Improving the Reliability of Inter-AS Economic Inferences Through a Hygiene Phase on BGP Data

Enrico Gregori\textsuperscript{a}, Alessandro Improta\textsuperscript{a,\ast}, Luciano Lenzini\textsuperscript{b}, Lorenzo Rossi\textsuperscript{a}, Luca Sani\textsuperscript{c}

\textsuperscript{a}Institute of Informatics and Telematics, Italian National Research Council, Pisa, Italy
\textsuperscript{b}Information Engineering Department, University of Pisa, Pisa, Italy
\textsuperscript{c}IMT - Institute for Advanced Studies, Lucca, Italy

Abstract

Over the last few years researchers have tried to shed light on the economic features that drive the inter-domain routing of the Internet, by inferring economic inter-AS relationships from raw BGP data collected by research projects such as BGPmon, PCH, RIS and RouteViews. Although this kind of data contains spurious entries mostly caused by router misconfigurations on BGP border routers and showing up during BGP path exploration, none of the methodologies provide an adequate data hygiene phase, thus affecting the accuracy of the inferences drawn. In this paper we outline a new methodology that can purge a large amount of spurious routes from BGP raw data by leveraging on robust statistical concepts rather than on debatable thresholds. To quantify the performance of our methodology we apply an enhanced version of an existing economic tagging algorithm on non-cleaned and cleaned data respectively. We found that 42.01\% of different AS paths advertised to BGP route collectors in July 2013 appear only in spurious routes and that, in the absence of an appropriate data hygiene phase, they can affect the accuracy of the economic inferences regarding about 8\% of connections found in BGP raw data.

Keywords: Autonomous Systems, BGP, Internet Measurement, Inter-AS

\ast Corresponding author - Contacts: +39 050 315 3169 - alessandro.improta@iit.cnr.it
Email addresses: enrico.gregori@iit.cnr.it (Enrico Gregori), alessandro.improta@iit.cnr.it (Alessandro Improta), l.lenzini@iet.unipi.it (Luciano Lenzini), lorenzo.rossi@iit.cnr.it (Lorenzo Rossi), luca.sani@imtlucca.it (Luca Sani)

Preprint submitted to Computer Networks April 15, 2014
1. Introduction

Very little is known about the real structure of the Internet and the economic dynamics behind it. To fill up this gap, researchers have analysed it from a topological perspective in order to reveal any potential weaknesses in its structure. The Autonomous System (AS) level topology offers one of the most suitable levels of abstraction for this type of analysis, since it directly represents how the organizations that constitute the Internet are connected. In this topology, the Internet is modeled as a graph in which each node represents an AS, and each edge represents at least one inter-domain connection established between pairs of ASes via the Border Gateway Protocol (BGP) [1]. However, the Internet AS-level topology in itself does not provide much useful information regarding the real interaction between ASes, since the IP address space exchanged between each couple of ASes strictly depends on the type of economic agreement established between them. This is technically implemented by applying outbound route filters described via BGP import/export policies. Despite the large number of possible economic agreements, inter-AS relationships can still be categorized by three main classes on the basis of the set of routes that each AS announces to the other: provider-to-customer (p2c) – or customer-to-provider (c2p) – peer-to-peer (p2p) and sibling-to-sibling (s2s) [2]. In p2c and c2p relationships, the provider announces to the customer the routes required to reach each Internet destination, by selecting them from the routes that the provider obtained from its customers, providers and peers (if any) plus the routes owned by the provider itself. The customer, on their part, announces to the provider only those routes related to the customer’s own IP prefixes and routes obtained from its customers (if any). In p2p relationships, each AS announces to the other peer the routes related to its own IP prefixes and the routes obtained from its customers, typically free-of-charge. Finally, in s2s relationships each AS acts like a provider by announcing its full routing table. This level of granularity is typically considered as consistent with reality [3], although there are some exceptions [4] mainly caused by policies established on a geographical basis [5, 6]. The existence of these types of relationships implies that the largest amount of graph paths that can be extracted from the Internet AS-level topology do not exist in reality. This means that the
bare undirected topology cannot be used to accurately study or model the real routing behavior of the Internet. Thus, knowledge of inter-AS economic relationships plays a fundamental role, and any realistic Internet AS-level analysis has to take these relationships into account. The common approach is to transform the undirected AS-level topology into an economic AS-level topology where each edge is tagged with a proper economic label which reflects the type of relationship existing between the involved ASes. Despite (or due to) their key role, details regarding inter-AS economic relationships are not usually publicly available, and researchers have needed to develop heuristics ([2, 7, 8, 9, 10, 11, 12]) to infer them and enhance the bare undirected topology, typically by exploiting data gathered via BGP route collectors (RCs). However, this type of data contains several spurious entries caused by router misconfigurations [13]. These entries show up during BGP path exploration phenomena [14] and can potentially affect the accuracy of the inferences drawn. Despite some of the heuristics used to try to minimize the impact of these routes by limiting the impact of short-lived routes [9, 11], no definitive solution has been developed yet. In this paper we propose a methodology to fill this gap by introducing a preliminary data hygiene phase where spurious routes are identified and purged from the BGP data available. In addition, we propose a threshold-free tagging heuristic – based on the valley-free principle [2] and on a well-known set of provider-free ASes – where cleaned BGP data are used to tag each connection with an economic label. The data hygiene phase is designed to remove spurious routes which may affect the accuracy of the inter-AS economic inferences. Consequently, it is not aimed at removing all general spuriousness in BGP data. To the best of our knowledge, this is the first work that significantly takes into account the temporal characteristics of BGP data to devise an appropriate data hygiene phase in order to perform more reliable economic inferences.

