# When BGP Goes MAD: Multi-Scale Anomaly Detection in Internet Routing

Justin Loye
IIJ Research Laboratory
Tokyo, Japan
jloye@iij.ad.jp

Esteban Bautista
Univ. Littoral Côte d'Opale
LISIC – UR 4491
F-62219 Longuenesse, France
esteban.bautista@univ-littoral.fr

Romain Fontugne
IIJ Research Laboratory
Tokyo, France
romain@iij.ad.jp

## Abstract

Internet disruptions are unpredictable events with potentially harmful societal and economic consequences. The Border Gateway Protocol (BGP) offers a valuable vantage point for their timely detection in the form of protocol-level anomalies. Yet, anomaly detection pipelines in BGP still suffer from severe limitations, such as needing large training datasets or relying on complex features that are hard to extract in real time. To address these issues, this paper introduces MADBGP, a novel anomaly detection pipeline that is based on (*i*) a modeling of the Internet as a temporal graph; and (*ii*) an anomaly detection algorithm that analyzes such temporal graph. Our contribution is threefold: (*i*) we propose a new graph-based modeling of the AS-level Internet; (*ii*) we develop and deploy QuickRIB: a scalable tool for generating temporal graphs from public BGP data; and (*iii*) we extend MAD, a state-of-the-art method for anomaly detection in temporal graphs, to weighted temporal graphs and use it to detect real anomalies for the first time. In sum, MADBGP provides an unsupervised, multi-scale, and near real-time anomaly detection pipeline that is suitable for both researchers and network operators. Extensive experiments on ten major BGP events involving outages and route leaks at different scales demonstrate that MADBGP significantly outperforms IODA's BGP results.

## CCS Concepts

• **Networks** → **Topology analysis and generation**; **Network dynamics**; • **Computing methodologies** → **Anomaly detection**.

## Keywords

BGP, anomaly detection, temporal graphs, Internet routing

**ACM Reference Format:**
Justin Loye, Esteban Bautista, and Romain Fontugne. 2026. When BGP Goes MAD: Multi-Scale Anomaly Detection in Internet Routing. In *Proceedings of the 2026 ACM Internet Measurement Conference (IMC '26), October 12–16, 2026, Karlsruhe, Germany*. ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3777912.3809143

## 1 Introduction

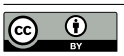
The Internet is the cornerstone of all digital communications, supporting countless services and applications that are integral to modern society. Detecting Internet outages is crucial not only for maintaining a reliable and resilient network, but also for documenting disruptive events that may have a societal impact. The research community has worked on several approaches for identifying such disruptions, among which the analysis of Border Gateway Protocol (BGP) data has become particularly prominent [8, 9, 23, 36, 40]. As BGP is designed to maintain Internet reachability by dynamically rerouting traffic in response to network failures or disconnections, the structural changes it reveals provide a valuable source of information for monitoring and characterizing network events at large-scale and in near real-time.

BGP anomaly detection has been studied for around three decades. The authors of [3] classify 20 of the most popular studies based on the technique used, analyzed BGP attributes, detection efficiency, and effectiveness to find the source of anomalies, concluding that most methods present unsatisfactory tradeoffs between processing times and detection accuracy.

A major contribution to the detection of Internet outages is IODA (Internet Outage Detection and Analysis system) [8], which performs an efficient yet accurate detection by searching for sharp drops in a time-series representing network connectivity over time. Namely, IODA continuously monitors BGP and other data sources to quickly estimate the instantaneous reachability of ASes on the Internet, so that a sudden drop in value is indicative of a network disconnection, thus allowing near real-time detection. This simple approach has proven highly effective in identifying prominent outages, particularly those resulting from physical infrastructure failures and Internet shutdowns [8]. However, IODA's methodology is less suited for detecting more complex routing anomalies, such as BGP route leaks due to misconfigurations, which may not manifest as drops in the connectivity time series but can nonetheless cause significant global routing instability: for example, the Telecom Malaysia (AS4788) route leak in 2015 [45] created connectivity issues at a global scale but as our experiments show IODA is insensitive to it. This should not come as a surprise since the detection of this type of events requires monitoring of the Internet's routing topology and such information is missing in pure time series data.

The most recent works in the domain of BPG anomaly detection have explored graph-based machine learning methods [23, 40]. These techniques capture AS-level topology but require training, lack explainability, and ignore the time dimension.

The goal of this paper is to develop a near real-time and unsupervised BGP anomaly detection pipeline that takes into account both the temporal and structural aspects of the Internet. Yet, this is not a straightforward challenge, as it calls for (a) new techniques capable of quickly processing the large amounts of BGP data provided by




*IMC '26, Karlsruhe, Germany*

ACM ISBN 979-8-4007-2327-8/2026/10
https://doi.org/10.1145/3777912.3809143

public archives (RIPE RIS and RouteViews); (b) an accurate modeling of the Internet's topology and its evolution from the collected BGP measurements; and (c) a real-time and unsupervised algorithm for detecting anomalies in the estimated Internet's temporal graph, being capable of taking into account that anomalies can appear at either the local or the global scale.

To address the above challenges, we propose MADBGP a BGP anomaly detection pipeline based on two main components: *(i)* QuickRIB, a tool for building complete Route Information Bases (RIBs) at a high-frequency rate; and *(ii)* an extension of the MAD algorithm [4] to detect anomalies in weighted temporal graphs inferred from BGP data. We evaluate MADBGP in a variety of historical and recent BGP anomalies, using IODA's BGP results as a baseline. Our experiments demonstrate that not only MADBGP successfully identifies outages, but it also detects rerouting events missed by IODA, pinpointing the impacted ASes.

This work makes the following key contributions:

- We propose QuickRIB, an efficient and extensible tool for building high-frequency RIBs from public BGP data and deriving AS-level topologies.
- We introduce the notion of hegemony graphs, a novel modeling of the Internet leveraging the AS-hegemony centrality metric to correct artificial biases towards vantage points, resulting in an accurate representation of the Internet in the form of a temporal graph.
- We extend the MAD algorithm, a state-of-the-art anomaly detector for binary temporal graphs, to weighted temporal graphs, applying it to real anomalies for the first time.
- We combine the previous tools into a unified pipeline coined MADBGP, providing an unsupervised, near real-time, multi-scale and theoretically sound approach for anomaly detection in BGP.
- We introduce six new evaluation scenarios that complement four classically used ones, collectively demonstrating that (i) MADBGP outperforms IODA BGP results; and (ii) MADBGP correctly spots the ASes concerned by the anomaly.
- We release the implementation of MADBGP as an open source library [1].

## 2 Background and Related Work

### 2.1 BGP data processing

Due to the volume and format of data in BGP public archives, specialized tooling is required to process large amount of data at a fine granularity. Fonseca [15] proposed a tool for extracting temporal features from BGP data. A key advantage of their tool is the RIB update mechanism, which allows for BGP feature time series extraction at a finer resolution than the publication of RIBs (2 hours and 8 hours for RouteViews and RIS, respectively). However, this approach does not support processing multiple route rollectors simultaneously.

This limitation was addressed with BGPview [34] and BML [24], which, in addition to enabling RIB updates, allow users to define their own analysis modules to extract features from RIBs. These tools represent the state of the art but are still hard to use in practice due to the computationally expensive processing of large RIBs. The analysis modules and RIB updating is de-coupled, meaning that each analysis module process in batch all RIB entries, which is computationally expensive due to the large size of the RIB (149 millions entries in our experiment reported in Table 7).

To overcome this performance limitation, BGPCorsaro [34] couples the RIB state and analysis modules, allowing analysis modules to be updated incrementally by each BGP update. However this tool has a limited set of available analysis modules and is hard to extend due to its complexity and lack of documentation. Thus, it is hard to build upon it in order to develop new analysis modules or exploit currently existing ones. In order to provide an alternative solution, we develop QuickRIB, a new BGP processing tool written in Python, which we designed to be easily reusable and extensible.

### 2.2 BGP anomaly detection

BGP anomaly detection research has evolved through several approaches. Early works focused on detecting outliers in BGP message time-series statistics using techniques like wavelet analysis to identify sudden changes in message volume or inter-event times [31, 33, 38]. While these unsupervised, real-time methods are practical for deployment, they make strong assumptions that limit detection accuracy.

More recent supervised approaches feed multiple BGP attributes to pre-trained classifiers [12, 14, 21, 22]. However, they require extensive labeled data and struggle with previously unseen anomalies, necessitating costly retraining that breaks real-time constraints.

The unsupervised IODA pipeline [8] achieves state-of-the-art outage detection by monitoring AS reachability [8, 35]. Yet it remains insensitive to topology-altering anomalies like route leaks. Recent work shows that AS topology features enhance detection accuracy: graph-level features like spectral properties outperform statistical features for hijacking detection [25], while local graph metrics, such as centrality and eccentricity, prove beneficial [40].

Graph neural networks (GNNs) have emerged for automatic topological feature extraction, either through end-to-end training on labeled anomalies [30] or by generating node embeddings for supervised classifiers [23]. GNN architectures consistently outperform statistical methods.

Despite these advances, current graph-based approaches face three key limitations: (i) supervised modules requiring extensive labeled data that miss novel anomaly types; (ii) time-consuming feature extraction incompatible with real-time Internet-scale deployment; and (iii) neglect of AS temporal graph dynamics, which provides complementary information to static topological analysis. This work addresses these limitations through a comprehensive anomaly detection pipeline.

### 2.3 The MAD algorithm

We hypothesize that BGP anomalies can be better detected if both the graph and the temporal aspects of the Internet are taken into account. Thus, calling us to search for the anomalies directly on the Internet's temporal graph. The topic of anomaly detection in temporal graphs is a highly active research area in the domains of data mining and machine learning, with numerous approaches proposed in recent years [7, 20, 26, 28, 32, 43, 46]. In a nutshell,

[1] https://github.com/JustinLoye/quickrib

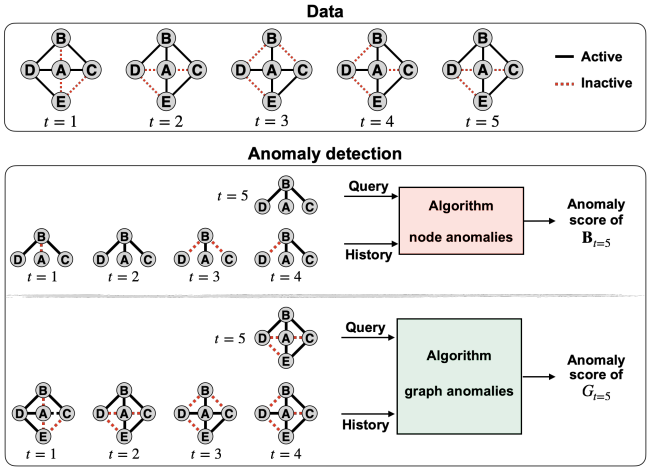



**Figure 1: Illustration of anomaly detection in temporal graphs. Algorithms can be seen as black-boxes that assess the anomaly level of a query structure based on its history.**

algorithms in this field can be seen as black-boxes that receive two inputs: (i) a query structure at a given time (node, edge, graph); and (ii) historical data. Then, they output an anomaly score assessing the extent to which the observed query can be explained from the past data, producing a larger score the less it can be explained. See Figure 1 for an illustration. As detailed in [4], algorithms proposed in the literature essentially differ in the types of queries, contexts, and anomaly definitions that they work with, with each algorithm corresponding to a particular combination.

While most algorithms for anomaly detection in temporal graphs are limited to handle queries of one specific type (either nodes, edges or graphs) and struggle to cope with unpredictable graph dynamics, the recent work of [4] proposed a more general methodology, coined Multi-Scale Anomaly Detection (MAD), that tackles the above two issues. Namely, it accepts arbitrary time-stamped subgraphs as queries, meaning that it contains edge, node, and graph anomaly detection as particular cases; and account for the data uncertainty through an scoring mechanism rooted in a probabilistic multi-scale analysis of temporal graphs, which labels a query as abnormal only when the system's inherent uncertainty cannot explain it.

Through an extensive empirical evaluation, the authors of [4] showed that their MAD method is not only more flexible in the types of queries it can assess, but is also much more flexible in the types of anomalies it can detect, significantly outperforming state-of-the-art alternatives without imposing any a-priori on the topology of the anomalies. These qualities make the MAD algorithm a very promising approach to address the BGP anomaly detection challenge. Yet, it suffers from one drawback that hinders its applicability in the BGP context: it was conceived in the context of binary temporal graphs, where edges can only be either active or inactive. In Section 3.2, we show that it is important to consider edge weights when modelling the Internet as a dynamic graph, as they help us to correct biases. Thus, to carry the success of MAD into the BGP context, it is necessary to adapt its scoring mechanism to cope with temporal graphs with weighted edges. This is a situation that we also address in this work.

## 3 Methodology

Figure 2 illustrates the overall architecture of MADBGP, our proposed pipeline for anomaly detection in BGP. It consists of three main steps: (i) collection of publicly available raw BGP data; (ii) inference of the Internet's temporal graph from the collected BGP data; and (iii) unsupervised and near real-time identification of local and global anomalies in the Internet's temporal graph. The rest of this section details these steps.

