



# Lightweight Network Delay Segmentation Based on Smoothed Hierarchical Clustering

Sanaa Ghandi, Alexandre Reiffers-Masson, Sandrine Vaton, Thierry Chonavel

## ► To cite this version:

Sanaa Ghandi, Alexandre Reiffers-Masson, Sandrine Vaton, Thierry Chonavel. Lightweight Network Delay Segmentation Based on Smoothed Hierarchical Clustering. IFIP Networking 2023, Jun 2023, Barcelone, Spain. hal-04154314

**HAL Id: hal-04154314**

**<https://inria.hal.science/hal-04154314>**

Submitted on 6 Jul 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution| 4.0 International License

# Lightweight Network Delay Segmentation Based on Smoothed Hierarchical Clustering

Sanaa Ghandi, Alexandre Reiffers-Masson, Sandrine Vaton, Thierry Chonavel

*IMT Atlantique, LAB-STICC laboratory*

Brest, France

{sanaa.ghandi, alexandre.reiffers-masson, sandrine.vaton, thierry.chonavel}@imt-atlantique.fr

**Abstract**—Delay measurements are important for network monitoring. With Internet monitoring platforms providing an unparalleled amount of data, it becomes necessary to automate their treatment. In particular, segmenting these delays permits supervising infrastructures and analyzing possible incidents. This paper explores the use of hierarchical clustering for the segmentation of multivariate network delays. The proposed method offers a computationally efficient way to identify the spatial correlation of network delay and to jointly segment time series within the same cluster. A post-treatment step is introduced that involves the Viterbi algorithm to smooth segmentation and handle the temporal dependency more effectively. This global method is evaluated on two real-world datasets, demonstrating its suitability for managing delays with varying variance and changing patterns. The proposed approach provides an efficient and cost-effective method for automated delay characterization.

**Index Terms**—Network delay, time series segmentation, hierarchical clustering, Viterbi algorithm

## I. INTRODUCTION

The importance of Internet end-to-end delay metrics, such as the round trip time (RTT), is well established, as it can be used to assess performance, the status of a specific path, the quality of user experience, and real-time applications. Typically, delay measurements exhibit similar patterns when the routes followed contain shared segments. In [1], it was observed that delays between pairs passing through the same damaged Internet Exchange Point (IXP), presented synchronized pattern changes. An example of spatially correlated paths can be seen in Figure 1, which displays a group of Internet delay time series that exhibit similar behaviors. This similarity can be attributed to the underlying network routing.

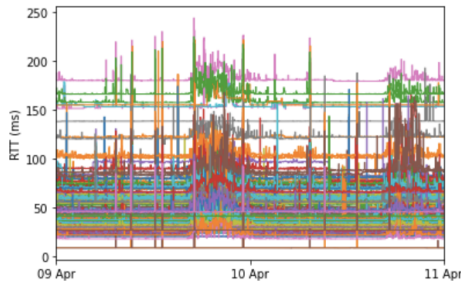


Fig. 1. Time series with correlated change patterns.

Despite these changes that can be due to congestion [2], path changes [3] or major Internet incidents [4], when observed over a long period, end-to-end delays display temporal stationarity as can be seen in figure 2.

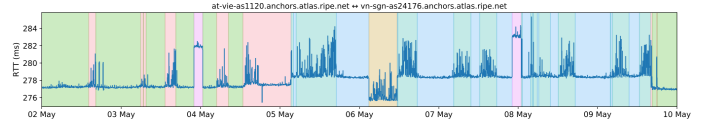


Fig. 2. A Ripe RTT time series over a week with the colors indicating its segmentation.

In the networking community, Internet delay segmentation and characterization have been widely studied as an intriguing problem. Traditionally, the segmentation of delays has been left to human experts, as illustrated in Figure 2, which demonstrates how easily an individual can perform this task. Nonetheless, with the abundance of data available on platforms such as Ripe Atlas and Caida [5], as well as on private ISP servers, relying on human analysts to segment delays is both expensive and challenging. Therefore, there is a need for an automated layer and the creation of appropriate tools and algorithms to meet this increasing demand.

Many works have aimed to model Internet delays in time [6], [7] in order to gain a better understanding of this metric and more insight into network performance and functioning [8]. Furthermore, Shao et al. [9] studied RTT segmentation in order to evaluate the impact of the routing changes at different levels (AS, IXP) on RTT changes, and Mouchet et al. [1] used HDP-HMM to detect IXP outages based on RTT changes frequency. Other characterization studies of the delays using different distributions have been conducted [10], [11], [12], as well as approaches based on deep learning [13].

Despite the high accuracy of RTT series segmentation achieved in some cited works, the state of the art presents two primary issues. Firstly, the suggested methods are computationally intensive. Secondly, many studies only take into account temporal stability, without taking full advantage of spatial dependencies among RTT series. To the best of our knowledge, this paper proposes the first multivariate segmentation of delay networks based on joint segmentation of multiple delay time series.

In this paper, we explore the use of hierarchical clustering for the segmentation of multivariate network delays. To

capture the spatial correlation in the dataset, we first apply hierarchical clustering on the Pearson correlation matrix of the RTT time series. Subsequently, when the clusters are identified, we jointly segment the time series within each cluster using hierarchical clustering. This clustering alone, however, does not incorporate the temporal stability property of these delays. Therefore, we propose a post-treatment step by exploiting the Viterbi algorithm, which is able to smooth the resulting clustering and handle the temporal dependency more effectively. Our global method is evaluated on two real-world datasets, demonstrating its suitability for managing delays with varying variance and changing patterns.

The paper is structured as follows. The second section is dedicated to the methodology accompanied by an illustrative example. It starts by presenting the hierarchical clustering, then explains the clustering of time series. Finally, the two-step multivariate segmentation using the Viterbi smoothed hierarchical clustering is described. The third section evaluates the methodology, first on the clustering of time series, then on the segmentation by comparing it to the most efficient state-of-the-art method both in terms of accuracy and execution time. Finally, the fourth section concludes this work and outlines possible avenues for further research.

