets of the LBEs. We do not consider the monitor sets because a) for an LBE, its monitor set is magnitudes smaller than its prefix set; b) prefix set provides more information about the cause of the LBE, which we describe in detail in Section VI.

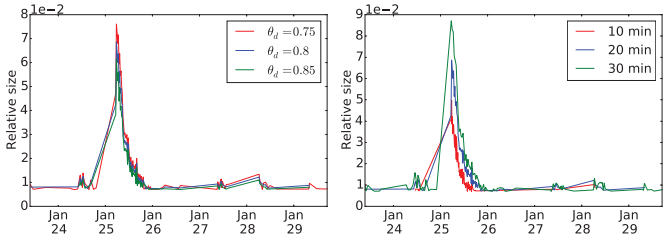
We adopt the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) technique [20], and we use the Jaccard distance as the distance metric. For two LBEs $\mathbf{A}_{I_1J_1}$, $\mathbf{B}_{I_2J_2}$, the distance between them is the Jaccard distance between their prefix sets:

$$Dist(\mathbf{A}_{I_1J_1}, \mathbf{B}_{I_2J_2}) = 1 - \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|} \tag{6}$$

For example, the Jaccard distance between two LBEs with identical prefix sets is zero, and two LBEs with completely different prefix sets have a Jaccard distance of one.

The DBSCAN technique requires two parameters, a) ε: the maximum distance to find directly reachable point, b) minPts: the minimum number of points to form a dense region. To best decrease the possibility of false positive, we set the parameters to strict values, i.e., $\varepsilon = 0.65$ and minPts $= 4$. The settings make the LBEs in a cluster highly related, the size of a cluster is at least four. As shown in Fig. 2j, LBEs belonging to different clusters are marked with different colors; note that black marks unclustered LBEs. There are four clusters, containing 18 (red), 7 (blue), 4 (green), and 5 (yellow) LBEs respectively. While some clusters persist for a short time, some span months. To understand the reason for the clusters, we conduct a case study of the largest cluster in Section VI.

C. The Impact of the Parameters



(a) Impact of θ_d. (b) Impact of length of time slot.

Fig. 6: Impact of parameters on the analysis of the Slammer worm incident. Unless otherwise specified, $\theta_d = 0.8$, $\theta_w = 40\%$, $\theta_s = 0.7\%$, and length of time slot is 20 minutes. It is visible that the trend persists with moderate varieties.

To demonstrate how varying the value of the parameters affects LBE identification, we conduct a case study of the Slammer worm incident, because it generates a large amount

of LBEs and it is easy to observe the change in trend. The impact of θ_d and the length of time slot is shown in Fig. 6, where we demonstrate the trend of the identified LBEs in accordance to different parameter settings. It is evident in the figures that minor changes to θ_d and the length of time slot do not affect the size trend of identified LBEs. In other words, generally speaking, despite changes in the parameter values, large LBEs remain large, small LBEs remain small or become undetected. To summarize, we believe that while moderate modification to the parameters affect the quantity and sizes of identified LBEs, it does not impair the correctness of our scheme.

VI. Case Study: the Largest Cluster

In this section, we analyze the largest cluster in the previous section, as the first step towards a systematic method to infer the major cause of LBE. The cluster contains 18 LBEs, among which 13 are in April and 5 are in June.

A. Analyze Common Prefix Set

We believe the prefix set of an LBE is the key to inferring the topological location of the major cause. Intuitively, if the prefixes in the set are largely uniformly owned by quite a lot of ASes, the major cause is likely in or near the core of the Internet. Other the other hand, if most of these prefixes belong to only one or few ASes, the major cause is likely in the periphery of the Internet, close to the AS(es). Based on this consideration, we obtain the common prefix set of the 18 LBEs; this set contains 1328 prefixes and they are our main clue to the topological location of the major cause.

For each one of the 18 LBEs, we extract the origin ASes of the 1328 prefixes from the corresponding updates. The top 3 most popular origin ASes for the LBEs are listed in TABLE 1. The result is abnormal: the origin ASes are inconsistent; instead, they change with time. Actually, the change frequency demonstrated in the table is underestimated, because a prefix can be attributed to different ASes within a single LBE; for simplicity, we record only the latest attribution. Moreover, some changes may not be captured and recorded in the LBEs.

ID	Top 3 origin ASes	ID	Top 3 origin ASes
1	4847:**833**,6629:**414**,6174:**60**	10	4847:**664**,47331:**632**,9304:**24**
2	6629:**1304**,47331:**16**,6174:**3**	11	1273:**880**,6629:**441**,6174:**3**
3	1273:**1302**,47331:**19**,6174:**2**	12	9304:**523**,6629:**409**,12654:**390**
4	4847:**1251**,9304:**43**,19406:**26**	13	47331:**1024**,4847:**297**,6174:**2**
5	4847:**1180**,6629:**137**,9304:**4**	14	47331:**1290**,9121:**26**,1273:**2**
6	1273:**731**,4847:**377**,6629:**214**	15	6629:**1316**,47331:**3**,4847:**2**
7	4847:**1310**,9304:**6**,1273:**5**	16	4847:**1150**,6629:**165**,47331:**3**
8	47331:**1289**,9121:**26**,9304:**2**	17	9304:**772**,6629:**498**,47331:**29**
9	1273:**997**,9304:**236**,47331:**56**	18	19406:**1202**,6174:**72**,6629:**28**

TABLE 1: Top 3 most popular origin ASes and the number (bold) of their originated prefixes.