### 3.1 QuickRIB: BGP data processing

We analyze historical BGP data collected from BGP Route Collectors (RCs) where volunteer ASes connect to and act as Vantage Points (VPs) by providing their view of the Internet topology. Specifically, VPs provide two types of BGP data: *RIBs* and *updates*. RIBs are full dumps of VPs' routing tables and are only available at a low temporal resolution (8 hours for RIS), which is longer than most of the outages we consider. RIBs can be reconstructed at a higher temporal resolution by combining them with BGP updates, which are incremental high-frequency (ms) changes to the RIB. In order to streamline this process and efficiently handle the large amount of collected BGP data, we developed QuickRIB.

The goal of QuickRIB is to provide a Python framework for high-frequency Internet-wide analysis in near-realtime. Unlike previous tools that require to analyze entire RIBs at the end of each time window [24, 34], QuickRIB leverages the subject (RIB) - observer (analysis module) pattern [18] to incrementally compute the changes induced by each BGP update.

For example, our analysis module inferring Internet topology snapshots every five minutes only processes the number of BGP updates of the last five minutes. This is between two and three orders of magnitude less data to process than simple methods that batch process all the RIB entries (Table 7). Figure 3 illustrates the implementation of this module using QuickRIB. Different data structures, represented as tables, are required to achieve the hegemony computation (subsection 3.2): "Paths count" for the betweenness centrality, "IP space" for accounting Internet space deaggregation, and "Mean" to compute the trimmed mean. A computation step can either occur for every BGP update or at the end of each time window (here every five minutes).

QuickRIB minimizes computation at the end of each time window by updating data structures while processing BGP updates. In commonly used BGP tools [24, 34], BGP updates are only recorded in the RIB. Then, at the end of the time window, the entire RIB is processed to produce large data structure (here "IP space" and "Paths count") which is costly and redudant as the RIB is marginally changing over time. With QuickRIB, these two data structures are entirely computed only once, at the RIB construction time (warm start), and then incrementally updated at each BGP update. Therefore at the end of each time window, costly RIB iteration is avoided and fewer computation steps are performed.

QuickRIB is designed to be flexible, easy to reuse and to extend. QuickRIB's implementation consist of chained observers sharing a common interface (a BGP update and optional data) that can

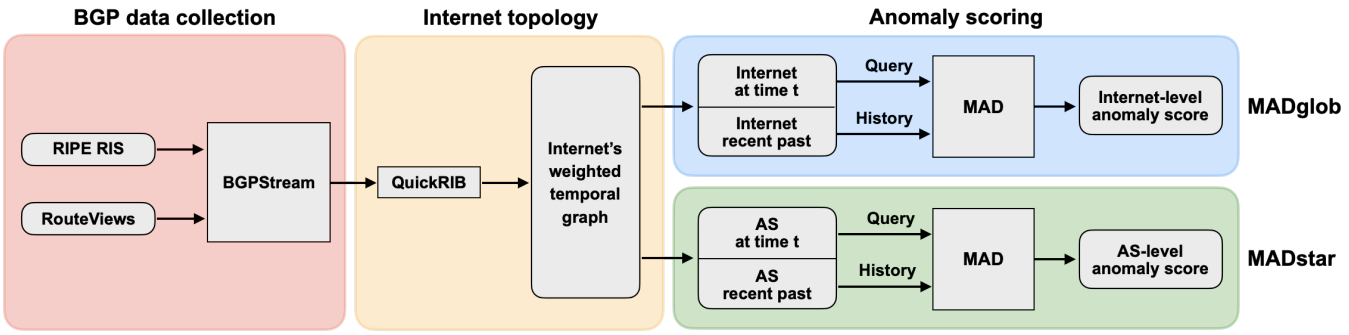



**Figure 2: The MADBGP pipeline for anomaly detection in BGP.**

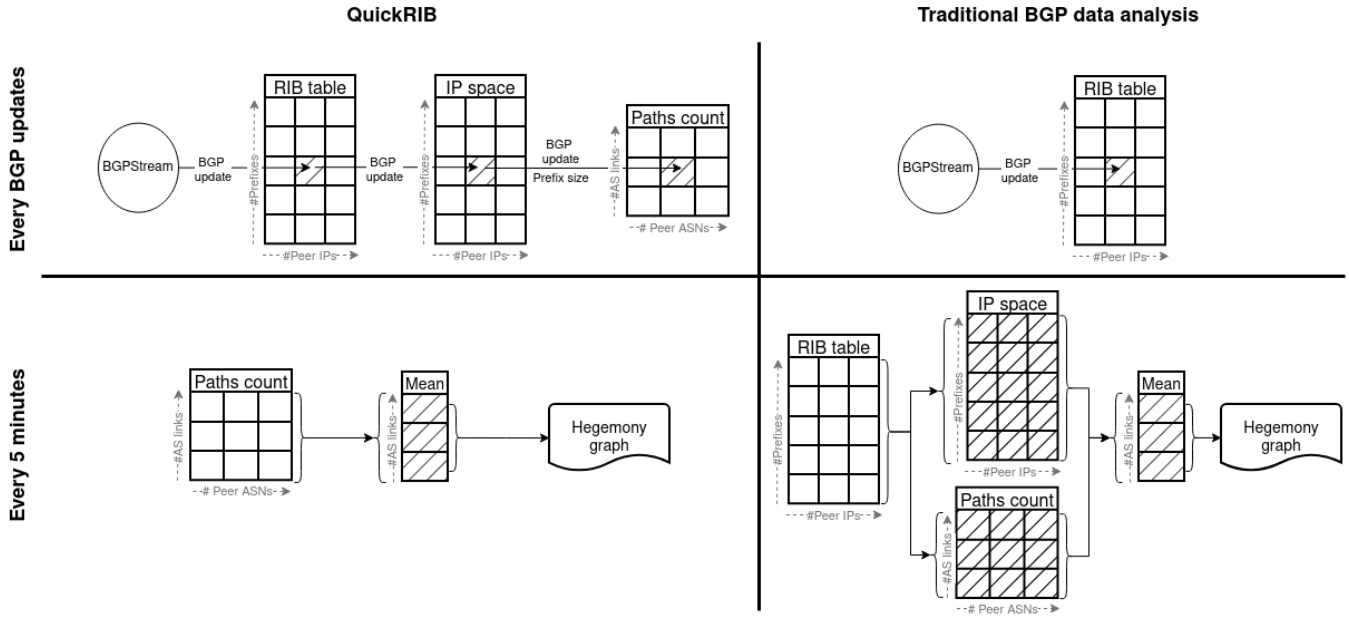



**Figure 3: Illustration of hegemony-topology computation using QuickRIB or traditional BGP data-analysis tools. The tools operate at two time scales: (i) on each BGP update to track topology changes, and (ii) at each analysis resolution step (every 5 minutes) to compute the full topology. Each hatched cell represents a modified data point (result of a computation step). The sizes of the data structures are annotated with gray arrows; typical values are: #Prefixes = $10^6$, #Peer IPs = $10^2$, #Peer ASNs = $10^2$, and #AS Links = $2 \times 10^5$.**

be reused in other applications. For example, the "IP space" observer of Figure 3 could provide data to another observer or be used stand-alone. Similarly, the "Paths count" module could use another observer to weight its paths. An observer requires defining three functions that react to possible changes in the RIB: i) process a new RIB entry (RIB construction), ii) process an update announcement and iii) process an update withdrawal. These three functions share a generic interface which makes QuickRIB observers compatible with any stream of BGP data. In our experiments, we use the popular PyBGPStream library that transforms RIPE RIS and RouteViews archive data to BGP stream. We also implemented our own BGP stream provider called pybgpflux, using the data broker and MRT parser from BGPKIT [47], and verified that results are consistent with PyBGPStream. We make both QuickRIB[2] and pybgpflux[3] available to the community.

## 3.2 Internet's temporal graph modeling

We pay particular attention to the graph modeling of BGP data. In the literature [23, 40], topologies are constructed from AS adjacency links in AS-paths, which are then weighted by the number of observed paths, or the size of the associated prefix. However, we stress that the resulting topologies from these procedures are not fully representative of the Internet. Indeed, BGP data is heavily biased toward VPs, which appear on all the paths they report, thus

[2]https://github.com/JustinLoye/quickrib
[3]https://github.com/JustinLoye/pybgpflux

artificially inflating their importance in the global Internet structure. Moreover, the procedures described above are prone to noise, as VPs frequently disconnect and reconnect to the RC, causing their entire routing table to be updated.

We propose to address the above issues through the notion of hegemony graphs: a new AS-level topology modelisation that leverages the AS-hegemony metric as a mean to weight the links of the Internet graph. The AS-hegemony metric was introduced in [16] as a robust and unbiased measure of the importance of ASes in the BGP topology. It computes the so-called betweenness centrality for each VP, independently, and then takes the average while excluding extreme values caused by VPs biased toward a given AS.

In formal terms, it is defined as follows: let $n$ be the total number of VPs and $2\alpha$ be the ratio of discarded VPs considered to be biased. Then, the AS hegemony of an AS $v$ is defined as:

$$W(v,\alpha) = \frac{1}{n - (2\lfloor \alpha n \rfloor)} \sum_{j=\lfloor \alpha n \rfloor + 1}^{n-\lfloor \alpha n \rfloor} BC_{(j)}(v), \tag{1}$$

where $\lfloor \cdot \rfloor$ is the floor function, and $BC_{(j)}(v)$ is the betweenness centrality value computed using the paths bound to VP $j$ and passing through AS $v$, where it assumed that such values are arranged in ascending order so that $BC_{(1)}(v) \leq BC_{(2)}(v) \leq \cdots BC_{(n)}(v)$.

In this work, we exploit this metric to measure the importance of adjacency links in the AS-PATH as realistically as possible. We do it by replacing the term $BC_{(j)}(v)$ with

$$BC((u,v)) = \frac{\sigma(u,v)}{\sigma}, \tag{2}$$

which refers to the betweenness centrality of edge $(u,v)$, where $\sigma(u,v)$ is the number of BGP paths passing through the AS link $(u,v)$, and $\sigma$ refers to the total number of BGP paths. The computation of all adjacency links betweenness centrality illustrates QuickRIB ability to efficiently process BGP data. The straightforward approach is to enumerate all AS paths of the RIB and accumulate the counts $\sigma(u,v)$ and $\sigma$. Instead, QuickRIB introduces a new data structure to maintain these counts incrementally as BGP updates arrive, avoiding the costly RIB iteration.

As proposed in [16], we account for the disparity between paths by weighting them during the BC computation based on the amount of associated IP space. Instead of simply using the network prefix length as in [23], we also consider Internet address space deaggregation [11, 17] to model more accurately the distribution of advertised address space.

## 3.3 Anomaly scoring

The output of our QuickRIB procedure is a weighted temporal graph. We model it as a function $L : T \times V \times V \to \mathbb{R}$, where $V$ denotes the set of nodes (ASes), $T \subset \mathbb{N}$ is the set of observed times, and $L(t,u,v) = W((t,u,v),\alpha)$ represents the AS hegemony value of the link between $u$ and $v$ at time $t$. By convention, we set $L(t,u,v) = 0$ if the link was not handled by QuickRIB. Based on this temporal graph, the question that we aim to address is: how can we assess, in real-time and in an unsupervised manner, whether the data observed at time $t$ is abnormal? To tackle this question, we propose to leverage the state-of-the-art MAD anomaly detection algorithm [4], which (a) is unsupervised; (b) runs in linear time with respect to the number of analyzed time-stamped edges, or, in other words, if there are $M$ edges to analyze during $N$ times it runs in $O(NM)$; (c) can monitor structures at multiple scales; and (d) has been shown to detect a rich class of anomaly types with superior performance compared to state-of-the-art alternatives. Yet, as discussed in Section 2.3, MAD was originally proposed for binary temporal graphs. Thus, we extend the MAD methodology to weighted temporal graphs.

To formalize our extension, let us denote the space of possible relations between ASes by $\mathcal{E} = V \times V$ and an arbitrary subset of them by $\phi \subseteq \mathcal{E}$. Let us also define a query as the subgraph induced by $\phi$ at time $t$, which we denote $Q : t \times \phi \to \mathbb{R}$, and by $\mathcal{H} : [t-N, t-1] \times \phi \to \mathbb{R}$ we refer to the $N$-length immediate history of the query. MAD is thus an algorithm that evaluates the anomaly level of $Q$ based on a normality reference extracted from $\mathcal{H}$. MAD operates as follows: it uses $\mathcal{H}$ to establish a relative ranking of edges according to their probability of appearance. Then, it assumes that $Q$ is normal if its edges preserve such ranking. MAD quantifies this through a specialized data structure that measures at multiple resolution levels how consistent the edge rankings in $Q$ are with those observed historically, assigning higher anomaly scores to queries that deviate the most. In [4], this is only developed for binary queries $Q : t \times \phi \to \{0,1\}$, giving MAD a probabilistic interpretation leveraged to automatically identify the optimal window length $N$. In the extension that we develop next, we mirror the developments of [4] but we strip-out the binary constraint. We stress that this comes at the price of losing the probabilistic interpretation, which prevents us from accessing the stationarity test used in [4] to identify the optimal window length. Yet, despite this limitation, we demonstrate in Proposition 1 that our weighted extension preserves the main asset of the original MAD method: it encodes the data into a minimal set of highly informative coefficients for anomaly detection that do not involve any loss of information.