## II. METHODOLOGY

### A. Principles of hierarchical clustering

Hierarchical clustering is a popular agglomerative clustering technique used to group variables based on similarity. This approach considers that each element is its own cluster initially. Then, clusters are iteratively merged into larger clusters. At each step, the two closest clusters are merged into one new parent cluster, and this process is repeated until a single global cluster remains after  $N-1$  iterations, with  $N$  the number of variables [14]. This iterative clustering process can be represented in a dendrogram, a tree structure plot.

The similarity between clusters is measured using a linkage method, such as single (sgl), complete (cpl), average (avg), centroid (ctr), or Ward (wrđ) linkage [15]. Single linkage considers the closest elements of the clusters, complete and average linkage considers the furthest elements and the average distance between all elements, respectively, and centroid linkage is based on the distance between the respective cluster centroids. Ward linkage minimizes the variance of the merged clusters, which is calculated by the error sum of squares defined for a given cluster  $C$  as  $ess(C) = \sum_{x \in C} (x - \frac{1}{|C|} \sum_{y \in C} y)^2$ , with  $|C|$  being the cardinal of the cluster  $C$ . The Ward linkage between two clusters  $C_1$  and  $C_2$  is then given by  $ess(C_1 \cup C_2) - (ess(C_1) + ess(C_2))$ .

In this paper, we use complete linkage for time series clustering and Ward linkage for the segmentation. In fact, we can see in table I that highlights pros and cons (in italic) of different linkage strategies [16] that ward and complete linkages are the only ones that can handle noise and outliers. The complete however is not good for handling stationarity since it tends to break large clusters.

TABLE I  
PROS AND CONS OF EACH LINKAGE METHOD

Pros and Cons	Sgl	Cpl	Avg	Ctr	Wrd
Handle noise between the clusters		✓	✓	✓	
Less susceptible to noise and outliers		✓			✓
Handle non-elliptical shapes	✓				
Capture clusters of different sizes	✓				
<i>Biased towards globular clusters</i>		✓	✓	✓	✓
<i>Sensitive to noise and outliers</i>	✓				
<i>Sensitive to noise between the clusters</i>	✓				
<i>Tend to break large clusters</i>		✓			

### B. The time series clustering

In this research, the hierarchical model is employed to efficiently cluster multivariate features. It is first used to make a profit from the spatial correlation of Internet delays by detecting groups of time series that display similar patterns. For this purpose, we introduce the following notation. We consider slotted time and suppose that a network containing  $n$  pairs of source/destination is observed during  $T$  timeslots. We observe the delay measurements between each source and destination of these pairs at different timeslots.  $D$  denotes the  $n$ -variate time series of length  $T$ ,  $D_i$ : the row  $i$  of the matrix  $D$  which represent the  $i$ -th observed time serie of delays and  $D_{:j}$  the column  $j$  of the matrix  $D$  which represent the vector of measured delays at instant  $j$ .

To shed light on the spatial correlation among time series, a Pearson correlation matrix is computed between each two time series of the matrix  $D$ . A hierarchical clustering is then performed on the values of the correlation matrix. Finally, the rows and columns of the Pearson correlation matrix are reordered according to the resulting cluster labels. In this new matrix, diagonal blocks are formed and each block represents groups of time series with higher correlation.

In order to estimate the number of clusters within a given dataset, we must determine the dendrogram cut-off. To set this threshold we would like to minimize the number of clusters while maximizing the correlation of time series within each cluster. For this purpose, we define the following criterion: For a partition  $P$  of the dataset, we define the metric:  $h(P) = \sum_{c \in P} \frac{1}{|c|} \sum_{s_k, s_l \in c} |corr(s_k, s_l)|$ , with  $corr(x, y)$  being the Pearson correlation between  $x$  and  $y$ .

An example of this process is given in figures 3 and 4 that show the correlation matrix of 200 time series before and after the hierarchical clustering. We can notice that in the second figure, blocks of highly correlated pairs are formed and are easier to identify.

In order to have a closer look at the formed clusters, consider for instance figures 1 and 5 that show the time series within the last two formed clusters at the bottom right of the Pearson matrix after clustering. It is easy to notice that within

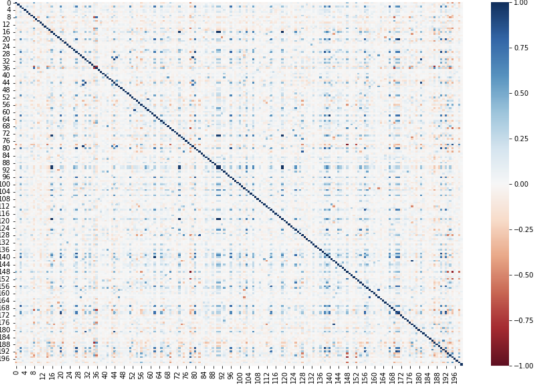


Fig. 3. Heatmap of the Pearson correlation matrix. On the axes are the numbers of paired series.

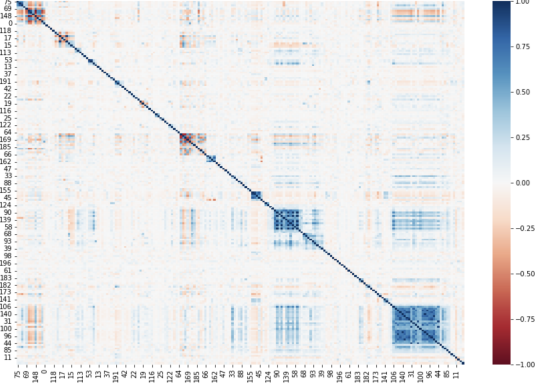


Fig. 4. Heatmap of the Pearson correlation matrix after hierarchical clustering.

each cluster, the time series show synchronous and similar patterns, whereas the pattern nature and temporal distribution change from one cluster to another. This property demonstrates that the change points are highly synchronized between correlated pairs. This motivates the intracluster segmentation that we prepare in the following. Let us denote by  $D^c$  the submatrix of  $D$  that contains the observed time series within the identified cluster  $c$ . Once the clusters of similar time series

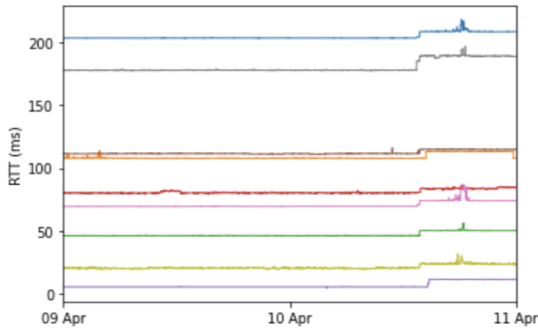


Fig. 5. Example of ten clustered time series

are identified, we are interested in segmenting jointly the delay time series within each cluster. In this second step, we first apply the hierarchical clustering on the columns of the matrix  $D^c$ . Figure 6 shows the result of this clustering. Despite the

effectiveness of capturing the spatial correlation, hierarchical clustering imperfectly incorporates the temporal correlation and stability. To address this problem, we propose a Viterbi algorithm as a post-treatment.