The paper is organized as follows. Section 2 describes the BGP data sources and the methodology used to work with raw BGP data. Section 3 details the causes and the characteristics of spurious BGP routes present in data repositories. Section 4 describes the methodology to clean raw BGP data by identifying and discarding spurious routes and introduces the threshold-free economic tagging algorithm. Section 5 provides the results of the proposed methodologies on data collected during July 2013. Section 6 describes the state of the art regarding inter-AS economic relationship inferences. Finally, Section 7 concludes the paper.
2. Dealing with BGP raw data

The Border Gateway Protocol is today the de-facto inter-domain routing protocol. The primary function of the BGP protocol is to “[…] exchange network reachability information with other BGP systems, […] and that includes information on the list of Autonomous Systems (ASes) that reachability information traverses.” [1]. To achieve this, BGP relies on the exchange of UPDATE messages between pairs of BGP speakers once a BGP session has been set up. UPDATE messages inform the other BGP party which part of the network can be reached through the BGP connection considered, by announcing or withdrawing a list of routes. In this context, a route is considered as “a unit of information that pairs a set of destinations with the attributes of a path to those destinations” [1].

From an Internet AS-level analysis perspective, the most interesting path attribute that is exchanged between BGP speakers is the AS_PATH. This attribute is mandatory and well-known, and contains the sequence of ASes that will be traversed to reach the announced destinations. This attribute is originally created by the AS border router (ASBR) which owns the announced prefixes contained in the Network Layer Reachability Information (NLRI) field, and is modified whenever another ASBR propagates the route on the Internet. This BGP feature is typically exploited by research projects to infer and analyse the Internet AS-level topology. Over the years, research projects such as RIS [15] and RouteViews [16] have deployed a set of route collectors around the world aimed at collecting as much information as possible regarding Internet routing. These route collectors are devices that mimic a BGP router with the sole purpose of gathering UPDATE messages from cooperating ASes. Collected messages contain announcements of new routes and/or withdrawals of previously announced routes and are used by the RCs to update their Routing Information Base (RIB) table, like a real AS Border Router (ASBR). These projects publicly release, typically in MRT format [17], periodic snapshots of the RIB of each RC and dumps of the related UPDATE messages, which can be then exploited to study the evolution of the RC RIB offline [18].
The routing information contained in this data is considered reliable since it is collected directly from devices that are actually participating in the inter-domain routing. However, depending on the type of analysis needed, BGP data cannot be used in its raw form. For example, only a few route collectors record the state information regarding their BGP sessions, and understanding when the routes contained in their local RIBs need to be invalidated due to a session failure is not straightforward. The lack of these entries leads routes that are withdrawn during the BGP session downtime to be considered as still active after the session has been re-established [19]. In the absence of state information, it is still possible to detect BGP session failures by identifying full routing table transfers in the stream of UPDATE messages – which occur right after a BGP session is (re)established [1]. This is done by exploiting one of the algorithms available in the literature (e.g. [20, 21]), and then considering each route in the RIB as withdrawn whenever a session failure is found. To perform our analyses we used data collected by BGPmon [22], PCH [23], RIS [15] and RouteViews [16] projects in July 2013. In detail, we downloaded the first RIB snapshot of the month for each route collector available and we traced the evolution of each RIB by exploiting the UPDATE messages available. This is done by withdrawing each route in the RIB whenever a session failure is found via the Minimum Collection Time (MCT) algorithm described in [21] and adding an implicit withdrawal for each route still present in the RIB at the end of the month. Note that, although a route is formally defined as “a unit of information that pairs a set of destinations with the attributes of a path to those destinations” [1], in this work we consider a route as being identified by the attributes that allow us to identify a BGP session and the AS path toward the destination, i.e. the attribute triplet <destination, BGP session ID\textsuperscript{1}, AS_PATH>.