To identify the true origin AS of the 1328 prefixes, we turn to the routing tables before and after the LBEs to avoid the impact of the underlying event. For the BGP table in March, through longest prefix matching, we attribute 1309 (98.6%) of the prefixes to 141 less-specific prefixes, all of which belong to AS 9121 (a national ISP). The 1309 prefixes themselves do not exist in the routing table. The result for July is similar, the numbers are 1305 (98.3%) and 140 respectively.

Note that by now we cannot ensure that AS 9121 originated the prefixes; the major cause could also be an AS along the paths from AS 9121 to the monitors. However, other than AS 9121, we cannot identify such an AS, which is supposed to exist in most AS paths from the monitors to these prefixes. In fact, AS 9121 is the only one that exists in almost all the AS paths in the updates for the 1328 prefixes, with less than 30 exceptions. The analysis suggests that AS 9121 initialized more than 98% of the 1328 prefixes, so AS 9121 is the 'major player' of the 18 LBEs. We speculate the frequent changes of origin ASes are due to some abnormal configurations by AS 9121. To validate our speculation, we investigate the type of the updates, described as follows.

B. Analyze Update Pattern

We categorize update pattern into 6 types according to [21], listed in TABLE 2, and the results for the 18 LBEs are shown in Fig. 7 and Fig. 8. While the former compares the ratio of raw number, the latter records the ratio of '1's (in LBE) that contain certain pattern types.

Pattern	Description
WW	Two successive withdrawals.
WADup	Withdrawal followed by a duplicated announcement.
WADiff	Withdrawal followed by a different announcement.
AADiff	Two announcements with different AS paths or next hops.
AADup1	Two announcements with exactly the same attibutes.
AADup2	Different announcements with identical AS path and next hop.

TABLE 2: Update pattern terms and description.

As shown in Fig. 7, AADiff is the most (53.99%˜70.19%), followed by AADup2 (22.86%˜42.90%); other types take quite small ratios. Compared with the ratio of AADup2 (15%) in [21], the ratio in our measurement is much higher. Since this pattern indicates policy change (more detailed analysis shows they are mostly changes to BGP communities), we believe AS 9121 made frequent policy changes. AADiff are mainly due to two factors: a) frequent changes in the AS paths (mostly changes to origin AS); b) the BGP convergence process.

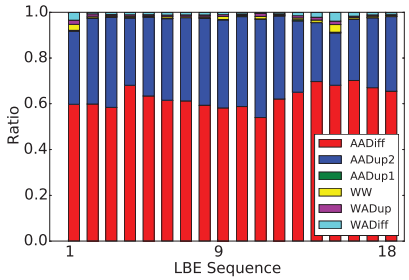


Fig. 7: Ratio of each update pattern by raw number.

Fig. 8 illustrates a notable decrease of AADiff and AADup2 patterns for the LBEs in June, implying that the activity of AS 9121 becomes weaker in June. For the '1's where no AADiff or AADup2 exists, the patterns are mostly individual updates,

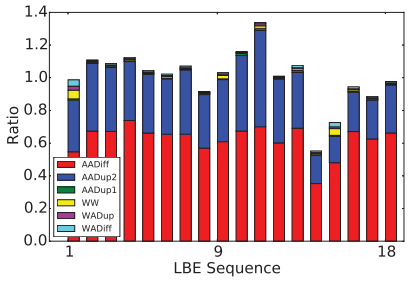


Fig. 8: Ratio of ones containing each update pattern.

so no pattern is recorded. This small number of updates also indicates weaker activity.

To summary, we speculate the LBEs in the cluster are caused by some kind of configuration errors in AS 9121. We believe the anomalous behaviour is not intended, primarily because the resulting instability mainly affects the connectivity and performance from other part of the Internet to the networks owned by AS 9121; in other words, AS 9121 itself is the major victim of the anomaly.

VII. Related Work

Many attempts have been made to detect Internet anomalies through analyzing BGP updates/tables; we just name a few. Comarela and Crovella focused on large-scale coordinated rerouting events reflected in daily routing table changes [4]; by comparison, we adopt much finer granularity and investigate updates instead of tables. I-Seismograph is dedicated to measuring the 'impact magnitude' of famous disruptive events by analyzing the statistics of various attributes of BGP updates [1]; but the data from all used monitors are aggregated. Liu et al. applied the metric 'Betweenness Centrality' to measure the extent of rerouting after disruptive events [3]; while they were concerned with ASes, our focus is on IP prefixes. The framework proposed by Mai et al relies on a simple count of BGP update messages to detect anomalies [6]. In contrast, we apply more sophisticated metrics like density, size, and height to quantify the impact of events. Deshpande et al applied statistical pattern recognition to the feature traces extracted from BGP update messages to detect anomalies [5]; however, like many other works, the impact of monitor-local events is not eliminated.