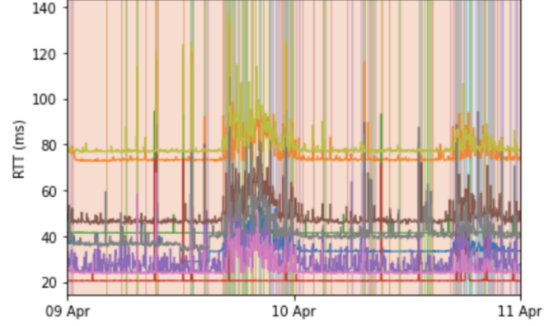


Fig. 6. Segmentation of the series in a cluster using hierarchical clustering

### C. Viterbi smoothing post-treatment

We suggest capturing the temporal correlation by modeling the multivariate delays present in  $D^c$  for each identified cluster  $c$  using a hidden Markov model (HMM). This model considers that the observed delay vectors  $(D_i^c)_{i \leq T}$  are generated by some hidden states  $(s_k)_{k \leq m}$ . In fact, the Markovian assumption regarding Internet delay changes is commonly used in the literature [1], [6]. The segmentation problem can hence be reformulated as follows: the segment labels are defined by the hidden states and the goal is to infer them based on the observed delay vectors. To this end, we use a Viterbi algorithm which is a dynamic programming algorithm for finding the most likely sequence of hidden states, called the Viterbi path, [17] that results in the sequence of observed events. It is commonly used in the context of Hidden Markov Models (HMMs). In simple terms, it calculates the most likely sequence of events that have led to the observed sequence.

We use the output of the hierarchical clustering segmentation on each matrix  $D^c$  to roughly identify the number of hidden states and estimate the emission probability of each state. In particular, we use observations identified within a cluster to initialize the distribution of emissions using either a density kernel method or a histogram method. Moreover, we consider a transition probability matrix to stress the fact that two successive delay vectors have more chance to be generated by the same hidden state. For this reason, we put ourselves in the context of a transition probability that is small compared to the probability of remaining in the same state. For the sake of simplicity, we define the transition probability  $m \times m$  matrix  $P_t$  as follows:  $P_t(i, j) = \epsilon / (m - 1)$  if  $i \neq j$  and  $P_t(i, i) = 1 - \epsilon$ , with  $m$  the number of hidden states. Choosing a small value of  $\epsilon$  enforces states' persistence. Finally, we consider the hypothesis that the network states are initially equally distributed.

However, this joint treatment of several time series is sensitive to the size of the cluster. Indeed, when the cluster contains many time series, the influence of the transition probability term tends to become negligible in front of the

emission probability term resulting in weak smoothing. In order to overcome this imbalance, we introduce the following alternative regularization that depends on the cluster size, say  $N$ : the transmission probability is set to  $P_t(i, j) = \epsilon^N / (m - 1)$  if  $i \neq j$  and  $P_t(i, i) = 1 - \epsilon^N$ . This version of smoothing will be called regularized Viterbi smoothing.

If we consider the result of the hierarchical clustering in figure 6 and apply the Viterbi algorithm, we get the segmentation result in figure 7. We can see that the segments are smoothed and more stability is incorporated. To further improve these results we estimate again emission probabilities from this new segmentation and apply again the Viterbi algorithm. The final result is provided in figure 8 which displays a slightly more smoothed segmentation than figure 7.

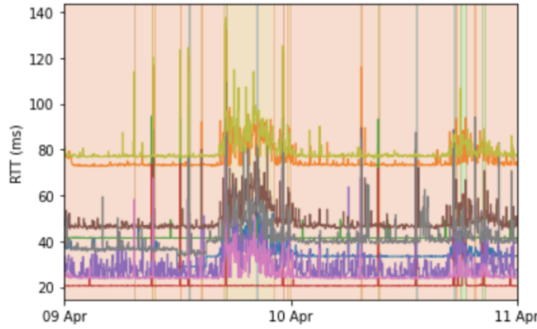


Fig. 7. Segmentation of the series after one Viterbi smoothing

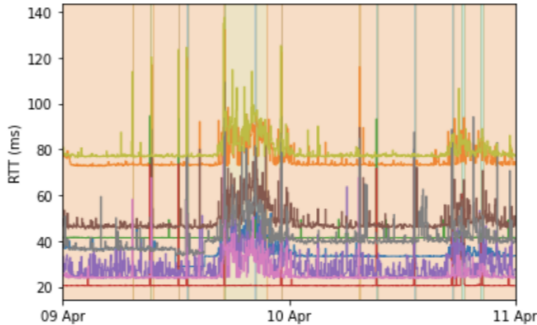


Fig. 8. Segmentation of the series after the second Viterbi smoothing

In order to quantify the smoothing effect and measure the quality of the segmentation, we have chosen to use the adjusted Rand index. Indeed, the Rand index is a measure of similarity between two partitions of the same set and it represents the proportion of pairs of points that are grouped in the same way in the two partitions [18]. This index will allow us to decide the best variant of the method to choose in terms of the Viterbi version and the number of smoothings to use.