3. BGP misconfigurations and inferences

Entries in BGP data contain AS paths announced on the Internet, however they still cannot be used in their raw form in all types of inference. Inter-AS economic relationship inferences suffer from a particular class of entries – hereafter spurious entries – which can lead to wrong results. These

\textsuperscript{1}We used the MRT fields Peer IP Address and Peer AS to identify a BGP session.
entries are typically caused by BGP export policy misconfigurations [13] and show up during BGP path exploration phenomena [14], which occurs when an ASBR receives a withdrawn message. In detail, before declaring the destination network unreachable, an ASBR tries to use several different routes available in its Adj-RIB-in (Adjacent RIB, incoming) tables for which a withdrawn message has not yet been received, very likely due to time propagation delays. As a consequence, during the lifetime of this process several UPDATE messages that contain routes not commonly used are generated and propagated. These may also include messages containing routes that violate the business relationship agreement between ASes – i.e. violate the valley-free principle described in [2] – typically due to human errors in defining BGP export policies on ASBRs ([9, 13, 14]). These spurious routes are then propagated through the Internet up to the route collectors and, thus, appear in public datasets.

Consider for example the scenario depicted in Figure 1, and assume that

1) each AS selects the route with the shortest path to reach $P$,
2) each link has the same propagation delay, and
3) $C$ applies a prefix-based outbound filtering, i.e. $C$ announces to its providers every route to reach its customer networks [9, 13], irrespectively which neighbor the route was received from. For example in this scenario, $C$ preferred path to reach prefix $P$ is through its customer $D$, over the AS paths $A$-$B$-$D$ and $B$-$D$ received from its providers $A$ and $B$ respectively. Note however that paths $A$-$B$-$D$ and $B$-$D$ are still stored in $C$ Adj-RIB-In tables as alternative routes [1]. Now suppose that $D$ sends a withdrawn message containing $P$ to both $B$ and $C$ due to a network failure. $C$ will then remove from its RIB the AS path $D$ to reach $P$ and will search
for alternatives before informing its neighbors that \( P \) is unreachable. Due to time propagation delays, \( C \) has not yet received any withdrawn message from either \( A \) or \( B \) concerning \( P \). Thus \( C \) will select the best route from the alternatives stored in its Adj-RIB-In tables, i.e. the route with AS path \( B-D \). The same thing happens on \( B \), where the chosen route is the route with AS path \( C-D \).

Since \( C \) performs an outbound filtering implemented as described above, it announces to \( A \) and \( B \) the route \( C-B-D \) to reach \( P \), while \( B \) announces to \( A \) and \( C \) the route \( B-C-D \). Note that the route announced by \( C \) to \( A \) is clearly in contrast with the p2c agreement signed with \( B \). As a consequence, \( A \) may decide to use this route (e.g. if \( A \) applies the commonly used prefer-customer policy [24]) and then announce to the RC \( R \) the spurious route \( A-C-B-D \). As soon as \( C \) and \( B \) realize that loop-free\(^2\) routes to reach \( P \) do not exist, they advertise a withdrawn message towards \( A \). Finally, since \( A \) has no other alternatives to reach \( P \), it will declare that the destination is unreachable, which will be recorded by \( R \).

In other words, \( R \) sees that \( C \) is transiting traffic between its providers \( A \) and \( B \) for a short time since the network \( P \) will be withdrawn from the Internet at the end of the convergence of BGP protocol. As a result, several economic inferences made using BGP public data are potentially wrong, since AS paths contained in spurious entries interfere with those obtained from stable AS paths. Note that these spurious routes do not actually affect the network performances of the AS owner of the misconfigured ASBR due to their ephemeral nature, thus it is very likely that several misconfigurations go unnoticed by network operators. Likewise, it is unlikely (but still possible) that an AS unconsciously maintains a BGP misconfiguration for a long period of time, since an AS which transits non-planned traffic for at least one of its providers (or peers) will perceive that its network performance is degraded.

The presence of spurious entries is well-known. The common solution it is to consider every transient route as a potential source of problem for the inferences, thus applying heuristics that limit their effect on the final tagging

\(^{2}\text{Due to the BGP loop avoidance mechanism, every route that contains the local AS number in the AS path attribute should not participate in the best route selection mechanism [1].}
results ([2, 9, 11]). This approach is not entirely correct, since not all transient routes carry no-valley-free AS paths. For example, backup routes may have a relatively short lifespan, but they carry perfectly legitimate AS paths from which previously unnoticed p2c relationships between pair of ASes may be inferred. One of the most evident characteristics which differentiates a spurious entry from a normal transient route is related to its AS path length. To prove this, Figure 2 depicts the Probability Distribution Function (PDF) of the AS path length of four sets of routes, consisting of: a) every route, b) routes that lasted less than 60 seconds, c) routes that lasted less than one hour and d) routes containing easily detectable no-valley-free AS paths. Note that the four sets are not disjoint, e.g. routes in set b are also included in set a and c. To identify the routes belonging to set d, we exploited a priori knowledge regarding the provider-free property owned by a limited set of ASes, i.e. those ASes that do not need to buy transit from any other AS