Therefore, we define weighted MAD as the following four-step procedure: (i) setup of a binary-tree data structure for anomaly detection; (ii) analysis of $Q$ via the data structure; (iii) analysis of the $\mathcal{H}$ via the data structure; and (iv) anomaly scoring by comparing the analyses of $Q$ and $\mathcal{H}$. We describe these steps next.

**Data structure.** The first step in constructing the data structure consists in aggregating the values of $\mathcal{H}$ over the time axis as

$$f(e_i) = \sum_{k=t-N}^{t-1} \mathcal{H}(k, e_i), \tag{3}$$

which we do for all $e_i \in \phi$. Then, we use $f$ to sort the elements of $\phi$ in decreasing order of their value in $f$. This is, we attribute the indices $i$ of $e_i \in \phi$ so that, $f(e_1) \geq \cdots \geq f(e_i) \geq f(e_{i+1}) \geq \cdots \geq f(e_M)$, where it is assumed that $|\phi| = M$. This allows us to rank the edges based on their relative importance in the recent past, measured in terms of AS hegemony. Based on this ranking, we construct a binary-tree data structure for anomaly detection. See Figure 4 for an illustration. This data structure associates each $e_i \in \phi$ to a leaf node of the tree, while respecting the order: the left-most leaf node corresponds to $e_1$ and the right-most leaf node corresponds to $e_M$. For mathematical convenience, we assume that $M$ is a power of 2. This can be easily enforced in practice by adding leaf nodes to the right of the tree, associated to virtual links. For the sake of notation lightness, we associate each node of the tree to the set of leaf nodes

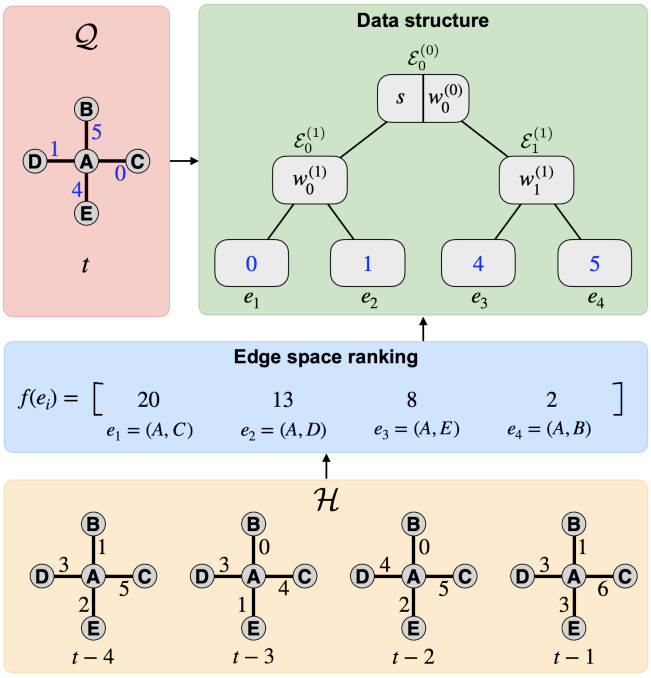



**Figure 4: Illustration of the binary tree data structure for anomaly detection.**

it covers: the $k$-th node at level $\ell$ covers the following set of leaf nodes:

$$\mathcal{E}_k^{(\ell)} = \left\{ e_i \in \phi : \frac{kM}{2^\ell} + 1 \leq i \leq \frac{(k+1)M}{2^\ell} \right\} \tag{4}$$

where $\mathcal{E}_0^{(0)} = \phi$ is the root node and $\mathcal{E}_0^{(\log_2(M))} = e_1$ refers to the left-most leaf. The MAD algorithm leverages this data structure to compute a set of coefficients upon which anomalies are assessed.

**Query analysis.** We analyze $\mathcal{Q}$ via the data structure by setting the values of the leaves with those of $\mathcal{Q}$. This is, the leaf node associated to $e_i$ is set with value $\mathcal{Q}(t, e_i)$. Then, for each other node $\mathcal{E}_k^{(\ell)}$ in the tree, we compute the following coefficient:

$$w_k^{(\ell)}(t) = \frac{\sqrt{2^\ell}}{\sqrt{M}} \left[ \sum_{e_i \in \mathcal{E}_{2k}^{(\ell+1)}} \mathcal{Q}(t, e_i) - \sum_{e_j \in \mathcal{E}_{2k+1}^{(\ell+1)}} \mathcal{Q}(t, e_j) \right] \tag{5}$$

Notice that the summations run over $\mathcal{E}_{2k}^{(\ell+1)}$ and $\mathcal{E}_{2k+1}^{(\ell+1)}$, where the former contains edges that are always ranked higher than those of the latter in the tree. Thus, the coefficients $w_k^{(\ell)}(t)$ essentially add and compare the AS hegemony values of two groups: $\mathcal{E}_{2k}^{(\ell+1)}$, which correspond to edges that historically had larger hegemony value; and $\mathcal{E}_{2k+1}^{(\ell+1)}$, which corresponds to edges that historically had smaller hegemony value. Interestingly, these coefficients have the asset of being extremely sensitive to sudden changes in the relative rank of relations. Indeed, notice that a negative value in $w_k^{(\ell)}(t)$ would be indicative that the edges in $\mathcal{E}_{2k+1}^{(\ell+1)}$ now have, on average, more hegemony value than those in $\mathcal{E}_{2k}^{(\ell+1)}$. However, this would be in total contradiction with what was seen in $\mathcal{H}$, that led $\mathcal{E}_{2k}^{(\ell+1)}$ to be ranked higher than $\mathcal{E}_{2k+1}^{(\ell+1)}$. This is the situation illustrated in Figure 4, where the AS hegemony values of the query do not follow the rank of edges in the history. In addition to being sensitive to rank changes, the tree coefficients are interesting because they allow us to assess the severity of an anomaly based on the level at which it manifests in the tree. Namely, a very drastic anomaly in which edges $e_1$ and edge $e_M$ suddenly switch roles: being now $e_M$ the most important and $e_1$ the less important, would only show in the coefficient associated to the root $w_0^{(0)}(t)$, while anomalies that involve less drastic rank switches appear in lower parts of the tree. This adds a layer of interpretability that can be used to characterise the signature of specific kinds of anomalies, as done in [4], or to filter-out anomalies by severity level.

The main result of [4] is the theoretical demonstration that the coefficients $w_k^{(\ell)}(t)$, accompanied by an additional coefficient

$$s(t) = \frac{1}{\sqrt{M}} \sum_{e_i \in \phi} \mathcal{Q}(t, e_i), \tag{6}$$

do not involve any loss of information about $\mathcal{Q}$. This is, $\mathcal{Q}$ can be fully reconstructed from its coefficients $s(t)$ and $w_k^{(\ell)}(t)$. Proposition 1 shows that this result remains valid in the case of weighted temporal graphs.

Proposition 1. *Let $\mathcal{Q}$ be an arbitrary weighted time-stamped sub-graph and let $\left\{ w_k^{(\ell)}(t), s(t) \right\}$ be the set of coefficients as defined in (5) and (6), respectively. Moreover, let $k(\ell, i) = \lfloor \frac{2^{\ell+1}}{M}(i-1) \rfloor$, where $\lfloor \cdot \rfloor$ refers to the floor function. Then, it holds that:*

$$\mathcal{Q}(t, e_i) = \frac{1}{\sqrt{M}} s(t) + \sum_\ell \frac{\sqrt{2^\ell}(-1)^{k(\ell,i)}}{\sqrt{M}} w_{\lfloor k(\ell,i)/2 \rfloor}^{(\ell,i)}(t) \tag{7}$$

**History analysis.** We assess if the observed value of $\mathcal{Q}$, and consequently the computed coefficients $w_k^{(\ell)}(t)$ and $s(t)$, constitute an anomaly based on the values observed in $\mathcal{H}$. In more precise terms, we use $\mathcal{H}$ to estimate the first statistical moments of $w_k^{(\ell)}(t)$ and $s(t)$ in order to assess whether the observed values of $w_k^{(\ell)}(t)$ and $s(t)$ are highly likely or not to be seen. While in [4] the expectation and variance of both $w_k^{(\ell)}(t)$ and $s(t)$ were computed theoretically and in closed form, this was possible due to the unweighted graph assumption. Here, we must estimate the mean and the variance empirically from $\mathcal{H}$. In order to do it, we simply apply the data structure to all the graph snapshots contained in $\mathcal{H}$, allowing us to obtain a time-series for each coefficient $w_k^{(\ell)}$ and $s$. Then, we simply estimate the sample mean and the sample standard deviation for each of such time series, producing $mean\left( w_k^{(\ell)}(t-N:t-1) \right)$ and $\sigma(w_k^{(\ell)}(t-N:t-1))$, respectively, where $w_k^{(\ell)}(t-N:t-1)$ refers to the time-series of the coefficient used for the estimation.

**Anomaly scoring.** We assume that the coefficients produced by a normal $\mathcal{Q}$ can be very well explained from the mean and the variance of the underlying random process, which we estimated from $\mathcal{H}$. This is, that the coefficients produced by $\mathcal{Q}$ are not far from the mean and remain well confined within the variance of the process. Otherwise, we assume that $\mathcal{Q}$ is abnormal. We formalize this idea by computing an anomaly score for each individual

coefficient via the Z-score:

$$\text{score}(w_k^{(\ell)}(t)) = \left| \frac{w_k^{(\ell)}(t) - mean\left(w_k^{(\ell)}(t-N:t-1)\right)}{max\left\{\epsilon, \sigma\left(w_k^{(\ell)}(t-N:t-1)\right)\right\}} \right|, \quad (8)$$

with $\epsilon$ a dampening parameter to avoid division by zero.

Then, in order to compute a single anomaly score for the whole query sub-graph, we aggregate the anomaly score of all coefficients:

$$\text{score}(Q) = \text{score}(s) + \sum_{k,\ell} \text{score}(w_k^{(\ell)}) \quad (9)$$

For simplicity, we give equal importance to the anomaly scores of each coefficient, regardless if its scale. Yet, we stress that it is possible to produce more complex anomaly scores that focus on anomalies at specific levels of the data structure: for instance, by truncating the above sum to a maximum value of $\ell$, then we would only focus on the more drastic anomalies that appear at the higher levels of the data structure. It would also be possible to define the above sum by giving different importance to the coefficients of different scales. We leave the investigation of such more complex anomaly scoring mechanisms as future work.

**The MADglob and MADstar algorithms**. Notice that the anomaly detection methodology derived above is general and holds for any time-stamped query sub-graph $Q$. While many topologies are worth investigating in depth, two specific kinds of sub-graphs are of particular interest for BGP anomaly detection: the whole graph and star-like sub-graphs centered at a node. Observe that the former offers a global view of the Internet where generalized outages must be visible, while the latter offers a detailed perspective in which localized anomalies at the AS scale must be visible. Thus, both query types are complementary and offer a multi-scale view of Internet. As a result, we derive two anomaly detection algorithms that we will investigate in our experimental section: **MADglob** and **MADstar**, which refer to the specific cases in which $Q = t \times \mathcal{E}$ and $Q = t \times u \times V$, respectively. We implement both MADglob and MADstar as QuickRIB observer.

## 4 Evaluation

In this section, we validate MADBGP accuracy and robustness on both historical and recent BGP anomalies. Our experiments aim to address the following questions:

- (Q1) How effective is MADglob in detecting Internet-level anomalies?
- (Q2) Does MADglob benefit from hegemony graphs compared to a topology built from AS adjacency links?
- (Q3) How robust is MADglob with respect to the choice of its hyperparameters?
- (Q4) How does MADstar compare to IODA's BGP results?
- (Q5) Can MADstar pinpoint the most likely impacted ASes during an abnormal event?