### III. EVALUATION

#### A. Datasets

We will test our methods on different real-world datasets. The datasets are collected from Ripe Atlas. They represent delays between pairs passing through specific IXPs, DE-CIX, and AMS-IX respectively. For these datasets, we use

as ground truth approximation for the segmentation an HDP-HMM method [19] that was validated and adapted by Ripe Atlas in an API [20]. In fact, Ripe Atlas is an Internet measurement platform created by the RIPE Network Coordination Centre (RIPE NCC). The platform enables users to measure and analyze Internet performance from multiple perspectives, providing valuable insight into the state of the Internet and its various components [21]. The different datasets are described more in detail below:

1) *Real dataset 1 (RD1), RTT traces passing through DE-CIX*: Between April 9th and April 20th 2018, DE-CIX Frankfurt experienced a disruption of its network connectivity to route servers, resulting in rerouted traffic and an interruption of traffic. This was evidenced by an analysis of the rates of BGP updates received by route collectors located at DE-CIX, which dropped close to zero between 19:43 and 23:28 on the 9th of April, and between 02:02 and 03:51 on the 10th of April, as reported in [22].

This dataset contains 38k pairs passing through DE-CIX. Each of these time series has 720 timeslots which correspond to two days of measurements, the 9th and 10th of April 2018. This dataset will be denoted RD1.

2) *Real dataset 2 (RD2), RTT traces passing through AMS-IX*: On May 13th, 2015, AMS-IX experienced a seven-minute, two-second partial outage due to a switch interface generating looped traffic on the peering LAN. This resulted in some peers at the exchange losing their BGP session, as reported by [23]. The outage lasted between 10:22:12 and 10:29:14 UTC before the switch interface was disconnected. This dataset contains 26k pairs passing through AMS-IX. Each has 360 timeslots corresponding to the 13th of May 2015. This dataset will be denoted RD2.

#### B. Baseline method

To our best knowledge, HDP-HMM (Hierarchical Dirichlet Process Hidden Markov Models) is state of the art for efficient segmentation of HMMs with unknown number of states and unknown emission distributions that are modeled as Gaussian mixtures (with unknown number of components).

We compare the successive treatments proposed with HDP-HMM segmentation: the hierarchical clustering alone (HC), HC with one Viterbi smoothing (V1) and HC with two Viterbi smoothings (V2) and finally HC with one regularized Viterbi smoothing (V1 REG) and HC with two regularized Viterbi smoothings (V2 REG). V2 version of Viterbi smoothing introduces a re-estimation of the emission densities after the first smoothing.

#### C. Delay time series clustering

Hierarchical clustering is employed to group time series with similar and homogeneous patterns. The optimal clustering is obtained by determining a threshold for cutting off the dendrogram. A low threshold leads to simple clusters with highly similar patterns, yet a large number of clusters; thus, failing to fully capitalize on the similarity and dimensionality reduction. A high threshold, on the other hand, compresses the data, yielding a low number of clusters, but also presenting

the risk of combining time series with distinct patterns within the same group, making it difficult to achieve an accurate multivariate segmentation. To maximize the correlation within each cluster, a threshold is chosen that lies between these two extremes.

In this section, we are going to explore the impact of the different linkage methods on the clustering quality. We focus in particular on the comparison between Ward and complete linkage along with defining the optimal dendrogram cut-off threshold.

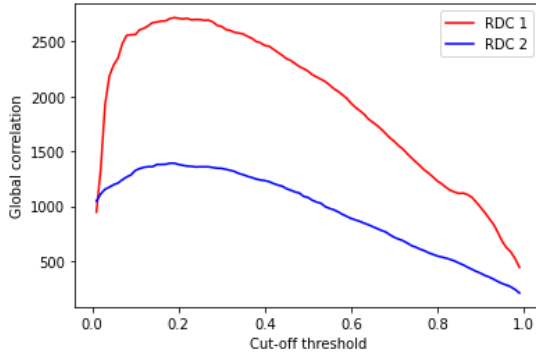


Fig. 9. Evolution of the global correlation with the dendrogram cut-off threshold

We can see in figure 9 that for RDC1 the optimal cut-off threshold is 0.2. When we cut the dendrogram at this value, we obtain the distribution given in table II and figure 10. We can clearly notice that a large part of the data remained unclustered with 16200 clusters containing one time series. This being said the rest of the data were clustered in a homogeneous way, with the majority of clusters containing between 2 and 50 time series.

TABLE II  
NUMBER OF CLUSTERS HAVING 5 TIME SERIES OR LESS

Nb of time series	1	2	3	4	5
Nb of clusters for RD1	16200	1393	544	316	180
Nb of clusters for RD2	5006	699	321	230	119

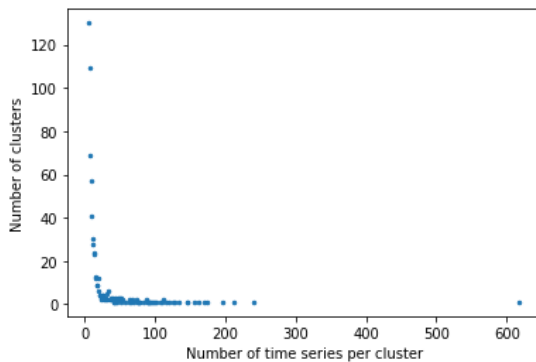


Fig. 10. Number of cluster having more than 5 time series for RD1

We should note that a maximum number of time series per cluster is 617 as can be seen in figure 11. We can see that the

clustering detects the pertinent common patterns despite the presence of noise and outliers. Figure 12 shows on the other hand a cluster of time series with high stability in time. 11

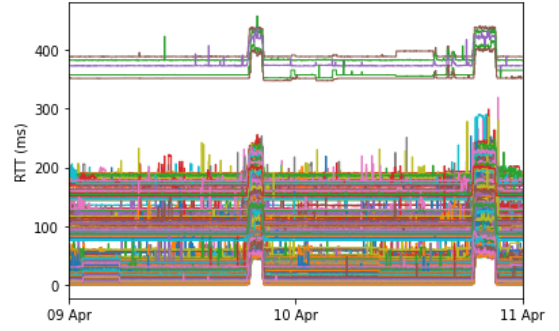


Fig. 11. Cluster from RDC1 using complete linkage having 617 time series

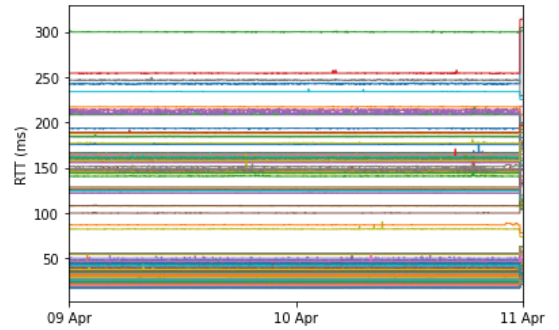


Fig. 12. Cluster from RDC1 using complete linkage having 162 time series

Figures 13 and 14 on the other side, presents two clusters with very similar patterns. We should notice that despite the similarity displayed between these clusters, the complete linkage separated them while they could have been grouped into one larger cluster. This result goes along with one of the cons of complete linkage described in table I which is the tendency to break large clusters.