VIII. Conclusion

Traditional metrics for detecting Internet disruptions are prone to monitor-local events. To cope with the issue, we propose the concept of Update Visibility Matrix (UVM) and Large-scale BGP Event (LBE). First, we formulate the problem of identifying LBE into a bi-clustering problem, then we propose a heuristic algorithm to solve it. We apply the method to the updates related to nine famous incidents; the results suggest a strong correlation between the incidents and the identified LBEs. Next, we analyze ten months' data in 2013 and find that LBEs do exist. By analyzing the properties of these LBEs, we illustrate their anomaly and strong impact. Finally, we conduct a case study of a high-impact incident that generated 18 LBEs. Our future work is two-fold. First, considering the difference between prefixes and monitors, we will weight each prefix and monitor according to their activity, importance, etc. Second, we try to come up with a systematic way to infer the cause of an LBE.

IX. Acknowledgment

This work was supported by the National Basic Research Program of China (973 Program) under Grant 2012CB315803, the National Natural Science Foundation of China under Grant 61133015, Specialized Research Fund for the Doctoral Program of Higher Education under Grant 20120002110060, the National High-Tech Research and Development Program of China (863 Program) under Grant 2015AA015701, and the National Science Foundation of China with No. 61402255.

References

[1] J. Li and S. Brooks, "I-seismograph: Observing and measuring internet earthquakes," in *IEEE INFOCOM*, 2011, pp. 2624–2632.
[2] C. Labovitz, R. Malan, and F. Jahanian, "Internet routing instability," in *ACM SIGCOMM*, 1997, pp. 115–126.
[3] Y. Liu, X. Luo, R. Chang, and J. Su, "Characterizing inter-domain rerouting by betweenness centrality after disruptive events," *IEEE JSAC*, vol. 31, no. 6, pp. 1147–1157, 2013.
[4] G. Comarela and M. Crovella, "Identifying and analyzing high impact routing events with PathMiner," in *IMC*, 2014, pp. 421–434.
[5] S. Deshpande, M. Thottan, T. Ho, and B. Sikdar, "An Online Mechanism for BGP Instability Detection and Analysis," *IEEE Transactions on Computers*, no. 11, pp. 1470–1484, 2009.
[6] J. Mai, L. Yuan, and C. Chuah, "Detecting BGP anomalies with wavelet," in *NOMS*, 2008, pp. 465–472.
[7] I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo, "The Maximum Clique Problem," in *Handbook of Combinatorial Optimization*. Springer US, 1999, pp. 1–74.
[8] "Route Views Project," http://www.routeviews.org/, accessed: November 2 2014.
[9] "RIPE RIS Raw Data," http://www.ripe.net/data-tools/stats/ris/ris-raw-data, accessed: November 2 2014.
[10] "Active bgp entries of AS 65000," http://bgp.potaroo.net/as2.0/bgp-active.html, accessed: November 19 2014.
[11] R. Oliveira, D. Pei, W. Willinger, B. Zhang, and L. Zhang, "The (in)Completeness of the Observed Internet AS-level Structure," *IEEE/ACM ToN*, vol. 18, no. 1, pp. 109–122, 2010.
[12] "Tier-1 ISPs," http://en.wikipedia.org/wiki/Tier_1_network, accessed: January 22 2015.
[13] "Customer cones of Autonomous Systems," http://data.caida.org/datasets/2013-asrank-data-supplement/data/, accessed: January 22 2015.
[14] "AS to nation/district mapping," http://www.cidr-report.org/as2.0/autnums.html, accessed: January 23 2015.
[15] P. Cheng, B. Zhang, D. Massey, and L. Zhang, "Identifying BGP routing table transfers," *Computer Networks*, vol. 55, no. 3, pp. 636–649, 2011.
[16] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian, "Delayed internet routing convergence," in *ACM SIGCOMM*, 2000, pp. 175–187.
[17] A. Feldmann, H. Kong, O. Maennel, and A. Tudor, "Measuring BGP pass-through times," in *Passive and Active Network Measurement*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, pp. 267–277.
[18] Y. Liu, X. Luo, R. Chang, and J. Su, "Characterizing inter-domain rerouting after japan earthquake," in *IFIP Networking*, 2012, pp. 124–135.
[19] E. Chan, X. Luo, W. Fok, W. Li, and R. Chang, "Non-cooperative Diagnosis of Submarine Cable Faults," in *Passive and Active Measurement*, 2011, pp. 224–234.
[20] E. Martin, K. Hans-peter, S. Jörg, and X. Xiaowei, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *ACM KDD*, 1996, pp. 226–231.
[21] J. Li, M. Guidero, Z. Wu, E. Purpus, and T. Ehrenkranz, "BGP routing dynamics revisited," *SIGCOMM CCR*, vol. 37, no. 2, pp. 5–16, 2007.