---

3 We removed multiple consecutive ASes in every AS path, as well as identifiable human errors [9]
to reach each Internet destination. A list of these ASes can be found on Wikipedia [25], whose validity has been discussed in [9] and [26]. We then selected every AS path that includes two (or more) provider-free ASes not directly connected to each other, i.e. every AS path in which a third AS transited traffic for one of the provider-free ASes. Hereafter we refer to these routes as inter-PF routes. As can be seen from Figure 2, inter-PF routes have a greater probability of having longer AS paths than routes in the other sets. This is not the general behavior of short-lived routes (sets b and c), which can be generated for other reasons (e.g. backup connections, route flapping, traffic engineering issues) and which typically carry perfectly legitimate AS paths. In addition routes belonging to set d are usually short-lived with respect to the total set of routes, as shown in the Complementary Cumulative Distribution Function (CCDF) depicted in Figure 3, thus confirming their extremely transient nature. The largest amount of long-lived routes belonging to this set can still be considered as legitimate, since it largely consists of routes in which provider-free ASes are separated by a third AS which can be traced back to organizations that own one of the provider-free ASes involved. Indeed, worldwide ISPs often own more than one AS number, both due to traffic engineering (e.g. Verizon Communications manages AS 701 for North-American routing, AS 702 for European, Middle Eastern and African routing and AS 703 for Asian and Pacific region routing) and because of mergers and acquisitions [27]. Thus, it seems reasonable to assume that spurious routes can be identified by their abnormal AS path length and by their short-lived appearance in BGP datasets. From an economic relationship inference perspective, it is also worth noting that the AS paths of the routes in set d involve a total of 9,143 connections over the 185,224 found in the BGP data, i.e. the inferences related to at least 4.94% of AS connections found on the Internet are potentially affected by spurious routes.

4. Towards spuriousness-free inferences

To date, economic tagging algorithms have tried to mitigate the effects of spurious routes by using thresholds, either on the number of entries used to infer transit relationships [2] or on the lifespan of each route, e.g. cutting off every route lasting less than two days [11] or whose lifespan is not comparable with long-lasting lifespans [9]. In both cases, the main problem is how to select a suitable value for the threshold. In this section we pro-
pose a methodology which, rather than relying on arbitrary and debatable thresholds, identifies spurious entries by analyzing the data characteristics themselves and purging them from the initial dataset before inferences are drawn.

4.1. Preliminary data hygiene phase

Section 3 highlighted that spurious routes have a short lifespan and abnormally long AS paths. To identify candidate spurious routes, we devised a three-step filter (see Fig. 4) which exploits the following consideration: an AS tends to select predominant routes to proficiently reach a destination during a month [28, 29] and, thus, predominant AS path lengths to reach a destination. It is thus possible to apply data binning to the time-based AS path length distribution in order to retrieve a binned normal distribution centered around the mode of the original distribution. As a consequence, AS paths that do not fall within the normal binned behavior can be marked as outliers by applying the three-sigma rule, commonly used to identify outliers in normal distributions [30]. More in detail, step a) of the algorithm (Fig. 5) aims to create the AS path length distribution experienced throughout the
Figure 4: Data hygiene phase filters

entire month by each pair \(<\text{feeder IP } f, \text{ destination } p>\) and to compute the proper bin value to consider this distribution as a well-approximated normal distribution. To do this, we recreate the RIB dynamics related to the pair \(<f,p>\) and we sample the \textit{AS path length} experienced between the first and the last announcement of \(p\) every second, i.e. we assign the total amount of time (in seconds) to each length found in which a route with an AS path with that length is found to be active for the pair \(<f,p>\) during the sampling period (lines 3-8). We then compute the correct size of the bin and we transform the distribution just computed into a binned normal distribution centered on the mode of the original distribution, i.e. bin zero contains \textit{at least} the predominant length of the distribution. The size of the bin is determined by the \textit{left\_edge} and \textit{right\_edge}, which are initially set equal to the distribution mode (line 10). Then, each length value is assigned to the appropriate bin (lines 15-25) and we check if the values of the first quartile and of the third quartile of the binned distribution are the same (line 29). In this case, the mode of the \textit{binned} distribution predominates over the other
Input: feeder f, destination d, routes f,d