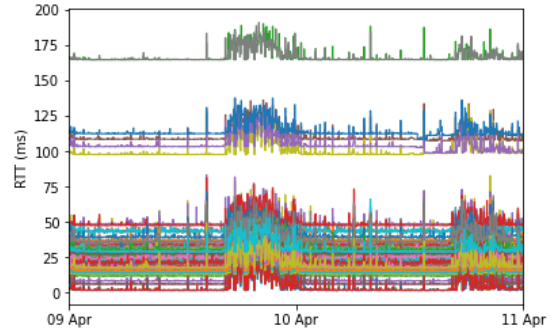


Fig. 13. Cluster from RDC1 using complete linkage having 115 time series

In figure 15 we can see the dendrogram of the Ward linkage. On the x-axis we see the number of time series per cluster, knowing that each vertical line refers to an independent cluster. On the y-axis, we can see the distance between these clusters

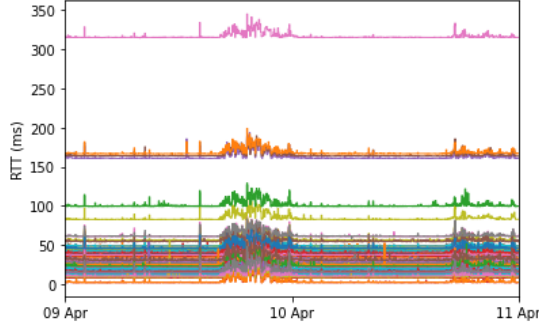


Fig. 14. Cluster from RDC1 using complete linkage having 108 time series

using the Ward linkage. In this case, we can see that the clustering remains over the interval 1 to 10. Starting from 10 the process of agglomeration starts again. In order to have a clustering from this stable zone, the cut-off value was decided to be 8, leading to 30 clusters. It is optimal since it is the value that ensures a good trade-off between the number of clusters and the degree of their compression.

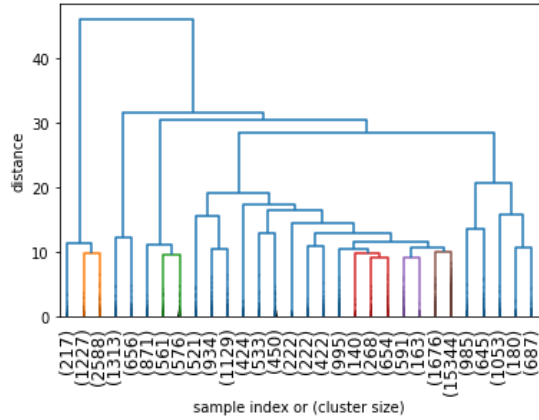


Fig. 15. The dendrogram of the hierarchical clustering of the delay time series

One difference that we can outline compared to the complete linkage, is the fact that the number of time series per cluster is very high. Moreover, in figures 17 and 18, we can see that the pattern that was split using the complete linkage in figure 16 is totally grouped into one cluster of 1227 time series using the Ward method.

In figures 18 and 17, we can notice that one drawback of the compression power of Ward is the fact that it will group together time series having a similar baseline, but not the same patterns. This is the case when the pattern in question is repeated peaks as we can see in figure 17). The complete linkage, on the contrary, didn't group these time series even though they have a common baseline behavior since they don't display a similar pattern change. Thus we see that tending to group series with a common baseline referred to as globular bias in table I is more present with the Ward method.

A possible approach to take advantage of the pros of both methods is to first cluster the data using Ward linkage, then

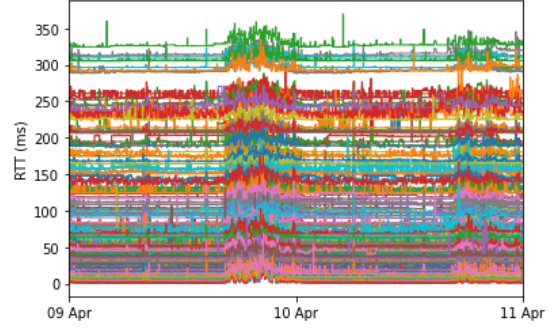


Fig. 16. Cluster containing 1227 time series in RD1 obtained using Ward linkage

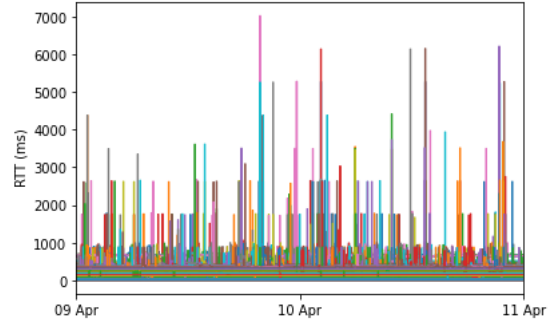


Fig. 17. Cluster from RDC1 using complete linkage having 500 time series

once the homogeneous clusters are identified, we can treat them using segmentation for example. The rest of the data, that was grouped can be considered as a new dataset and clustered using complete linkage for a finer grouping in order to have more homogeneous clusters later.

After the analysis of the RDC1, the same results were observed on RDC2. The cut-off threshold was set as well to 0.2 for the complete linkage and to 12 for the Ward method. In the figure and table, we can notice that complete linkage provided a similar distribution aspect of the number of time series per cluster. Furthermore, this clustering helps as well to explore the various patterns of delays along the global dataset.

*The spatial correlation within the clusters* An analysis was conducted on each cluster to investigate possible spatial correlations. Specifically, for the RDC2 dataset, we selected clusters with more than 10 time series, which represents a set with 3200 clusters, and determined the cardinality of the set of sources and destinations for each cluster. As shown in Figure 19, a high number of clusters has series with only a small number of sources or destinations. This suggests that spatial correlations can often be related to measurements with a common source or destination and potentially share common routes.