### 4.1 Experimental setup

We evaluate MADBGP on the incidents listed in Table 1. It includes four scenarios widely studied in the literature [23, 40]: `aws`, `indosat`, `tm`, `ttnet`, for which we use the same route collectors as past studies. It also includes six more recent scenarios for which the outage start and end times, as well as the ASes involved, are well documented [10, 19, 27, 39, 41, 42]. For these six scenarios, we obtain BGP data from a mix of global and large RIS collectors in order to increase the number of VPs and therefore reduce the bias toward them. To ensure that our results do not depend on the BGP stream provider (PyBGPStream) and data collection project (RIS), we also run experiments with PyBGPKITStream and RouteViews collectors and observe similar results (see Figure 10 in subsection C.4).

| Scenario | Year | Start | End | Collectors | Outage start | Dur. |
|---|---|---|---|---|---|---|
| `aws` [29] | 2016 | 04-20 16h | 04-24 19h | rrc04 | 04-22 17:10 | 1:55 |
| `indosat` [44] | 2014 | 03-31 16h | 04-05 00h | rrc04 | 04-02 18:25 | 2:30 |
| `tm` [45] | 2015 | 06-10 08h | 06-14 16h | rrc04 | 06-12 08:43 | 3:02 |
| `ttnet` [37] | 2004 | 12-22 08h | 12-27 00h | rrc05 | 12-24 09:20 | 10:27 |
| verizon [41] | 2019 | 06-22 08h | 06-26 16h | rrc00,01 | 06-24 10:30 | 2:00 |
| allegheny | | | | rrc03,12 | | |
| `iran` [27] | 2024 | 07-09 00h | 07-13 16h | rrc00,01 | 07-10 11:00 | 3:20 |
| | | | | rrc03,12 | | |
| china [10] | 2019 | 06-04 08h | 06-08 16h | rrc00,01 | 06-06 09:43 | 2:00 |
| europeanmobile | | | | rrc03,12 | | |
| `mainone` [19] | 2018 | 11-11 16h | 11-16 00h | rrc00,01 | 11-13 21:13 | 2:00 |
| | | | | rrc03,12 | | |
| pakistan [39] | 2008 | 02-22 16h | 02-27 00h | rrc00,01 | 02-24 18:10 | 2:14 |
| youtube | | | | rrc03,12 | | |
| `italy` [42] | 2023 | 02-03 00h | 02-08 00h | rrc00,01 | 02-05 11:00 | 5:00 |
| | | | | rrc03,12 | | |

**Table 1: Evaluation scenarios and data sources.**

We set our experiments to begin at least 2 days before each anomaly and extend at least 2 days after, thereby recreating realistic BGP monitoring conditions where anomalies (typically lasting a few hours) occupy a small fraction of the total monitoring period. Following [40], we used a 5-minute resolution to ensure sufficient data points during the anomaly period. This results in Internet graphs that, in recent scenarios, contain approximately 80,000 ASes (nodes) and 200,000 edges at each timestamp.

Concerning the implementations of MADglob and MADstar, we truncate the tree data structures to only represent edges $e_i$ in the leaves that have activity in $\mathcal{H}$ or in $Q$. This is, edges that satisfy $\mathcal{H}(t-n, e_i) > 0$ for some $n$ or $Q(t, e_i) > 0$. This is because inactive edges do not contribute to the anomaly scores, yet excluding them from the data structure improves computational efficiency. Our experiments with a commodity server demonstrate MADBGP's real-time capabilities. For the most demanding case MADBGP takes on average less than 3 minutes to compute 5-minute snapshots (see details in Appendix C).

### 4.2 Evaluation Metrics

We compare MADBGP's anomaly scores to ground-truth labels derived from the outage time period given in Table 1 using standard binary classification metrics. Since MAD outputs continuous scores rather than binary labels, in practice we must set a threshold that flags scores as abnormal if they are above it, enabling the computation of a confusion matrix and performance metrics. Choosing such a threshold is a difficult and application-dependent task: for instance, a low threshold catches more anomalies but at the cost of allowing more false alerts, which may benefit situations where missing anomalies is catastrophic; while a more selective threshold may useful in situations where false alerts are costly. Therefore,

we employ the performance metrics ROC AUC and PR AUC [13], which do not evaluate the quality of decisions based on a specific threshold, but rather assess the distribution of the scores assigned to the positive and negative samples.

Let $TP$, $FN$, $FP$, $TN$ stand for True Positive, False Negative, False Positive, True Negative respectively. The ROC AUC metric operates as follows; it starts setting the lowest possible threshold, yielding the maximum True Positive Rate, $TPR = TP/(TP + FN)$, but also the maximum False Positive Rate, $FPR = FP/(TN + FP)$, as everything is flagged as an anomaly. The threshold is then progressively increased to observe the evolution of $TPR$ and $FPR$. If the detector is accurate, the score given to negative samples will make them fall below the threshold first, allowing the $TPR$ to remain high while the $FPR$ decreases. If the two decrease at the same rate, it is a sign that positive samples are given similar scores to negative ones. The ROC AUC value corresponds to the area under the curve formed by plotting $TPR$ against $FPR$: a perfect detector leads to an area of 1, while a random one leads to an area of 0.5. Interestingly, ROC AUC also admits a probabilistic interpretation: it corresponds to the probability that a randomly chosen positive sample is assigned a higher score than a randomly chosen negative one.

Despite its relevance, ROC AUC is not fully satisfactory in imbalanced scenarios. Consider a case with 100 positive samples and 10,000 negative ones, evaluated by a detector that ranks all positive samples within the top 200 scores. Such a detector would achieve a high ROC AUC, since 99% of the negative samples receive lower scores than the positive ones. However, if we focus only on the top 200 ranked samples, and the positive and negative instances are intermixed within this subset, a practitioner would observe roughly as many false positives as true positives. Thus ROC AUC may not fully reflect detection accuracy from the practitioner perspective. The PR AUC metric is specifically designed to provide more insights in this situation by evaluating how accurately the positive samples are identified among all predictions. It does so by also varying the decision threshold, but this time monitoring the evolution of two metrics that depend only on the positive samples: Precision $P = TP/(TP + FP)$, which measures how many of the identified anomalies are truly anomalies, and Recall $R = TP/(TP+FN)$, which measures how many of the positive samples are correctly identified.

While a perfect detector still yields a PR AUC of 1, a random one does not correspond to a fixed value. Instead, the PR AUC of a random detector equals the proportion of positive samples in the dataset: in our case, this value oscillates around 0.03 (see Table 1) and it corresponds to the ratio of anomalous time stamps to the total number of time stamps. In sum, ROC AUC and PR AUC should be viewed as complementary: ROC AUC quantifies the extent to which the algorithm reduces the search space in which anomalies are likely to occur, whereas PR AUC evaluates how well true anomalies are ranked within that reduced subset.

## 4.3 Hyper-parameter optimization

MADBGP has two main parameters: the window size $N$ and $\epsilon$ (Equation 8). We study the hyper-parameter space via hyper-parameter optimisation (HPO), which explores the space for the configuration that gives the maximum PR AUC value. For that we use the state-of-the-art "Tree-structured Parzen Estimator" [6], implemented in the Optuna library [2]. The algorithm requires the bounds of the parameter space and the number of trials to be specified. We choose generous bounds for continuous parameters: parameter $N$, the window length, is sampled between $[15, 180]$, which correspond to durations between [75 minutes, 15 hours]; and the damping parameter $\epsilon$ is sampled in the log domain within the interval $[10^{-12}, 10^{-2})$. The search is based on 50 trials.

Since the Z-score (see Equation 8) can be sensitive to outliers, we also investigate the modified Z-score "mZ" as scoring mechanism, since it tends to be more robust to outliers. It is defined as

$$\mathrm{mZ}(w_k^{(\ell)}) = 0.6745 \left| \frac{w_k^{(\ell)} - median(w_k^{(\ell)}[t-N, t-1])}{max\left\{\epsilon, MD\left(w_k^{(\ell)}[t-N, t-1]\right)\right\}} \right|, \quad (10)$$

where $MD$ refers to the median absolute deviation. Thus, we add the choice of scoring method, Equation 8 or Equation 10, as an extra categorical parameter in the HPO.

In order to speed up the HPO search, we develop a fast approximation of MAD referred to as "MADstatic". This comes out of necessity, as even if MAD is highly efficient, the combinatorial nature of the HPO makes the computation time explode. Namely, even if MAD analyzes 5 days of BGP data in 3 days of computational time (Table 7), the HPO of 50 trials applied to 4 scenarios of 5 days each, would take 600 days to perform the experiments sequentially. Our MADstatic approximation assumes that the relative ranking of AS hegemony values is stable over time, implying that the coefficients $w_k^{(\ell)}$ can be only computed once per time-stamp, for all queries. The coefficients of the graphs in $\mathcal{H}$, necessary for the estimation of the the statistical moments in Equation (8), coincide with those computed from previous queries. This re-use of previous queries is not possible when the ranking is reset at each time stamp, forcing us to analyze all $\mathcal{H}$ at each time step. Formally, we achieve it by fixing the data structure only once at $t = 0$ with $\tilde{f}(e_i) = Q(0, e_i)$ instead of Equation 3. We stress that this approximation of MAD has the effect of ignoring any links that were not present at $t = 0$.

## 4.4 Internet-level anomaly detection

In this subsection, our ambition is threefold: we aim to tackle Q1, concerning the effectiveness of MADglob; Q2, concerning the impact of hegemony graphs; and Q3, concerning the robustness to hyper-parameters. To assess Q2, we repeat the HPO for several temporal graphs that we construct via two Internet models. The first model we call `graph` is commonly found in the litterature [23, 40]. It consists in building the topologies from AS adjacency links in AS-paths. We consider two variants based on two weighting mechanisms: (i) `paths_count`: the number of BGP paths going through the adjacency link (similar to [40]); and (ii) `ips_count`: the sum of the associated prefix sizes without taking care of the prefix space deaggregation (similar to [23]). The second Internet model that we consider is the one derived in subsection 3.2 and to which we refer as `hegemony`. For this case, we consider temporal graphs obtained with different ratio of discarded VPs $\alpha \in [0, 0.1, 0.2, 0.3, 0.4]$.

Table 2 shows the performance of MADstatic when run on the temporal graphs produced by the models above and when the hyper-parameters have been optimized for optimal performance. In almost all scenarios, MADstatic achieves very good performance with a

near perfect PR AUC and ROC AUC. The only exception is `indosat`, which achieves a near perfect ROC AUC but a less strong PR AUC of 0.3, which is nonetheless one order of magnitude better than the random baseline of 0.03. If we compare `graph` and `hegemony`, in general, the latter improves the detection, providing equal or better ROC AUC for all scenarios and similar or better PR AUC, particularly on `ttnet`. Experiments for recent scenarios are deferred to the appendix (see Table 5), where a similar trend can be observed. In particular, for 5 scenarios (`china europeanmobile`, `iran`, `italy`, `mainone`, `verizon`) the hegemony graphs significantly enhance the PR AUC and sustain or improve the ROC AUC. Interestingly, neither modelisation was able to detect the `pakistan youtube` event. Both, Table 2 and Table 5, suggest that performance is not drastically affected by the parameter $\alpha$.

In Figure 5, we provide a detailed analysis of MADglob's performance on the best previously identified topologies and the best HPO trial parameters from Table 2. We see that MAD and MADstatic perfectly identify the anomaly in the `aws` and `tm` scenarios, as they both produce a prominent spike in the anomaly score during the anomalous period. In the case of `indosat`, both MAD and MADstatic achieved a good ROC AUC but only a moderate PR AUC. This indicates that high scores are produced during the anomalous period and low scores during most of the normal period, yet some subset of normal times are also receiving large scores. This can be confirmed in the score plot of Figure 5, where it can be seen that high peaks appear before the event. For `ttnet`, we observe that MAD yields recurrent (daily) spikes in the anomaly score. These peaks suggest the emergence of new, short-lived edges that join the data structure for the first time. Since new edges are not represented in $\mathcal{H}$, they cannot be explained at all and thus receive a high anomaly score. In contrast, MADstatic is designed to ignore newly formed links, which explains its insensitivity to these events.

The above results highlight the ability of MADglob to effectively detect anomalies with best parameter settings. Yet, from the practical perspective, it is important to study its robustness to the choice parameters. We do so in Table 3, which reports the sensitivity of ROC AUC (left) and PR AUC (right) to variations in $\epsilon$ and $N$ across 50 HPO trials. Overall, it can be seen that PR AUC is more sensitive than ROC AUC, even though both metrics show a region around the optimal configuration where performance remains stable. Notably, this region is particularly broad for `aws` and `tm`, where ROC AUC stays stable over almost the entire parameter space, and PR AUC declines only when $\epsilon$ deviates by several orders of magnitude from its optimal value. For `indosat` and `ttnet`, the stable regions are much narrower, yet they still span roughly one order of magnitude. Our HPO also found that the choice of Equation 8 and Equation 10 has no impact on the detection performance.