foreach r in routes f,d
    as_path = r.as_path
    as_path = remove_prepending(as_path)
    length_as_paths [as_path_length].insert(as_path)
    length_distr [as_path_length] += lifespan(as_path)
left_edge = right_edge = compute_mode(distr)
ranked_length_distr = sort_by_lifespan_desc( length_distr )
for (i = 1; i < size(ranked_length_distr); i++)
    bin_size = right_edge - left_edge + 1
    binned_distr f,d[bin] = length_in_bins = 0
    foreach (length, lifespan) in length_distr
        if (length < left_edge)
            bin = (length−left_edge)/ bin_size
        if ((length−left_edge)%bin_size)
            bin−−
        if (length > right_edge)
            bin = (length−right_edge)/ bin_size
        if ((length−right_edge)%bin_size)
            bin++
        binned_distr f,d[bin] += lifespan
        as_paths_in_bins f,d[bin].insert( length as_paths [length] )
1st_quartile = compute_1st_quartile( binned_distr f,d )
3rd_quartile = compute_3rd_quartile( binned_distr f,d )
if (1st_quartile == 3rd_quartile)
    break
if (ranked_length_distr [i] < left_edge)
    left_edge = ranked_length_distr [i]
if (ranked_length_distr [i] > right_edge)
    right_edge = ranked_length_distr [i]
bin_size = right_edge − left_edge

Output: binned_distr f,d, as_paths_in_bins f,d

Figure 5: Step a) Binned AS path length distribution creation

bin values and the binned distribution can be considered as normal with a
good approximation. Otherwise, the procedure is repeated increasing the bin
size by including the next longest-lasting length and updating the value of
the left or right edge accordingly (lines 32-35).

In step b) (Fig. 6), we apply the three-sigma rule [30] on the binned
distribution that has just been created, considering every AS path included
in a bin whose value is greater than μf,d + 3σf,d as an outlier, where μf,d
and σf,d are the temporal mean and the standard deviation respectively of
the binned distribution created during step a). We compute μf,d and σf,d as
Input: f, d, binned_distr_{f,d}, as_paths_in_bins_{f,d}

\[
\mu_{f,d} = \text{compute\_mean}(\text{binned\_distr}_{f,d})
\]

\[
\sigma_{f,d} = \text{compute\_std\_dev}(\text{binned\_distr}_{f,d})
\]

foreach bin in binned_distr_{f,d}

if bin > \mu_{f,d} + \sigma_{f,d}

\text{D}_{f,d}.insert(\text{as\_paths\_in\_bins}_{f,d}[\text{bin}])

else

\text{M}_{f,d}.insert(\text{as\_paths\_in\_bins}_{f,d}[\text{bin}])

Output: discarded AS paths \text{D}_{f,d}, maintained AS paths \text{M}_{f,d}

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input: f, d, binned_distr_{f,d}, as_paths_in_bins_{f,d}</td>
</tr>
<tr>
<td>2</td>
<td>\mu_{f,d} = \text{compute_mean}(\text{binned_distr}_{f,d})</td>
</tr>
<tr>
<td>3</td>
<td>\sigma_{f,d} = \text{compute_std_dev}(\text{binned_distr}_{f,d})</td>
</tr>
<tr>
<td>4</td>
<td>foreach bin in binned_distr_{f,d}</td>
</tr>
<tr>
<td>5</td>
<td>if bin &gt; \mu_{f,d} + \sigma_{f,d}</td>
</tr>
<tr>
<td>6</td>
<td>\text{D}<em>{f,d}.insert(\text{as_paths_in_bins}</em>{f,d}[\text{bin}])</td>
</tr>
<tr>
<td>7</td>
<td>else</td>
</tr>
<tr>
<td>8</td>
<td>\text{M}<em>{f,d}.insert(\text{as_paths_in_bins}</em>{f,d}[\text{bin}])</td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Output: discarded AS paths \text{D}<em>{f,d}, maintained AS paths \text{M}</em>{f,d}</td>
</tr>
</tbody>
</table>

Figure 6: Step b) Three-sigma rule filtering

follows:

\[
\mu_{f,d} = \frac{\sum_{i=1}^{N_{f,d}} b_i \cdot w_i}{\sum_{i=1}^{N_{f,d}} w_i} \tag{1}
\]

\[
\sigma_{f,d} = \sqrt{\frac{\sum_{i=1}^{N_{f,d}} w_i \cdot (b_i - \mu_{f,d})^2}{\sum_{i=1}^{N_{f,d}} w_i}} \tag{2}
\]