#### D. Viterbi smoothing

We should note that the colors of the segments in the following figures refer to different states within the same segmentation, but are independent of one figure to another.

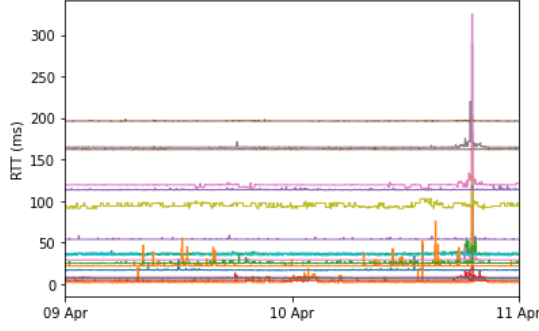


Fig. 18. Cluster from RDC1 using complete linkage having 150 time series

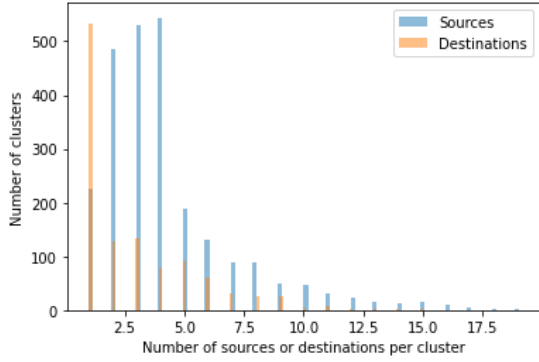


Fig. 19. Repartition of the number of common sources and destinations per cluster

In this section, we evaluate the impact of some parameters on the smoothing in adequacy to the stationarity of the time series that corresponds to the sequence of states for each time serie. First, we show the impact of  $\epsilon$  on the transition between states.

For values of  $\epsilon$  ranging from  $1e5$  to  $1e2$ , the distribution state doesn't change. An example of this can be seen in figures 20 and 21, where we can clearly notice that there is little difference between the state distribution for  $\epsilon = 1e2$  on the left and  $\epsilon = 1e5$  on the right. The state transitions remain consistent which indicates that the assumption regarding stability within the same state is validated in this model.

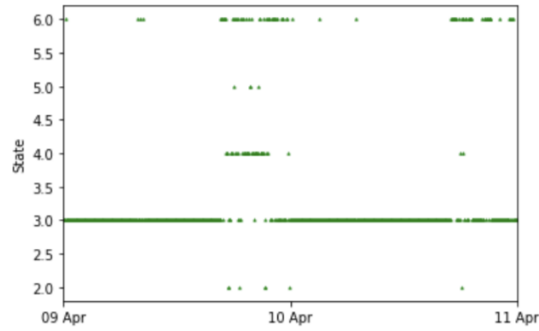


Fig. 20. States after the first Viterbi smoothing for  $\epsilon = 1e-2$

For the following experiences, we fix  $\epsilon$  to 0.001 for the first

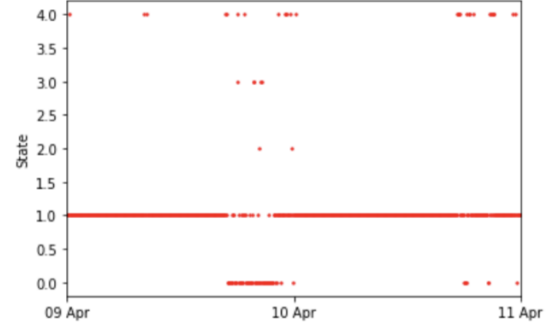


Fig. 21. States after the first Viterbi smoothing for  $\epsilon = 1e-5$

smoothing of Viterbi and to 0.0001 for the second smoothing, since we consider that the states are more stable after the smoothing leading to smaller transition probabilities between states.

We will now evaluate the segmentation quality and the impact of the number of time series in the cluster on both the quality of segmentation and the execution time. For this purpose, we compare the segmentation of each cluster with the segmentation of a representative of that cluster using both our method and the method based on the HDP-HMM.

*Comparison with the HDP-HMM method: execution time* We measured the processing time of the series per cluster for the method based on the histograms. We can see that a hierarchical plus Viterbi approach offers processing gain by a factor of 9 compared to HDP-HMM, as shown in Table III. The values given in the table are mean values for ten experiments and time series are 720 timeslots long.

TABLE III  
COMPARISON OF THE EXECUTION TIME IN SECONDS

Method	Hierarchical + Viterbi	HDP-HMM
Execution time in seconds	0.28	2.6

*Effect of the smoothing on the number of states* In figure 22 we show a summary of multivariate emission probabilities before and after the Viterbi smoothing. This summary is obtained by gathering data in all dimensions and computing the compounding kernel density estimator. We can notice that the number of states diminishes throughout the process passing from 12 to 4 states. This illustrates the smoothing effect that happens since the model adopted for the transition probabilities favors stability within one state instead of transitions. This enables us to get rid of states with low probability and replace them with more dominant ones in the segment.

*Effect of the smoothing on the segmentation* Moreover, the impact of Viterbi smoothing is also very clear in terms of segmentation quality. In fact, in figure 23, we can notice that the segments become more homogeneous from figure 23 (a) to 23(c). The cluster represented in figure 23 presents roughly two major states: one that is stable and one that has a high variance. At the end of the process in figure 23(c) we can see that the stable state (in green) is well segmented, and the unstable state on the other hand is separated into two segments. This segmentation makes sense since the delay distribution

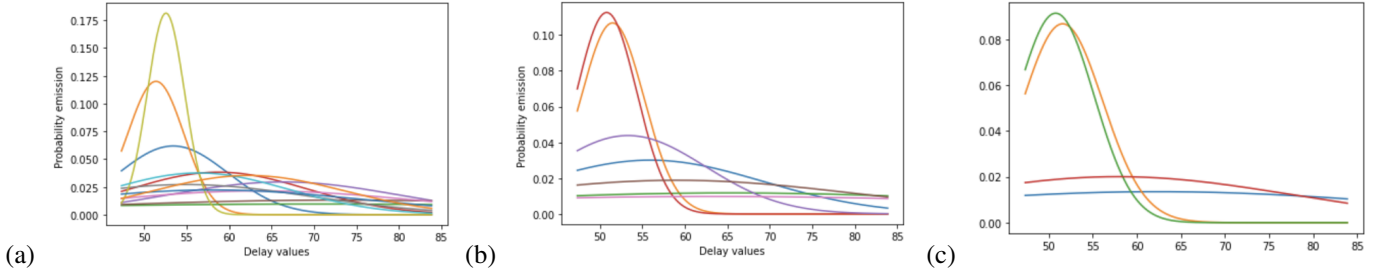


Fig. 22. The emission probabilities for each state after (a) hierarchical clustering (12 states), (b) first Viterbi smoothing (7 states) (c) second Viterbi smoothing (4 states) for a time series from RDC1.