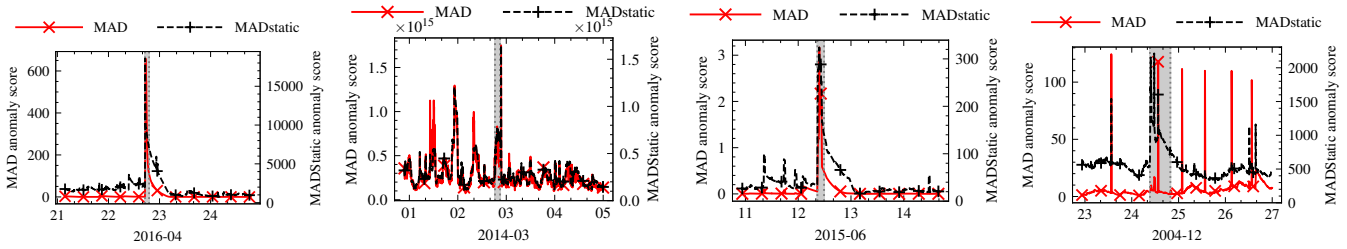
An interesting observation from Table 3 is that three out of four scenarios (`aws`, `tm`, `ttnet`) have their optimal parameters in nearby regions of the space, raising the question of whether it would be possible to leverage the HPO of one scenario in order to detect the others. To investigate this, we perform cross-validation between scenarios: the best parameters obtained for a scenario are tested on the other scenarios. For performance reasons, we rely on MADstatic. The resulting cross-validation matrix is shown in Figure 6. Columns represent the tested scenario and rows the scenario from which the hyper-parameters were taken from. In terms of ROC AUC, it can be seen that the HPO of one scenario, in general, tends to generalize well to the remainder of scenarios (with the notable exception of `pakistan_youtube` for which MADglob does not perform well). In the case of PR AUC, the situation is less stable and only `tm` and `aws` succeeded. Thus, it can be concluded from this experiment that MADglob is, in general, an effective tool to reduce the set of times where anomalies are likely to be found, yet within the top ranked time-stamps there can be an important amount of false positives if the parameters are not well chosen.

| Temporal graph | | | Detector performance | | | |
|---|---|---|---|---|---|---|
| | | | MAD | | MADstatic | |
| Scenario | Graph type | Weight param | PR AUC | ROC AUC | PR AUC | ROC AUC |
| `aws` | graph | ips_count | | | 0.93 | 0.97 |
| | | paths_count | | | 0.27 | 0.96 |
| | **hegemony** | **0** | 0.96 | 0.98 | **0.96** | **0.99** |
| | | 0.1 | | | 0.96 | 0.99 |
| | | 0.2 | | | 0.96 | 0.99 |
| | | 0.3 | | | 0.96 | 0.99 |
| | | 0.4 | | | 0.96 | 0.99 |
| `indosat` | **graph** | **ips_count** | 0.16 | 0.92 | **0.31** | **0.96** |
| | | paths_count | | | 0.24 | 0.92 |
| | hegemony | 0 | | | 0.29 | 0.96 |
| | | 0.1 | | | 0.27 | 0.96 |
| | | 0.2 | | | 0.28 | 0.96 |
| | | 0.3 | | | 0.30 | 0.96 |
| | | 0.4 | | | 0.24 | 0.95 |
| `tm` | graph | ips_count | | | 1.00 | 1.00 |
| | | paths_count | | | 0.96 | 1.00 |
| | **hegemony** | 0 | | | 1.00 | 1.00 |
| | | 0.1 | | | 1.00 | 1.00 |
| | | **0.2** | 0.99 | 1.00 | **1.00** | **1.00** |
| | | 0.3 | | | 1.00 | 1.00 |
| | | 0.4 | | | 1.00 | 1.00 |
| `ttnet` | graph | ips_count | | | 0.52 | 0.72 |
| | | paths_count | | | 0.76 | 0.83 |
| | **hegemony** | 0 | | | 0.95 | 1.00 |
| | | 0.1 | | | 0.94 | 1.00 |
| | | 0.2 | | | 0.92 | 1.00 |
| | | **0.3** | 0.10 | 0.43 | **0.95** | **1.00** |
| | | 0.4 | | | 0.95 | 1.00 |

**Table 2: MADglob best HPO trial performance metrics for temporal graphs generated from different scenario and Internet modelisations. A random detector has a ROC AUC of 0.5 and a PR AUC of 0.03, 0.03, 0.02 and 0.11 respectively for the scenarios `aws`, `indosat`, `tm` and `ttnet`.**

## 4.5 AS-level anomaly detection

In this subsection, we address Q4 and Q5, which concern the evaluation of MADstar and its capacity to pinpoint affected ASes, respectively. To evaluate MADstar, we benchmark it against IODA's BGP results.




(a) `aws` (b) `indosat` (c) `tm` (d) `ttnet`

**Figure 5: Anomaly scores for best identified Internet modelisation and best HPO trial parameters (Table 2). Ground-truth anomaly times are highlighted in gray.**

| Scenario | HPO trials in the search space ROC AUC($N$, $\epsilon$) | HPO trials in the search space PR AUC($N$, $\epsilon$) |
|---|---|---|
| `aws` | $\epsilon$: 6.93e-06, $N$: 138, Z-score | $\epsilon$: 6.93e-06, $N$: 138, Z-score |
| `indosat` | $\epsilon$: 9.42e-10, $N$: 66, mZ-score | $\epsilon$: 9.42e-10, $N$: 66, mZ-score |
| `tm` | $\epsilon$: 1.22e-03, $N$: 174, Z-score | $\epsilon$: 1.22e-03, $N$: 174, Z-score |
| `ttnet` | $\epsilon$: 4.50e-05, $N$: 173, Z-score | $\epsilon$: 4.50e-05, $N$: 173, Z-score |

**Table 3: Impact of MADglob hyperparameters on ROC AUC (left column) PR AUC and (right column).**

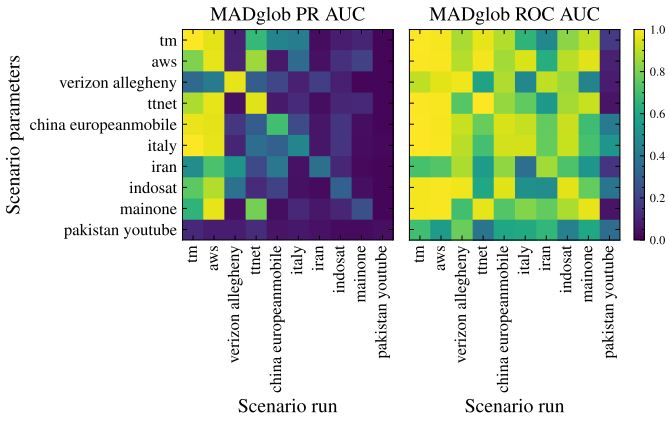



**Figure 6: Hyperparameters cross-validation. Each scenario best HPO parameters is tested on the other scenarios.**

Our experimental setup is as follows: we compute anomaly scores for all ASes for all scenarios via MADstar. To fix the hyperparameters, we use the ground truth label of an AS that has been reported in the literature to have had an anomaly, and run an HPO of 50 trials so as to maximize PR AUC detection of that AS. For performance reasons, we rely on MADstatic for the HPO. Then, the best identified parameters are used to compute the anomaly scores for the remainder of ASes on that scenario. We consider only one hegemony graph with parameter $\alpha = 0.2$, based on our results from subsection 4.4. We re-implement the BGP part of IODA [8] via a QuickRIB observer that gives us access to older data not accessible via their API, like the `ttnet` scenario of 2004. Also this implementation allows us to ensure that the input BGP data is the same for MADstar and IODA. We validate the correctness of our implementation by comparing our prefix visibility signals to those of their API. Since IODA's visibility signal decreases during an anomaly, it cannot be directly used as an anomaly score. Thus, we invert their signal (without altering it's range) as an anomaly score.

Table 4 (see Table 6 in appendix for additional scenarios) compares MADstar and IODA in the task of detecting the ASes that have been reported in the literature as potentially impacted by the anomalies. For comparison, we also report the performance of a random detector as a baseline.

For the `aws` and `tm` scenarios, MADstar identifies with excellent accuracy the concerned ASes in both ROC AUC and PR AUC. In contrast, IODA struggles detecting them, particularly in terms of

| Potentially impacted ASes | | | IODA | | MADStar | | Random | |
|---|---|---|---|---|---|---|---|---|
| ASN | Name | CC | PR AUC | ROC AUC | PR AUC | ROC AUC | PR AUC | ROC AUC |
| *Scenario:* `ttnet` | | | | | | | | |
| **9121** | **TTNet** | **TR** | 0.69 | 0.73 | 0.97 | 1.00 | 0.11 | 0.50 |
| 1239 | Sprint | US | 0.11 | 0.35 | 0.93 | 0.98 | 0.11 | 0.50 |
| 6762 | SEABONE-NET | IT | 0.11 | 0.51 | 0.78 | 0.80 | 0.11 | 0.50 |
| 3561 | LVLT-3561 | US | 0.09 | 0.50 | 0.41 | 0.85 | 0.11 | 0.50 |
| 701 | Verizon | US | 0.09 | 0.26 | 0.33 | 0.85 | 0.11 | 0.50 |
| 174 | COGENT | US | 0.15 | 0.55 | 0.30 | 0.81 | 0.11 | 0.50 |
| 6453 | TATA | US | 0.19 | 0.78 | 0.29 | 0.65 | 0.11 | 0.50 |
| 6939 | HURRICANE | US | 0.09 | 0.50 | 0.25 | 0.82 | 0.11 | 0.50 |
| 3491 | PCCW | US | 0.09 | 0.50 | 0.13 | 0.61 | 0.11 | 0.50 |
| *Scenario:* `aws` | | | | | | | | |
| **16509** | **AMAZON-02** | **US** | 0.02 | 0.13 | 0.96 | 0.99 | 0.02 | 0.50 |
| 200759 | FLOW | CH | 0.01 | 0.25 | 0.96 | 0.97 | 0.02 | 0.50 |
| 6939 | HURRICANE | US | 0.02 | 0.50 | 0.96 | 0.96 | 0.02 | 0.50 |
| 46841 | FORKNETWORKING | US | 0.02 | 0.50 | 0.81 | 0.98 | 0.02 | 0.50 |
| *Scenario:* `tm` | | | | | | | | |
| **4788** | **TMNET-AS-AP** | **MY** | 0.02 | 0.22 | 1.00 | 1.00 | 0.03 | 0.50 |
| 3356 | LEVEL3 | US | 0.08 | 0.76 | 1.00 | 1.00 | 0.03 | 0.50 |
| 3549 | LVLT-3549 | US | 0.03 | 0.52 | 0.98 | 1.00 | 0.03 | 0.50 |
| 10753 | LVLT-10753 | US | 0.03 | 0.46 | 0.77 | 0.98 | 0.03 | 0.50 |
| *Scenario:* `indosat` | | | | | | | | |
| **4761** | **INDOSAT-INP-AP** | **ID** | 0.02 | 0.05 | 0.94 | 0.99 | 0.03 | 0.50 |
| 7018 | ATT-INTERNET4 | US | 0.03 | 0.47 | 0.08 | 0.81 | 0.03 | 0.50 |
| 58682 | LEVEL3-BD | BD | 0.02 | 0.50 | 0.06 | 0.78 | 0.03 | 0.50 |
| 4637 | ASN-TELSTRA-GLOBAL | HK | 0.02 | 0.50 | 0.05 | 0.73 | 0.03 | 0.50 |
| 251 | KAIAGLOBAL-AS | EU | 0.02 | 0.49 | 0.02 | 0.32 | 0.03 | 0.50 |

**Table 4: MADstar and IODA node anomaly performance metrics for potentially impacted ASes. Bold ASes are used for the hyper-parameter optimisation. Countries and AS names are obtained from CAIDA [1].**

PR AUC, where its performance is similar to that of the random baseline. In the `aws` event, AS6939, a transit provider for AWS, erroneously accepted AWS prefixes announced by AS200759, consequently redirecting traffic from AWS customers (such as AS46841) to AS200759 [29]. In the `tm` event, AS4788 (Telekom Malaysia) originated a route leak that was subsequently propagated by the tier-1 provider AS3549 [45], which happens to belong to the same organization as AS3356 and AS10753.

For the `indosat` scenario, we observe an unusually low ROC AUC for IODA on the Indosat AS. This indicates that abnormal graphs receive low scores and normal graphs large scores. In other words, positive samples are interpreted as negative ones and vice-versa. We attribute this inversion to the nature of the anomaly: Indosat performed a massive prefix hijack, announcing roughly 400,000 prefixes instead of the usual 300 [44]. This actually augments prefix visibility rather than decreasing it (which opposes our interpretation of a positive sample). While IODA effectively detected such change for Indosat, it did not for the other ASes, on which it performed similar to the random baseline. MADstar also struggled in PR AUC, as it only performs well on Indosat. Yet, it performed significantly better than IODA in terms of ROC AUC. This indicates that MADstar flags the concerned ASes, but several others as well.

Considering `ttnet`, MADstar detects TTNET with excellent accuracy. For the reminder of ASes, MADstar achieves good ROC AUC, in general, and good PR AUC on three key ASes: AS1239, AS6762, and AS3561. Indeed, in this scenario, TTNET incorrectly announced approximately 100,000 prefixes that were propagated by AS6762, a tier-1 main transit provider. AS1239 and AS3561 were among the ASes that announced the most paths associated to bad prefixes [37]. Concerning IODA, it can be seen that it is less accurate in detecting TTNET and that it severely struggles in detecting the remainder of ASes in both ROC AUC and PR AUC.