where \( N_{f,d} \) is the number of bins, and \( b_i \) and \( w_i \) are the value of the \( i \)-th bin and its temporal weight, respectively. Note that \( \mu_{f,d} + 3\sigma_{f,d} \) is still a threshold, but its value is not arbitrary. Rather, it has both a real and statistical significance, since it depends on the AS path lengths that \( f \) uses to reach \( d \). As a result of this filtering step, each AS path is classified as dropped or maintained. However some AS paths generated during a path exploration phenomenon may still not have been correctly discarded, due to their limited AS path length. Step c) (Fig. 7) aims to identify these remaining routes by checking whether any route was announced less than thirty seconds earlier than any spurious route identified in step b). In such cases, the closest precedent route is flagged as part of the path exploration phenomenon (line 9) and checks are made once again starting from this route, until no more routes are found in the time interval analysed. Every AS path passing through this last filter is declared as valid and can be used to draw economic inferences. Note that thirty seconds is not an arbitrary value, but instead reflects a protocol operational standard value, since it is the the default value of the \textit{MinRouteAdvertisementIntervalTimer} (MRAI timer) [1]. This parameter is the minimum time interval that should elapse between consecutive UPDATE messages for a given route sent by an ASBR to its neighbor. It is usually used to limit the amount of announced routes.
Input: $f$, $d$, $\text{routes}_{f,d}$, $D_{f,d}$, $M_{f,d}$

foreach $r$ in $\text{routes}_{f,d}$
  if ($r\.as\.path \in D_{f,d}$)
    $\text{curr}_r = r$
    while (exists previous route)
      $\text{prev}_r = \text{previous\.route}(\text{curr}_r)$
      if ($\text{birth}(\text{curr}_r) - \text{birth}(\text{prev}_r) \leq \text{MRAI}$)
        $\text{to\.check}.\text{remove}(\text{prev}_r\.as\.path)$
      else
        break
    $\text{curr}_r = \text{prev}_r$

$\mathcal{V}.\text{insert}\left(\text{to\.check}\right)$
$\text{to\.check} = \emptyset$
else
  $\text{to\.check}.\text{insert}\left(r\.as\.path\right)$

$\mathcal{V}.\text{insert}\left(\text{to\.check}\right)$

Output: valid AS paths $\mathcal{V}$

Figure 7: Step c) MRAI-based event filtering during BGP transients to improve BGP routing convergence ([31, 32, 33]). Note that network operators typically leave this parameter at the default value, but some use a lower value to reduce the convergence times even further [34].

To better understand the methodology, consider as an example the evolution of the routes related to pair $<d, f>$ (Table 1), where $d = 2a03:2040::/32$ and $f = (2001:43f8:1f0::29, \text{AS } 6968)$, which has been found to involve at least one inter-PF route. The steps involved are depicted in Figure 8. Firstly, the correct bin value is computed in order to consider the distribution as normal. The distribution is initially centered around its mode value ($\text{length}=6$) and analysed with a bin size equal to one. Since the distribution does not show a predominant bin, the bin size is enlarged to two, i.e. the difference between the first ($\text{length}=6$) and the second ($\text{length}=5$) modes plus one. The resulting distribution, although more peaked, still does not satisfy the condition of normality required. Thus, the bin is further enlarged ($\text{bin size}=3$) to also include the third mode ($\text{length}=7$). The distribution is now extremely peaked, and step b) cuts off all the AS paths included in the final +1 bin, including the inter-PFs.
Figure 8: Binned distribution creation related to routes collected for \( <\mathbb{d}, \mathbf{f}> \)

<table>
<thead>
<tr>
<th>AS path length</th>
<th>lifespan [s]</th>
<th>final bin</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>220,858</td>
<td>0</td>
<td>valid</td>
</tr>
<tr>
<td>6</td>
<td>282,158</td>
<td>0</td>
<td>valid</td>
</tr>
<tr>
<td>7</td>
<td>211,402</td>
<td>0</td>
<td>valid</td>
</tr>
<tr>
<td>10</td>
<td>26,774</td>
<td>+1</td>
<td>dropped</td>
</tr>
</tbody>
</table>

\[ \mu_{\text{binned}} = 0.036 \]
\[ \sigma_{\text{binned}} = 0.03 \]
\[ \mu_{\text{binned}} + 3\sigma_{\text{binned}} = 0.09 \]

Table 1: AS path length distribution of routes related to \( <\mathbb{d}, \mathbf{f}> \)