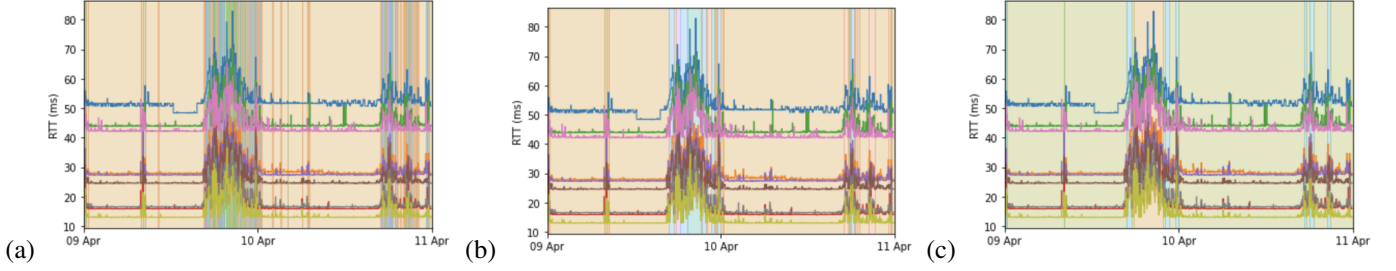


Fig. 23. Segmentation result after hierarchical clustering (a), one (b) and two (c) Viterbi smoothings. We respectively get 12, 7 and 4 states. The cluster contains 10 time series from RDC1.

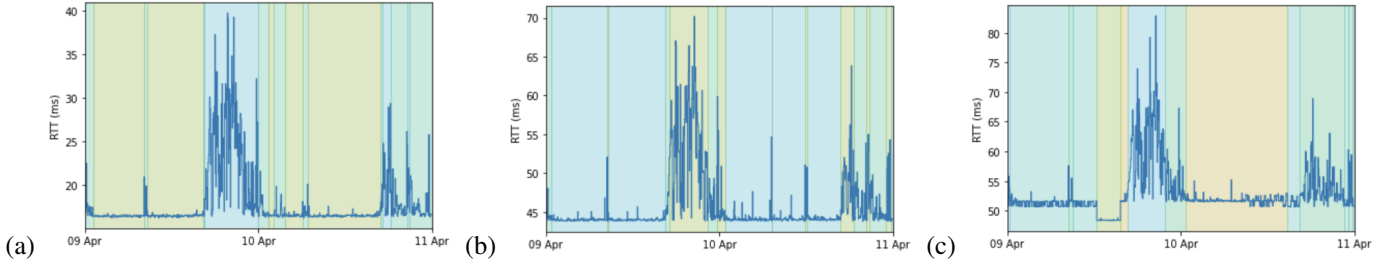


Fig. 24. Segmentation of three time series from the cluster in Fig. 23 using HDP-HMM. 3 states are found for each time series.

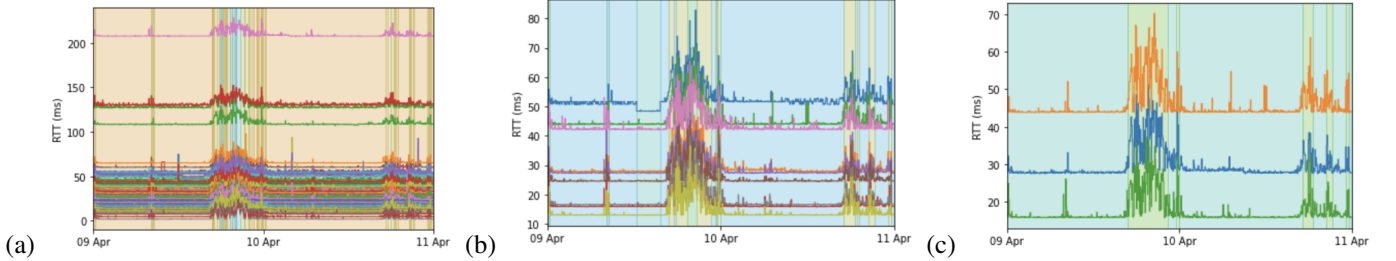


Fig. 25. Impact of the number of time series on the segmentation quality, cluster with 130 (a), 10 (b) and 3 (c) time series.

is homogeneous within the blue segments while presenting higher values and a maximum peak in the yellow one.

*Global patterns VS local patterns* When segmenting jointly a high number of series, we can notice the fact that local patterns that are specific for each time series tend to disappear from the segmentation result of the whole cluster. For instance, many local peaks first detected with hierarchical clustering in figure 23(a) were merged to a more global state (in green) in figure 23(c). This phenomenon can also be noticed in univariate segmentation with the HDP-HMM method in figure 24 where we can see that in 24 (a) and 24 (b) visible peaks that are small compared to the overall time series variation level are segmented along with the stable baseline of the time series. But

we should note however, that for both our approach and the HDP-HMM, if an isolated peak or pattern is visible enough it still can be detected by being in a separate segment as is the case in figures 24 (b) and 23(c).

*Effect of the number of time series* In figure 25 we highlight the impact of the number of time series inside the cluster on the segmentation quality. In fact, as we can see in figure 24, the segmentation of three time series that belong to the same cluster using the HDP-HMM is different despite the similarity. This means that each time series has its own specificities that can be related to the path taken or to other network properties.

When segmenting jointly a high number of time series as is the case in 25 (a), the smoothing effect wasn't very efficient

compared to figure 23. We can see that for a smaller number of time series, 10 in 25 (b) and 3 in 25 (c), the segmentation is better smoothed and is closer to the one given by the HDP-HMM.