The rest of scenarios are reported in Table 6 (see appendix), where it can be seen that the performance of MADstar and IODA remain consistent with our findings in Table 4. We consider that MADstar is able to systematically perform better, particularly in ROC AUC, due to a better exploitation of the BGP signal: while IODA leverages BGP based only on prefix visibility, MADstar leverages temporal and structural patterns. This is particularly beneficial in cases like the `tm` scenario, where a globally impactful route leak caused a massive routing rewiring that significantly altered the topological structure, yet keeping the prefix visibility largely stable.

After demonstrating MADstar's ability to flag ASes that are already known in the literature to have been impacted by the anomalies, we now study how MADstar scores the rest of ASes for which there is no such information available. Given the lack of comprehensive ground-truth data, we analyze the geographical distribution of flagged ASes as a means to assess if they align with expected impact patterns. We focus on three specific scenarios which are the most likely to exhibit well-defined geographical effects: *(i)* **Italy** [42]: in February 2023, an incident at Italy's largest ISP disconnected approximately 38% of the country's Internet users; *(ii)* **Iran** [27]: in July 2024, Iran's second-largest AS by customer population disappeared from the global routing table following a government-mandated shutdown; and *(iii)* **Indosat** [44]: in April 2014, one of Indonesia's largest telecommunication networks executed a large-scale prefix hijack by announcing 417,038 prefixes, compared to its normal total of approximately 300.

We investigate the geographical distribution of ASes whose changes in anomaly scores over time best align with the outage period. This is, we compute a PR AUC value per AS, based on the known outage duration, and then we rank the ASes in decreasing order of PR AUC. We investigate if among the top ASes some countries are over-represented. For this, we collect AS-to-country mappings from CAIDA's AS-organization dataset [1] and report in Figure 7 the empirical cumulative distribution function (ECDF) of countries for the top-k ASes, as k varies.

For the `iran` scenario, two countries are over-represented: Iran itself and Iraq. We attribute Iraq's presence to its geographical proximity with Iran. This finding aligns with a statement from the Deputy Minister of Communications and Information Technology and CEO of the Infrastructure Communications Company indicating that "about 40% of the incoming international internet capacity in neighboring countries had been cut"[4]. For the `italy` and `indosat` scenarios, we find that only the curves corresponding to Italy and Indonesia, respectively, are over-represented, which precisely matches our expectations given the localized nature of these incidents. Overall, these results show that MADstar can be used to pinpoint the geographical scope of anomalies.

[4]https://www.iranintl.com/en/202407111864

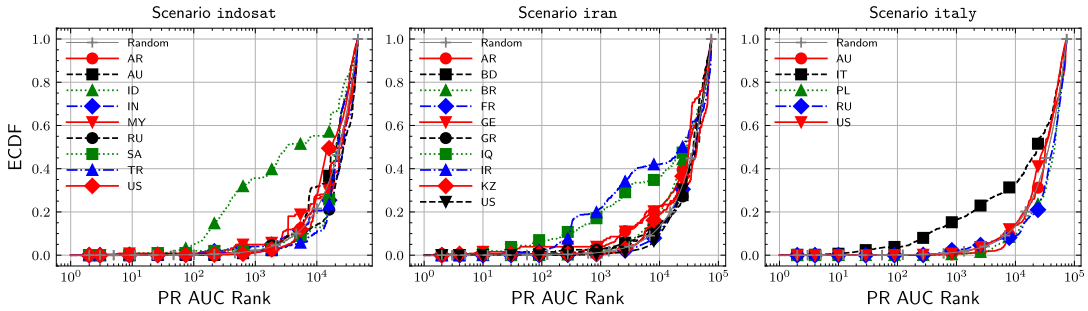



**Figure 7: PR AUC rank ECDF of countries associated to ASes scored by MADstar.**

## 5 Discussion

Despite the growing number of machine learning approaches proposed in the literature, we found that many of these techniques are not well suited for the practical design of a BGP anomaly detector. A primary limitation lies in the lack of labeled BGP data required to train supervised learning models. Although BGP data is abundant, BGP anomalies are often not or poorly annotated, hence relying on supervised methods raises substantial challenges in terms of generalization and reliability. In contrast, unsupervised approaches, such as MADBGP, seem better suited for the challenge.

The lack of ground truth data is however still an issue when evaluating unsupervised anomaly detectors. For evaluating MADBGP, we had to compile a ground truth dataset, and this process revealed several challenges inherent to the definition and validation of BGP anomalies. Firstly, the actual start and end times of a BGP anomaly may not perfectly align with the times reported in the literature. Such misalignment can bias the evaluation of an algorithm, especially when using performance metrics like PR AUC, which strongly penalize false positives and false negatives. Secondly, it remains ambiguous whether certain outages should manifest in BGP data at all, and if so, at what granularity (Internet-wide, country-level or AS-level). These are fundamental challenges to advance in this domain, requiring more attention from the research community.

MADBGP is built upon the MAD algorithm, which offers potential for further enhancements. Like any window-based method, MAD struggles when the anomaly last for a long time compared to the window, as it starts to consider the anomaly as the normality. This calls for the enhancement of MAD to take into account the stationary or transient nature of the anomalies, which can be achieved by boosting it with mechanisms used by change point detection methods [48]. Another promising direction is the incorporation of a time-frequency analysis [5] into the definition of normality, so as to take into account the cyclical dynamics of the Internet topology. Continuing the developments in [4], it would also be relevant to investigate the signatures that Internet events have in the MAD coefficients, as a means to automatically classify the events.

In addition, MADBGP analyzes the temporal Internet graph at two different scales. MADglob is designed to detect large-scale anomalies with global impact on the Internet's topology, while MADstar focuses on individual nodes, enabling the identification of localized disruptions. These approaches represent two extremes in the spectrum of topological granularity. A natural extension of this work would be to explore intermediate, regional-scale techniques.

## 6 Conclusion

In this paper, we presented MADBGP, a BGP anomaly detection pipeline that reconciles the graph and temporal aspects of the AS-level topology. With QuickRIB, a novel tool for BGP RIB and updates processing, combined with an extension of the MAD anomaly detection method, we proposed an unsupervised, multi-scale, and near real-time anomaly detection pipeline. In particular, we study a local version of MAD, which we coined MADstar, and a global version, which we coined MADglob, each allowing us to focus on anomalies at either local or the global level, respectively. Through extensive experiments on ten real-world scenarios at both the global Internet scale and local AS-level scale, we demonstrated that MADBGP's is able to leverage the BGP signal more effectively than IODA, permitting a more accurate detection of the anomalies in the majority of the studied scenarios. Moreover, a robustness analysis shows that MADBGP consistently ranks abnormal samples above normal ones across a broad range of its hyperparameters, making it a highly suitable tool for reducing the search space where anomalies are likely to be found. We believe that this can be of great practical utility for both researchers and network operators. In particular, our experiments allow to see that MADGBGP can be used in two modes: MADglob as a means to monitor the whole Internet topology, thus allowing to raise alarms in near real-time but without localizing the anomalies, and then MADstar, which allows to spot the concerned ASes by an event. Notably, we leveraged MADstar to shed light on several ASes that were affected in our studied scenarios, but that had not been previously reported in the literature. The next steps consist of further improving the anomaly detection algorithm, potentially through recent approaches that account for the cyclic nature of temporal graphs, and of assessing its generalization capabilities on future events through a practical deployment.

## Acknowledgments

This work is funded in part by the ANR (French National Agency of Research) through the CPJ contract of Esteban Bautista (ANR-23-CPJ1-0048-01) and the France 2030 Data2Laws (ANR-24-EXEN-0002) grants. We would like to thank Malte Tashiro for their help with BGP data processing.

## References

[1] [n. d.]. AS to organizations mappings. https://catalog.caida.org/dataset/as_organizations. https://doi.org/dataset/as_organizations Dates used: 2014-01-01, 2023-01-01, 2024-07-01. Accessed: 2025/04/15.

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-Generation Hyperparameter Optimization Framework. In *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2623–2631.

[3] Bahaa Al-Musawi, Philip Branch, and Grenville Armitage. 2016. BGP anomaly detection techniques: A survey. *IEEE Communications Surveys & Tutorials* 19, 1 (2016), 377–396.

[4] Esteban Bautista, Laurent Brisson, Cécile Bothorel, and Grégory Smits. 2024. MAD: Multi-Scale Anomaly Detection in Link Streams. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 38–46.

[5] Esteban Bautista and Matthieu Latapy. 2023. A frequency-structure approach for link stream analysis. In *Temporal Network Theory*. Springer, 449–482.

[6] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems* 24 (2011).

[7] Siddharth Bhatia, Bryan Hooi, Minji Yoon, Kijung Shin, and Christos Faloutsos. 2020. Midas: Microcluster-based detector of anomalies in edge streams. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 3242–3249.

[8] Zachary S Bischof, Kennedy Pitcher, Esteban Carisimo, Amanda Meng, Rafael Bezerra Nunes, Ramakrishna Padmanabhan, Margaret E Roberts, Alex C Snoeren, and Alberto Dainotti. 2023. Destination unreachable: Characterizing internet outages and shutdowns. In *Proceedings of the ACM SIGCOMM 2023 Conference*. 608–621.

[9] Min Cheng, Qian Xu, LV Jianming, Wenyin Liu, Qing Li, and Jianping Wang. 2016. MS-LSTM: A multi-scale LSTM model for BGP anomaly detection. In *2016 IEEE 24th international conference on network protocols (ICNP)*. IEEE, 1–6.

[10] Catalin Cimpanu. 2019. *For two hours, a large chunk of European mobile traffic was rerouted through China*. ZDNET. Retrieved 2025-05-15 from https://www.zdnet.com/article/for-two-hours-a-large-chunk-of-european-mobile-traffic-was-rerouted-through-china

[11] Luca Cittadini, Wolfgang Mühlbauer, Steve Uhlig, Randy Bush, Pierre Francois, and Olaf Maennel. 2010. Evolution of internet address space deaggregation: myths and reality. *IEEE Journal on Selected Areas in Communications* 28, 8 (2010), 1238–1249.

[12] Marijana Cosovic, Slobodan Obradovic, and Emina Junuz. 2018. Deep Learning for Detection of BGP Anomalies. In *Time Series Analysis and Forecasting*, Ignacio Rojas, Héctor Pomares, and Olga Valenzuela (Eds.). Springer International Publishing, Cham, 95–113.

[13] Jesse Davis and Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning*. 233–240.

[14] Iñigo Ortiz de Urbina Cazenave, Erkan Köşlük, and Murat Can Ganiz. 2011. An anomaly detection framework for BGP. In *2011 International Symposium on Innovations in Intelligent Systems and Applications*. 107–111. https://doi.org/10.1109/INISTA.2011.5946083

[15] Paulo Fonseca, Edjard S Mota, Ricardo Bennesby, and Alexandre Passito. 2019. BGP dataset generation and feature extraction for anomaly detection. In *2019 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 1–6.

[16] Romain Fontugne, Anant Shah, and Emile Aben. 2018. The (thin) bridges of as connectivity: Measuring dependency using as hegemony. In *Passive and Active Measurement: 19th International Conference, PAM 2018, Berlin, Germany, March 26–27, 2018, Proceedings 19*. Springer, 216–227.

[17] Julien Gamba, Romain Fontugne, Cristel Pelsser, Randy Bush, and Emile Aben. 2017. Bgp table fragmentation: what & who?. In *Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication*.

[18] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. 1995. *Design patterns: elements of reusable object-oriented software*. Pearson Deutschland GmbH.

[19] Dan Goodin. 2018. *Google goes down after major BGP mishap routes traffic through China*. Arstechnica. Retrieved 2025-05-15 from https://arstechnica.com/information-technology/2018/11/major-bgp-mishap-takes-down-google-as-traffic-improperly-travels-to-china/

[20] Xingzhi Guo, Baojian Zhou, and Steven Skiena. 2022. Subset Node Anomaly Tracking over Large Dynamic Graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 475–485.

[21] Rifqy Hakimi and Martin J. Reed. 2025. Feature Analysis and Selection for BGP Anomaly Detection. In *2025 28th Conference on Innovation in Clouds, Internet and Networks (ICIN)*. 99–106. https://doi.org/10.1109/ICIN64016.2025.10942841

[22] Mahmoud Hashem, Ahmed Bashandy, and Samir Shaheen. 2019. Improving anomaly detection in BGP time-series data by new guide features and moderated feature selection algorithm. *Turkish Journal of Electrical Engineering and Computer Sciences* 27, 1 (2019), 392–406.

[23] Kevin Hoarau, Kévin Hoarau, Tahiry Razafindralambo, and Pierre Ugo Tournoux. 2024. Unsupervised representation learning for BGP anomaly detection using graph auto-encoders. *ITU Journal on Future and Evolving Technologies* 5, 1 (2024), 120–133.

[24] Kevin Hoarau, Pierre Ugo Tournoux, and Tahiry Razafindralambo. 2021. BML: an efficient and versatile tool for BGP dataset collection. In *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 1–6.