4.2. Economic inference phase

Now that we have a methodology able to eliminate most spurious routes, to infer inter-AS economic relationships we need an appropriate algorithm
that can exploit the filtered routes. We decided to enhance the techniques described in [9] and [11] due to their strong bonds with collected raw data and few hypotheses and assumptions supporting the heuristic. The original algorithms rely on the valley free property of AS paths [2] and on a priori knowledge of a set of provider-free ASes [25]. The valley-free property implies that in a given AS path: a) at most one single p2p connection exists, b) no c2p connections can follow a p2p connection, and c) no c2p connections can follow a p2c connection. By also considering the provider-free property of a well-identifiable set of ASes, this also means that d) any connection that appears before a provider-free AS can be considered a c2p, and e) any connection that appears after a provider-free AS can be considered a p2c. Note that connections between provider-free ASes are considered p2p by definition. The main difference between the two algorithms is in how they handle spurious routes. Since the preliminary step described in Section 4 entails handling problematic routes, our algorithm is exactly the same as the one in [11] but without the two-day time threshold filtering. Likewise it is the same as the algorithm proposed in [9] but with $N_{MAG} = \infty$, i.e. with no time threshold. In addition, we introduce an enhancement step, which enables the quality of the inferences drawn to be refined further.
Figure 10: Example of application of the enhanced step

The enhancement (Fig. 9) exploits the inferences drawn by the original algorithm to infer additional p2c (or c2p) relationships also from those AS paths that do not contain any provider-free ASes. In the original algorithm, these AS paths led only to simple p2p connections, which were potentially overwritten by p2c (c2p) relationships whenever a path was found containing a provider-free AS and the proof that one of the two ASes was providing transit to the other. However, despite the lack of provider-free ASes, these AS paths may still contain useful information to infer further transit relationships. Connections inferred to be p2c (or c2p) may appear because they were found in at least one other AS path containing a provider-free AS. This information can be exploited to force these AS paths to comply with the valley-free property, by turning some p2p relationships present in AS paths without provider-free ASes into c2p (or p2c) relationships by exploiting the same rationale as the original algorithm. If a set of p2p connections precedes a c2p connection, each is converted into a c2p. Likewise, if any p2p connection follows a p2c connection, each is converted into a p2c. In both cases, in order to detect an s2s connection the presence of other inferences made on different AS paths must be taken into account.

The key rationale behind this algorithm is that a p2c label is tangible proof that an AS announced networks previously received from one of its peers or providers to another AS, while a p2p label only proves that the ASes
exchanged routing information related to each others’ customers. Thus, if multiple different tags are found for the same connection, a p2p label is overruled by a p2c label, while contrasting p2c and c2p labels lead to an s2s label, i.e. each AS is a provider of the other. For example, consider the situation in Figure 10. Despite the fact that the AS path D-A-B-E-F does not contain any provider-free AS, it is still possible to infer that D is customer of A, and that F is customer of E. This because, from the AS path A-B-P1-P2-C, there is proof that A and E announced routing information to them which was related to their providers.

5. Results

We applied our methodology to the routes extracted from BGP data provided by BGPmon [22], PCH [23], RIS [15] and RouteViews [16] in July 2013. By applying the data hygiene phase on this set of routes we found that 42.01% of the total 41 million different AS paths collected appear only in spurious routes. The data hygiene phase was able to eliminate 93.30% of the inter-PF routes that led existing tagging algorithms to inexistent s2s relationships. Table 2 compares the results obtained using every AS path available and only the AS paths purged of spurious routes. Despite the large number of AS paths discarded, their effect on economic inferences is rather limited. This can be explained by the fact that only a limited set of ASes are misconfigured and contribute to creating no-valley-free AS paths. In addition, over the last few years several techniques have been proposed to reduce the propagation of spurious routes [35] to limit the BGP traffic volume due to excessive UPDATE messages. However, the percentage of changed tags is not negligible (8.52%), and cannot be ignored. It should also be noted that 5,641 connections out of 185,224 (3.04%) were found to appear only in spurious routes, and due to their ephemeral nature were not tagged.
Table 3: Comparison of the results of economic tagging algorithms

<table>
<thead>
<tr>
<th></th>
<th>p2c</th>
<th>p2p</th>
<th>s2s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gregori et al. [9]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N_{MAG} = 1$</td>
<td>93,234</td>
<td>90,374</td>
<td>1,616</td>
</tr>
<tr>
<td>$N_{MAG} = 2$</td>
<td>94,616</td>
<td>88,624</td>
<td>1,984</td>
</tr>
<tr>
<td>$N_{MAG} = 3$</td>
<td>95,618</td>
<td>87,410</td>
<td>2,196</td>
</tr>
<tr>
<td>$N_{MAG} = 4$</td>
<td>97,409</td>
<td>85,271</td>
<td>2,544</td>
</tr>
<tr>
<td>$N_{MAG} = 5$</td>
<td>100,387</td>
<td>81,359</td>
<td>3,478</td>
</tr>
<tr>
<td>$N_{MAG} = 6$</td>
<td>100,615</td>
<td>81,047</td>
<td>3,562</td>
</tr>
<tr>
<td>$N_{MAG} = 7$</td>
<td>100,702</td>
<td>80,917</td>
<td>3,605</td>
</tr>
<tr>
<td>$N_{MAG} = \infty$</td>
<td>100,858</td>
<td>80,708</td>
<td>3,658</td>
</tr>
<tr>
<td>Oliveira et al. [11]</td>
<td>91,433</td>
<td>78,919</td>
<td>1,646</td>
</tr>
<tr>
<td>Proposed tagging algorithm</td>
<td>98,931</td>
<td>78,725</td>
<td>1,930</td>
</tr>
</tbody>
</table>