*Effect of the Viterbi regularization on clusters with high cardinality* We used the average adjusted Rand index of time series in a cluster as a function of its cardinal for the four variants of the method. It is computed between the multivariate segmentation given by the cluster and the univariate segmentation given by the HDP-HMM for each series in this cluster. It has been shown that the contribution of the post-processing steps always improve the quality of segmentation compared to hierarchical clustering alone. It has also been observed that the Viterbi regularized version with one run shows the best results. Therefore, when treating jointly a high number of time series we choose the regularized version with only one pass of Viterbi as the segmentation strategy.

#### IV. CONCLUSION

This paper suggests a multivariate segmentation approach for network delays. It takes advantage of the spatial correlation between pairs of a network and their temporal stability. The method is based on hierarchical clustering and on the Viterbi algorithm. The hierarchical clustering for time series enabled us to group a large number of series having similar patterns. We have seen that the linkage method adopted has an impact on the homogeneity of the patterns within the same cluster. Complete linkage tends to give more homogeneous clusters but does not have a big compression power, whereas Ward linkage gives clusters with a large number of elements. The spatial correlation within clusters has also been proved, since most clusters show the presence of common sources or destinations. When the clusters are defined, the process of segmentation begins by applying again the hierarchical clustering followed by the Viterbi algorithm. We have shown the impact of Viterbi smoothing on the enhancement of the segmentation quality. Its results are similar to the ones given by HDP-HMM. Finally, we have shown that the computational burden is reduced by a factor of 9 compared to HDP-HMM. In future work, we will investigate more in depth the reasons behind the spatial correlation based on the network topology by using traceroutes measurements. Next, we will explore spatial correlation as a tool for dimensionality reduction for some smart monitoring schemes by considering only a few representatives of the cluster. Moreover, we will apply this method to a real-world use case such as Internet exchange points incidents. It will be interesting as well to cross-reference our segmentation results with the routing information in the perspective of performing root-cause analysis.

#### REFERENCES

- [1] M. Mouchet, S. Vaton, and T. Chonavel, "Statistical characterization of round-trip times with nonparametric hidden markov models," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2019, pp. 43–48.
- [2] Y. Schwartz, Y. Shavitt, and U. Weinsberg, "A measurement study of the origins of end-to-end delay variations," in *Passive and Active Network Measurement Conference*, 2010.
- [3] H. Pucha, Y. Zhang, Z. M. Mao, and Y. C. Hu, "Understanding network delay changes caused by routing events," *ACM SIGMETRICS performance evaluation review*, vol. 35, no. 1, pp. 73–84, 2007.
- [4] Does the internet route around damage? - edition 2021. [Online]. Available: <https://labs.ripe.net/author/emileaben/does-the-internet-route-around-damage-edition-2021/>
- [5] Caida. [Online]. Available: <https://www.caida.org>
- [6] W. Zhang and J. He, "Modeling end-to-end delay using pareto distribution," in *Second International Conference on Internet Monitoring and Protection (ICIMP 2007)*. IEEE, 2007, pp. 21–21.
- [7] M. Mouchet, S. Vaton, and T. Chonavel, "Poster abstract: A flexible infinite hmm model for accurate characterization and segmentation of rtt timeseries," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 1055–1056.
- [8] L. Davisson, J. Jakovleski, N. Ngo, C. Pham, and J. Sommers, "Re-assessing the constancy of end-to-end internet latency," in *IFIP Network Traffic Measurement and Analysis Conference*, 2021.
- [9] W. Shao, J.-L. Rougier, A. Paris, F. Devienne, and M. Viste, "One-to-one matching of rtt and path changes," in *2017 29th International Teletraffic Congress (ITC 29)*, vol. 1. IEEE, 2017, pp. 196–204.
- [10] W. Zhang and J. He, "Modeling end-to-end delay using pareto distribution," in *Second International Conference on Internet Monitoring and Protection (ICIMP 2007)*, 2007, pp. 21–21.
- [11] Y. Sato, S. Ata, I. Oka, and C. Fujiwara, "Using mixed distribution for modeling end-to-end delay characteristics," in *Proceedings of the 8th Asia-Pacific Network and Management Symposium (APNOMS 2005)*, Japan, Okinawa, 2005.
- [12] J.-A. Hernández and I. W. Phillips, "Weibull mixture model to characterise end-to-end internet delay at coarse time-scales," *IEE Proceedings-Communications*, vol. 153, no. 2, pp. 295–304, 2006.
- [13] D. Perdice, J. E. L. de Vergara, and J. Ramos, "Deep-fda: Using functional data analysis and neural networks to characterize network services time series," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 986–999, 2021.
- [14] X. Liu, X.-H. Zhu, P. Qiu, and W. Chen, "A correlation-matrix-based hierarchical clustering method for functional connectivity analysis," *Journal of neuroscience methods*, vol. 211, no. 1, pp. 94–102, 2012.
- [15] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: an overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012.
- [16] S. Sharma, N. Batra *et al.*, "Comparative study of single linkage, complete linkage, and ward method of agglomerative clustering," in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. IEEE, 2019, pp. 568–573.
- [17] H.-L. Lou, "Implementing the viterbi algorithm," *IEEE Signal processing magazine*, vol. 12, no. 5, pp. 42–52, 1995.
- [18] Rand index. [Online]. Available: [https://en.wikipedia.org/wiki/Rand\\_index](https://en.wikipedia.org/wiki/Rand_index)
- [19] M. Mouchet, S. Vaton, T. Chonavel, E. Aben, and J. Den Hertog, "Large-scale characterization and segmentation of internet path delays with infinite hmms," *IEEE Access*, vol. 8, pp. 16 771–16 784, 2020.
- [20] Api atlas. [Online]. Available: <https://github.com/maxmouchet/atlas-trends-demo>
- [21] Ripe atlas. [Online]. Available: <https://atlas.ripe.net/m>
- [22] Does the internet route around damage in 2018? [Online]. Available: <https://labs.ripe.net/author/emileaben/does-the-internet-route-around-damage-in-2018/comments>
- [23] Does the internet route around damage? a case study using ripe atlas. [Online]. Available: <https://labs.ripe.net/author/emileaben/does-the-internet-route-around-damage-a-case-study-using-ripe-atlas/>