[25] Kevin Hoarau, Pierre Ugo Tournoux, and Tahiry Razafindralambo. 2021. Suitability of graph representation for bgp anomaly detection. In *2021 IEEE 46th Conference on Local Computer Networks (LCN)*. IEEE, 305–310.

[26] Shenyang Huang, Yasmeen Hitti, Guillaume Rabusseau, and Reihaneh Rabbany. 2020. Laplacian change point detection for dynamic graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 349–358.

[27] IODA. 2024. *IODA shows two significant network disruptions in Iran*. Georgia Institute of Technology. Retrieved 2025-05-15 from https://x.com/IODA_live/status/1811044793763463585

[28] Yan Jiang and Guannan Liu. 2022. Two-stage anomaly detection algorithm via dynamic community evolution in temporal graph. *Applied Intelligence* 52, 11 (2022), 12222–12240.

[29] Mohit Lad. 2016. *Decoding Performance Data from Large Scale Internet Outages*. Thousand Eyes. Retrieved 2025-05-15 from https://youtu.be/8FRLZl23WJ4?si=NgV9O3hKms86qOAI&t=1132

[30] Hamid Latif, Jordi Paillissé, Jinze Yang, Albert Cabellos-Aparicio, and Pere Barlet-Ros. 2022. Unveiling the potential of graph neural networks for BGP anomaly detection *(GNNet '22)*. Association for Computing Machinery, New York, NY, USA, 7–12. https://doi.org/10.1145/3565473.3569188

[31] Jianning Mai, Lihua Yuan, and Chen-Nee Chuah. 2008. Detecting BGP anomalies with wavelet. In *NOMS 2008 - 2008 IEEE Network Operations and Management Symposium*. 465–472. https://doi.org/10.1109/NOMS.2008.4575169

[32] Maxwell McNeil, Carolina Mattsson, Frank W Takes, and Petko Bogdanov. 2023. CADENCE: Community-Aware Detection of Dynamic Network States. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*. SIAM, 1–9.

[33] Pablo Moriano, Raquel Hill, and L. Jean Camp. 2021. Using bursty announcements for detecting BGP routing anomalies. *Computer Networks* 188 (2021), 107835. https://doi.org/10.1016/j.comnet.2021.107835

[34] Chiara Orsini, Alistair King, Danilo Giordano, Vasileios Giotsas, and Alberto Dainotti. 2016. BGPStream: a software framework for live and historical BGP data analysis. In *Proceedings of the 2016 Internet Measurement Conference*. 429–444.

[35] Ramakrishna Padmanabhan, Arturo Filastò, Maria Xynou, Ram Sundara Raman, Kennedy Middleton, Mingwei Zhang, Doug Madory, Molly Roberts, and Alberto Dainotti. 2021. A multi-perspective view of Internet censorship in Myanmar. In *Proceedings of the ACM SIGCOMM 2021 Workshop on Free and Open Communications on the Internet*. 27–36.

[36] Songtao Peng, Jiaqi Nie, Xincheng Shu, Zhongyuan Ruan, Lei Wang, Yunxuan Sheng, and Qi Xuan. 2022. A multi-view framework for BGP anomaly detection via graph attention network. *Computer Networks* 214 (2022), 109129.

[37] Alin C. Popescu, Brian J. Premore, and Todd Underwood. 2004. *The Anatomy of a Leak: AS9121*. Renesys. Retrieved 2025-05-15 from https://youtu.be/l_UCneZm6dE?si=hBucR19P8JYxNWBk&t=619

[38] B. Aditya Prakash, Nicholas Valler, David Andersen, Michalis Faloutsos, and Christos Faloutsos. 2009. BGP-lens: patterns and anomalies in internet routing updates. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Paris, France) *(KDD '09)*. Association for Computing Machinery, New York, NY, USA, 1315–1324. https://doi.org/10.1145/1557019.1557160

[39] RIPE. 2008. *YouTube Hijacking: A RIPE NCC RIS case study*. RIPE. Retrieved 2025-05-15 from https://www.ripe.net/about-us/news/youtube-hijacking-a-ripe-ncc-ris-case-study/

[40] Odnan Ref Sanchez, Simone Ferlin, Cristel Pelsser, and Randy Bush. [n. d.]. Comparing Machine Learning Algorithms for BGP Anomaly Detection Using Graph Features. In *Proceedings of the 3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks* (Orlando FL USA, 2019-12-09). ACM, 35–41. https://doi.org/10.1145/3359992.3366640

[41] Tom Strickx. 2019. *How Verizon and a BGP Optimizer Knocked Large Parts of the Internet Offline Today*. Cloudflare. Retrieved 2025-05-15 from https://blog.cloudflare.com/how-verizon-and-a-bgp-optimizer-knocked-large-parts-of-the-internet-offline-today

[42] Massimiliano Stucchi. 2023. *Italy's Internet Outage a Perfect Storm*. Internet Society. Retrieved 2025-05-15 from https://pulse.internetsociety.org/blog/italys-internet-outage-a-perfect-storm

[43] Zoltán Tasnádi and Noémi Gaskó. 2022. A new type of anomaly detection problem in dynamic graphs: An ant colony optimization approach. In *International Conference on Bioinspired Optimization Methods and Their Applications*. Springer, 46–53.

[44] Andree Toonk. 2014. *Hijack event today by Indosat*. Cisco. Retrieved 2025-05-15 from https://www.bgpmon.net/hijack-event-today-by-indosat

[45] Andree Toonk. 2015. *Massive route leaks causes massive Internet slowdown*. Cisco. Retrieved 2025-05-15 from https://www.bgpmon.net/massive-route-leak-cause-internet-slowdown

[46] Minji Yoon, Bryan Hooi, Kijung Shin, and Christos Faloutsos. 2019. Fast and accurate anomaly detection in dynamic graphs with a two-pronged approach. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 647–657.
[47] Mingwei Zhang. 2023. *BGPKIT*. BGPKIT. Retrieved 2025-05-15 from https://bgpkit.com/
[48] Yuxuan Zhou, Shang Gao, Dandan Guo, Xiaohui Wei, Jon Rokne, and Hui Wang. 2024. A survey of change point detection in dynamic graphs. *IEEE Transactions on Knowledge and Data Engineering* (2024).

| Temporal graph | | | Detector performance | | | |
|---|---|---|---|---|---|---|
| | | | MAD | | MADstatic | |
| Scenario | Graph type | Weight param | PR AUC | ROC AUC | PR AUC | ROC AUC |
| china europeanmobile | graph | ips_count | | | 0.23 | 0.95 |
| | | paths_count | | | 0.06 | 0.83 |
| | **hegemony** | 0 | 0.84 | 0.96 | 0.32 | 0.97 |
| | | 0.1 | | | 0.69 | 0.92 |
| | | **0.2** | | | **0.70** | **0.94** |
| | | 0.3 | | | 0.63 | 0.92 |
| | | 0.4 | | | 0.17 | 0.90 |
| iran | graph | ips_count | | | 0.05 | 0.68 |
| | | paths_count | | | 0.21 | 0.88 |
| | **hegemony** | 0 | 0.20 | 0.82 | 0.10 | 0.84 |
| | | 0.1 | | | 0.30 | 0.80 |
| | | **0.2** | | | **0.37** | **0.86** |
| | | 0.3 | | | 0.17 | 0.81 |
| | | 0.4 | | | 0.10 | 0.85 |
| italy | graph | ips_count | | | 0.07 | 0.64 |
| | | paths_count | | | 0.17 | 0.76 |
| | **hegemony** | 0 | 0.57 | 0.93 | 0.44 | 0.88 |
| | | **0.1** | | | **0.46** | **0.94** |
| | | 0.2 | | | 0.53 | 0.93 |
| | | 0.3 | | | 0.49 | 0.89 |
| | | 0.4 | | | 0.50 | 0.90 |
| mainone | graph | ips_count | | | 0.06 | 0.80 |
| | | paths_count | | | 0.07 | 0.86 |
| | **hegemony** | 0 | 0.03 | 0.68 | 0.24 | 0.97 |
| | | **0.1** | | | **0.24** | **0.97** |
| | | 0.2 | | | 0.20 | 0.95 |
| | | 0.3 | | | 0.16 | 0.94 |
| | | 0.4 | | | 0.12 | 0.92 |
| pakistan youtube | **graph** | **ips_count** | 0.02 | 0.33 | **0.04** | **0.36** |
| | | paths_count | | | 0.02 | 0.49 |
| | hegemony | 0 | | | 0.02 | 0.55 |
| | | 0.1 | | | 0.03 | 0.63 |
| | | 0.2 | | | 0.02 | 0.53 |
| | | 0.3 | | | 0.03 | 0.55 |
| | | 0.4 | | | 0.03 | 0.64 |
| verizon allegheny | graph | ips_count | | | 0.55 | 0.98 |
| | | paths_count | | | 0.18 | 0.95 |
| | **hegemony** | 0 | 0.11 | 0.78 | 0.94 | 0.98 |
| | | 0.1 | | | 0.53 | 0.97 |
| | | **0.2** | | | **0.96** | **0.98** |
| | | 0.3 | | | 0.83 | 0.98 |
| | | 0.4 | | | 0.78 | 0.98 |

**Table 5: Same as Table 2 for additional scenarios. A random detector has a ROC AUC of 0.5 and a PR AUC of 0.02, 0.03, 0.04, 0.02, 0.02 and 0.02 respectively for the scenarios `china europeanmobile`, `iran`, `italy`, `mainone`, `pakistan youtube`, `verizon allegheny`.**

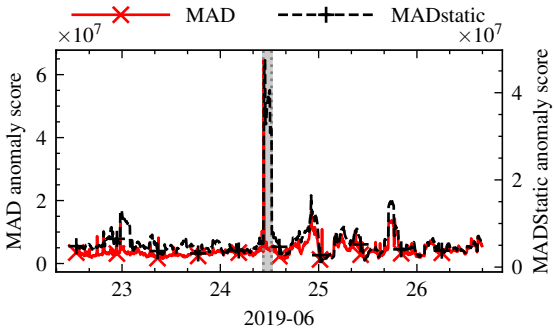



**Figure 8: MAD and MADstatic anomaly scores for `verizon allegheny`, computed on its best temporal graph with the optimal HPO trial parameters.**

## A Ethics

This work does not raise any ethical issues.

## B Proof of Proposition 1

The outline of the proof is as follows: we show that the coefficients produced by the data structure can be expressed as a matrix-vector product and then we identify the inverse of the matrix. Namely, let $Q$, which corresponds to the tree leaves, be arranged into a vector that follows the same indexing of the leaves. Moreover, let $\psi_k^{(\ell)}$ be a vector defined as

$$\psi_k^{(\ell)}(e_i) = \begin{cases} \sqrt{2^\ell}/\sqrt{M} & \text{if } e_i \in \mathcal{E}_{2k}^{(\ell+1)} \\ -\sqrt{2^\ell}/\sqrt{M} & \text{if } e_i \in \mathcal{E}_{2k+1}^{(\ell+1)} \\ 0 & otherwise. \end{cases} \tag{11}$$

Then, we have that Eq. (5) can be rewritten as $w_k^{(\ell)} = (\psi_k^{(\ell)})^\top Q$, where, for the sake of notation lightness, the time reference was omitted. Moreover, we have that $s$ can also be expressed as $s = \phi^\top Q$, where $\phi(e_i) = 1/\sqrt{M}$ for all $e_i$. As a result, the whole coefficient analysis made by the data structure can be expressed as the matrix-vector product $c = FQ$, where $F$ is the $M \times M$ matrix that contains the vectors $\psi_k^{(\ell)}$, for all $k$ and $\ell$, and the vector $\phi$, as rows. We have that any two vectors $\psi_k^{(\ell)}$ and $\psi_j^{(\ell)}$, for $k \neq j$, associated to one same level $\ell$, are orthogonal, as the tree leaves they cover form disjoint sets. We also have that any pair of vectors $\psi_k^{(\ell_1)}$ and $\psi_j^{(\ell_2)}$, for $\ell_1 \neq \ell_2$, are orthogonal. To show this, we have two possibilities, either $\psi_k^{(\ell_1)}$ has $\psi_j^{(\ell_2)}$ as ancestor in the tree, in which case $\psi_j^{(\ell_2)}$ has a constant value for all the edges $e_i$ where $\psi_k^{(\ell_1)}(e_i)$ is non-zero, thus making their product proportional to the sum of values in $\psi_k^{(\ell_1)}$, which is zero. The second case is when $\psi_j^{(\ell_2)}$ is not an ancestor of $\psi_k^{(\ell_1)}$, thus making them orthogonal as the tree leaves they cover form disjoint sets. Concerning $s$, we also have that it is orthogonal to all vectors $\psi_k^{(\ell_1)}$, as their product is proportional to the sum of values in $\psi_k^{(\ell)}$, which is zero. Lastly, we have that $(\psi_k^{(\ell)})^\top \psi_k^{(\ell)} = \sum_{e_i \in \mathcal{E}_k^{(\ell)}} 2^\ell/M = 1$, which follows from the fact that $|\mathcal{E}_k^{(\ell)}| = M/2^\ell$. The same holds for $s$, where $s^\top s = 1$ for the same reason. Thus, we have that $\psi_k^{(\ell)}$,