Note also that the increasing amount of p2p connections together with the simultaneous decrease in p2c connections means that several p2c inferences were based solely on spurious routes, and are thus likely to be wrong. Due to the nature of our tagging algorithm, a p2c relationship is inferred only when a tangible proof is found of such relationship in the AS paths, while every other connection is considered to be a p2p. Table 3 highlights that the enhancement step enables us to infer an amount of p2c connections that is typically larger than other algorithms, still maintaining the spuriousness-free characteristic introduced by the data hygiene phase. An arbitrary time threshold is not a good solution. In fact the algorithm proposed by Oliveira et al. [11] misses a large amount of p2c exclusively due to their two-day threshold. This also happens in the results obtained with the most conservative $N_{MAG}$ values of Gregori et al.’s algorithm [9], i.e. small values of $N_{MAG}$ that represent the strictest transient filters. On the other hand, by using less conservative $N_{MAG}$ values, some of the missing p2c increase, but also the number of wrongly inferred s2s increases due to the presence of short-lived no-valley-free AS paths. In particular, the larger the value of $N_{MAG}$, the larger the number of inter-PF AS paths (as shown in Section 3) that also participate in the s2s generation.

6. Related work

The first works regarding the inter-domain economic characteristics of the Internet appeared during the late 1990s [36, 37]. A couple of years later
the BGP export policies related to the most common economic inter-AS relationships [2] were described for the first time. These relationships, together with the related BGP export policies, were defined by identifying the routes announced from an AS to its neighbors, leading to the definition of the valley-free property that all AS paths should respect. This property has been widely used in algorithms aimed at inferring the economic relationships behind each AS connection ([7, 8, 9, 10, 11, 12]). The major drawback of these algorithms is that the valley-free property does not always hold, mainly due to the presence of misconfigured export policies on BGP border routers [13], which show up in the form of no-valley-free AS paths announced during BGP path exploration phenomena [14]. This is a well-known issue that could affect the results of inferences regarding economic relationships based on the analysis of raw BGP data. Almost every approach proposed so far has developed a strategy to minimize their impact, but most have focused only on limiting the impact of short-lived routes on the final inferences. Some have tried to minimize the number of no-valley-free routes by formulating an optimization problem ([7, 8, 10, 12]). Others have tried to minimize them by applying a time threshold that is both absolute [11] and relative [9]. In both cases, the choice is debatable. On the one hand, the minimization of the number of no-valley-free paths in the optimization algorithm forces the property to hold, regardless of the data available and the real routing behaviors. On the other hand, identifying the optimal time threshold able to discard transient routes from legitimate routes is hard, and can potentially affect the inferences of the algorithms. Moreover, as shown in Section 3, not every short-lived route carries spurious routes, and discarding them indiscriminately can thus lead to a loss of potentially valuable routing information. To the best of our knowledge, no methodology to identify spurious routes has been developed yet, although several methodologies have been proposed to identify BGP instabilities (e.g. [38, 39]).

7. Conclusions

In this paper we have provided a detailed analysis of the dynamics of the Internet routes collected in July 2013 from the most important route collectors. This analysis highlights the non-negligible presence of spurious routes in BGP data which are potential sources of error for economic-based AS-level analyses. We thus provided a new methodology to detect their presence and purge BGP data from the spuriousness that would lead to wrong
inferences. We developed an appropriate tagging algorithm based on the valley-free principle and on a priori knowledge of a small set of provider-free ASes that exploit cleaned routes to infer inter-AS relationships. The most significant result is that approximately one in every different AS paths collected by route collectors is found only in spurious routes. This large amount of spuriousness has been proved to impact on inferences concerning about 8% of Internet AS-level connections.

We believe that both the spuriousness detection methodology and the proposed tagging algorithm represent a step forwards in inferring more reliable inter-AS relationships. However, the inferences made are still affected by the incompleteness of the data available [11, 40], in particular regarding the number of p2p connections inferred. However that can be solved by extending the number of feeders in route collector projects, especially the number of small-medium multi-homed ASes feeding them [40]. We also believe that our methodology could still be extremely useful for all those works that rely on AS path information gathered by route collectors for their analyses. Examples of this include the monitoring of the Internet AS-level ecosystem in response to extreme situations, network accidents or misconfigurations and the design of realistic BGP simulators. The results of this new tagging approach can be found on the Isolario website [41] and updated monthly.

References