| Scenario | Potentially impacted ASes | | | IODA | | MADStar | | Random | |
|---|---|---|---|---|---|---|---|---|---|
| | ASN | Name | Country | PR AUC | ROC AUC | PR AUC | ROC AUC | PR AUC | ROC AUC |
| verizon allegheny | **33154** | **DQECOM** | **US** | 0.02 | 0.50 | 0.97 | 1.00 | 0.02 | 0.50 |
| | 701 | UUNET | US | 0.02 | 0.50 | 0.83 | 0.99 | 0.02 | 0.50 |
| | 396531 | ATIMETALS | US | 0.02 | 0.50 | 0.80 | 1.00 | 0.02 | 0.50 |
| italy | **6762** | **SEABONE-NET** | **IT** | 0.13 | 0.56 | 0.97 | 1.00 | 0.04 | 0.50 |
| | 3269 | ASN-IBSNAZ | IT | 0.09 | 0.52 | 0.94 | 0.99 | 0.04 | 0.50 |
| | 16232 | ASN-TIM | IT | 0.04 | 0.50 | 0.91 | 0.93 | 0.04 | 0.50 |
| pakistan youtube | **17557** | **PKTELECOM-AS-AP** | **PK** | 0.02 | 0.21 | 0.06 | 0.83 | 0.02 | 0.50 |
| | 36561 | YOUTUBE | US | 0.02 | 0.50 | 0.06 | 0.79 | 0.02 | 0.50 |
| | 3491 | BTN-ASN | US | 0.02 | 0.50 | 0.02 | 0.34 | 0.02 | 0.50 |
| iran | **58224** | **TCI** | **IR** | 0.15 | 0.47 | 0.98 | 1.00 | 0.03 | 0.50 |
| | **59441** | **Hostiran-Network** | **IR** | 0.98 | 0.99 | 0.92 | 0.97 | 0.03 | 0.50 |
| china europeanmobile | **21217** | **SAFEHOSTNET** | **CH** | 0.02 | 0.50 | 1.00 | 1.00 | 0.02 | 0.50 |
| | 4134 | CHINANET-BACKBONE | CN | 0.02 | 0.49 | 0.84 | 0.99 | 0.02 | 0.50 |
| | 3303 | SWISSCOM | CH | 0.02 | 0.50 | 0.29 | 0.97 | 0.02 | 0.50 |
| | 21502 | ASN-NUMERICABLE | FR | 0.02 | 0.50 | 0.12 | 0.93 | 0.02 | 0.50 |
| | 5410 | ASN-BOUYGTEL-ISP | FR | 0.02 | 0.50 | 0.07 | 0.83 | 0.02 | 0.50 |
| mainone | 20485 | TRANSTELECOM | RU | 0.02 | 0.50 | 0.03 | 0.69 | 0.02 | 0.50 |
| | 37282 | **MAINONE** | **NG** | 0.02 | 0.50 | 0.03 | 0.67 | 0.02 | 0.50 |
| | 4809 | CHINATELECOM | CN | 0.03 | 0.52 | 0.02 | 0.47 | 0.02 | 0.50 |

**Table 6: Same as Table 4 for additional scenarios.**

for all $k$ and $\ell$, and $s$, are orthonormal vectors. As a result, it can be concluded that the matrix $F$ is an orthogonal matrix. Due to its orthonormal and real-valued nature, it follows that $F^\top F = \mathbb{I}$, thus being inverted by its transpose. Eq. (7) follows from applying the product $Q = F^\top c$.

## C MADBGP performance analysis

To demonstrate the real-time capability of MADBGP, we report here benchmarks for MADglob and MADstar. The benchmarks were run on a single core of an Intel(R) Xeon(R) E5-2670 v3 CPU @ 2.30 GHz system.

### C.1 MADglob

In Table 7, we show benchmarks for the least and most computationally intensive experiments listed in Table 1. In the problem size description, we report the number of BGP entries processed to build the initial RIB, the average number of BGP updates applied between two Internet topology snapshots, the number of unique vantage point ASN used to compute the hegemony (subsection 3.2), the size of the resulting graph, and the length of the history considered by MAD. The execution time is decomposed in three parts: *(i)* the time taken to build the RIB and Internet topology observers at the beginning of the experiment (from the RIB MRT files); *(ii)* the average time taken to produce the hegemony graph snapshot associated to the last 5 minutes of BGP updates (i.e., application of updates to the helper data structures, computation of hegemony values, construction of graph snapshot); and *(iii)* the average time taken by MAD to compute the anomaly score of the current graph snapshot (query) under analysis.

In the case of `ttnet`, Table 7 shows that MADBGP runs quickly for this scenario, with less than a minute needed to build the initial RIB and Internet topology. Moreover, updating them from 5 minutes of real-time BGP updates takes only only 2.15 seconds. This scenario is not computationally intensive, as it involves a single route collector and only a few vantage points reporting data from the relatively small Internet of 2004 (only 18526 ASN compared to the estimated 80000 of 2025). MADBGP also operates in near real-time in the much more demanding `iran` scenario. In this scenario, the initial RIB must be constructed from 150 million BGP entries, which results in a setup time of 195.75 minutes. Even though this operation is slow, it needs to be performed only once. Once the initial RIB is set, it takes only 166 seconds (2.7 minutes), on average, to process any upcoming batch of 5 minutes of real-time BGP updates (741000 updates per 5 minute batch, on average). It is worth mentioning that our experiments were done on a single core CPU but they can be easily deployed on a cluster of machines that process route collectors and/or vantage point in parallel (as is often done in practice). Interestingly, MAD is extremely fast in computing the anomaly scores, taking, on average, roughly 1.2 seconds per snapshot (query) in both scenarios. Even though `ttnet` is smaller than `iran` in size, MAD takes similar time in both scenarios due to the fact that `ttnet` involved the analysis of a much larger window size. In the next subsection, we study more in depth the impact of the window size on MAD's computation time.

| | | ttnet | iran |
|---|---|---|---|
| Problem size | Number of BGP entries processed to build initial RIB | 501 000 | 149 000 000 |
| | Number of BGP updates between successive snapshots (avg.) | 2 506 | 741 000 |
| | Number of vantage points used to compute AS hegemony | 3 | 149 |
| | Number of links per snapshot (avg.) | 48 766 | 262 000 |
| | Number of nodes in temporal graph | 18 526 | 76 139 |
| | MAD window size (N) | 173 | 33 |
| Execution time (seconds) | Time to build the initial RIB and initial topology | 38.82 | 11 745 |
| | Time to generate a 5-minute hegemony graph snapshot (avg.) | 1.02 | 166.17 |
| | Time to compute the anomaly score of a graph snapshot (avg.) | 1.13 | 1.20 |

**Table 7: Benchmarks for the least and most computationally intensive experiments listed in Table 1.**

## C.2 Impact of the window size on MADglob

Beside the input data size, the other parameter impacting the execution time is the window size used in MAD anomaly scoring. We report in Figure 10 the execution time of MAD anomaly scoring from already generated Internet topologies of the most computationally intensive scenario. We see that MAD and MADstatic scales linearly, with execution time below 200 seconds for MAD which allow MADglob pipeline to run in near-real time. For MADstatic, the execution time is of few seconds which allows us to do extensive hyper-parameter search in subsection 4.4.

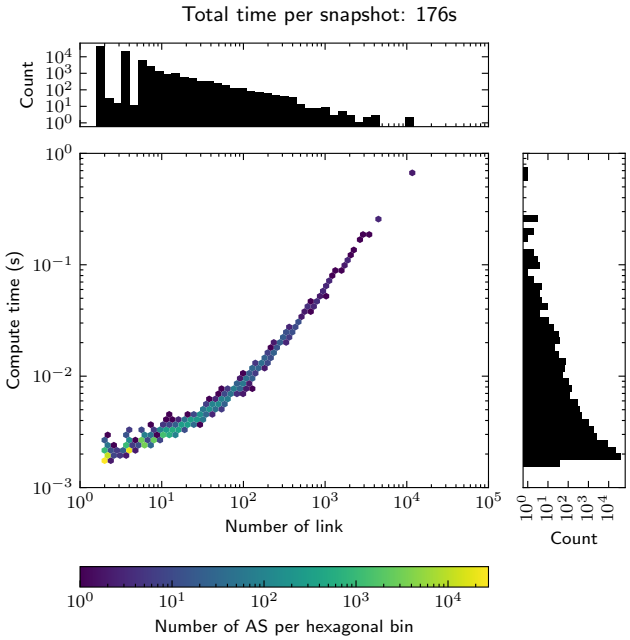



**Figure 9: MADstar compute time for all ASes of the scenario `iran`. Averaged over ten 5-minutes snapshot processing.**

## C.3 MADstar

We compute MADstar for all ASes and report execution times in Figure 9. Processing 5 minutes of real-time data took only 176 seconds, and the execution time for a single AS is negligible. Because execution time scales linearly with the number of links, we expect MADstar to remain efficient as the Internet continues to grow.

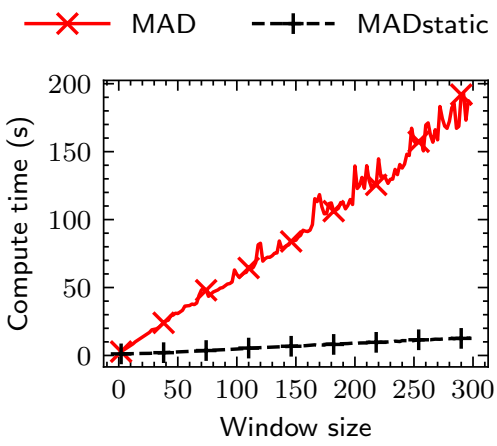



**Figure 10: Impact of the window size on the compute time of MAD and MADstatic anomaly scoring for the scenario `iran`. Averaged over ten 5-minutes snapshot processing.**

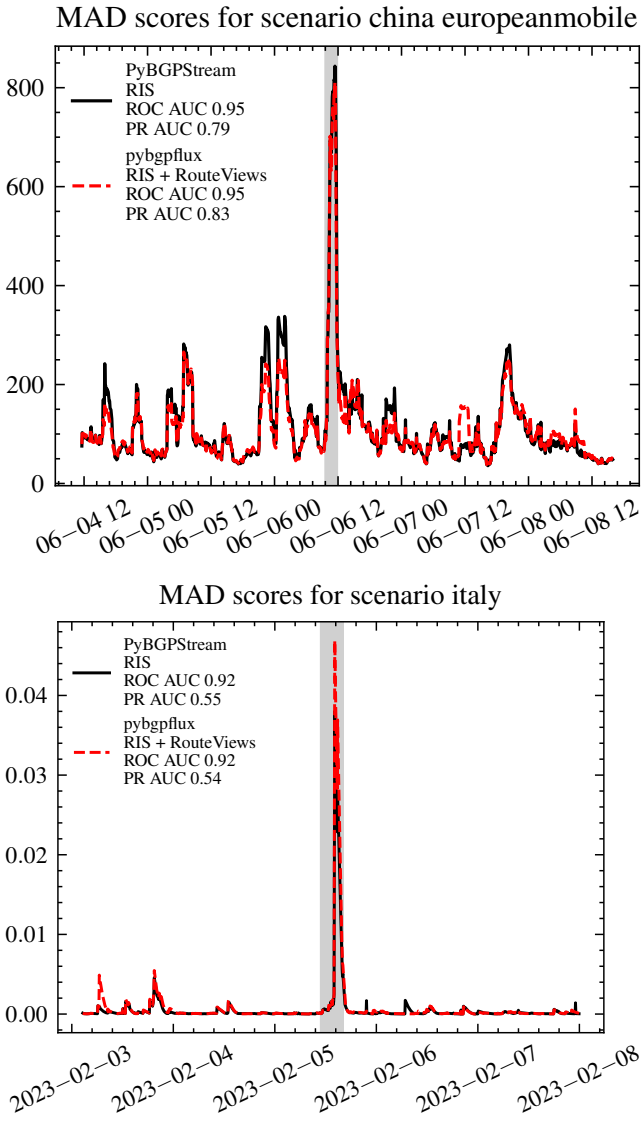



**Figure 11: Impact of the BGPStream provider and the selection of route collectors on anomaly scores. For two scenarios, we show the results presented in the paper (using PyBGPStream and RIS collectors), as well as the results when adding 2 RouteViews collectors (route-views2 and route-views.linx) and using PyBGPKITStream.**

## C.4 Impact of BGPStream provider and route collectors

To investigate the impact of the BGPStream provider and route collectors selection, we re-run two Internet-scale anomaly detections and compare the anomaly scores. In Figure 11, the dashed line show the anomaly scores obtained by replacing PyBGPStream with our own implementation PyBGPKITStream, as well as by adding two large collectors from the RouteViews project (route-views2 and route-views.linx). In both cases the curves almost perfectly overlap, showing that MADBGP results are not heavily impacted by the BGPStream provider and route collectors selection.