ON THE MULTI-FRACTAL NATURE OF OBSERVED IP ADDRESSES IN MEASURED INTERNET TRAFFIC

by

WALTON G. O'CONNOR

A THESIS

Presented to the Department of Computer Science
and the Division of Graduate Studies of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Master of Science

March 2023

THESIS APPROVAL PAGE

Student: Walton G. O'Connor

Title: On the Multi-Fractal Nature of Observed IP Addresses in Measured Internet Traffic

This thesis has been accepted and approved in partial fulfillment of the requirements for the Master of Science degree in the Department of Computer Science by:

Reza Rejaie Chair

and

Krista Chronister

Original approval signatures are on file with the University of Oregon Division of Graduate Studies.

Degree awarded March 2023

THESIS ABSTRACT

Walton G. O'Connor

Master of Science

Department of Computer Science

March 2023

Title: On the Multi-Fractal Nature of Observed IP Addresses in Measured Internet Traffic

We examine the presence of multifractal properties in the spatial structure of observed IPv4 addresses in measured Internet traffic. A collection of traffic samples from a variety of network settings are assembled and their spatial structures evaluated for multifractal properties using the **method of moments** approach. We show that all collected traces have properties consistent with multifractal scaling, but that the scaling behaviors vary by trace. We propose mechanisms which may give rise to these behaviors, and then discuss a number of ways by which our empirical finding concerning the spatial structure of observed IP addresses in measured network traffic can be utilized in practice, including its use in modern dataplane network monitor settings, both as a metric to monitor and as a means to increase hardware utilization efficiency.

ACKNOWLEDGEMENTS

I am deeply indebted to Dr. Reza Rejaie and Chris Misa for patiently mentoring me and putting up with my antics for the better part of four years. I have grown as a programmer, an academic, and most importantly as a person under your guidance. Additionally, this would not have been possible without Dr. Walter Willinger, who took an inordinate amount of time out of his life to get the ball rolling on this project and to make sure I was applying mathematics correctly. Finally I would like to thank Dr. Arpit Gupta and his students at UCSB, who kindly took the time to provide access to their network traffic for this project.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Page

Figure Page

LIST OF TABLES

CHAPTER I

INTRODUCTION

Current work in network automation promises that the ultimate goal of a self-driving network is contingent on the ability to concurrently execute hundreds or thousands of network telemetry queries that collect key information about the network state in a timely manner. Collected information serves as the primary input in to a bevy of network management systems designed to detect, characterize, and counteract all manner of network related problems, in both the security and performance domains, at scale, and near real-time. This trend is made possible by strides in hybrid network telemetry architectures that back line rate hardware telemetry collection (implemented in NICs or programmable switches) with flexible and scaleable userspace stream processors.

A key limitation of dataplane telemetry hardware is that resources (namely memory, which is typically implemented as expensive SRAM in hardware), are very limited and hotly contested, and hardware runtime reconfigurability and runtime programmability are underwhelming at best compared to the possibilities in software. Consequently, running queries in the dataplane at scale and in real time poses a persistent challenge, especially given the dataplane's preexisting duties as the backbone of the network. Prior efforts on scaling network telemetry queries have focused mainly on attempting to stretch a limited set of dataplane capacity and capabilities as far as possible to answer a set of telemetry queries to a satisfactory precision while ensuring that packets continue to get forwarded at line rate. We propose an alternative approach to scaling dataplane telemetry queries where, rather than trying to fit more queries on to the same hardware, we instead opt to take advantage of pronounced spatial structures present in the network traffic to reduce the workload required to satisfy the queries being submitted.

1.1 Traffic of Interest: Definition and Examples

The observation that the traffic required to answer a telemetry query forms a fraction of the overall traffic is mostly intuitive, with telemetry operators often drawing comparisons between telemetry queries and “finding needles in a haystack”. To put concrete numbers to this observation, in the case of one real world intrusion detection system (IDS), 99% of the traffic processed by the IDS was found to be benign, and 1% or less triggered an alert or was flagged by the IDS. For a given query or set of queries, we define the associated “traffic of interest” (**ToI**) to be a subset of the overall packet traffic that satisfies those queries’ desire for data (i.e. it is the minimum set of packets required for the query to arrive at the same result that it would have given using the full set of packets). In the context of the IDS example, the ToI would be the set of traffic that the IDS identifies as malicious.

1.1.1 Examples of Traffic of Interest. The size and content of the ToI is highly dependent on the nature of the query it satisfies. On one end of the spectrum, consider the ToI for something specialized like a DNS reflection attack detecting query. The query is really only interested in certain DNS packets, so its ToI would be a subset of all DNS packets. The number of DNS packets is pretty minuscule compared to the overall volume of packets, so in this case the ToI is indeed a needle and the haystack of non-DNS packets can be ignored. On the other end of the spectrum, there are a number of very nonspecific queries that would like to view huge portions of the traffic on a network: the simplest one of these would just be the null query, which returns every packet observed at the collection point.

More realistic examples include queries examining all outbound or inbound packets, queries examining all TCP traffic and so on. In these instances, the desired packets comprise such a large portion of the overall traffic that the proverbial needle is now effectively the haystack itself. For a final example of ToI, consider a query examining internet background radiation (i.e. packets destined for an inactive address assigned to an active network). As this query only needs to examine data headed towards IPs with no machine behind them, the query could, for instance, ignore any TCP connections that successfully completed the handshake. Detecting this is nontrivial, but it illustrates the flexibility inherent to our definition of ToI.

1.2 Utility of ToIs to Telemetry

As most packets encountered by the dataplane are simply irrelevant to the queries they are running, focusing the hardware on the ToI saves precious dataplane resources. For example, the available state space for modern dataplane telemetry systems constrains the volume of flows or connections they can track, but filtering mechanisms taking advantage of multifractal clustering properties can substantially ease this constraint for certain classes of query. Finding the true set of ToI itself can often only be done once the results of a telemetry query are already known, but the notion of ToI as an ideal query input to strive for remains, and there are a variety of heuristics that can be used to carve subsets out of the traffic data that contain ToI, where the goal of these heuristics is to maximize the portion of traffic being fed into a query that is ToI. One such approach for filtering ToI from background traffic is to try and identify addresses that are generating ToI and exclusively examining their traffic. This approach can reduce the volume of traffic that needs to be examined by the dataplane substantially when the number of ToI-generating addresses is a small fraction of the overall traffic (like in the IDS case presented above), and importantly, filtering by address or subnet is a nearly ubiquitous capability among hardware telemetry systems.

1.2.1 Background on Switch Based Telemetry Systems.

Many existing network telemetry systems have converged on expression their queries as a sequence of dataflow operators (e.g., `map`, `filter`, `reduce`, `distinct`) applied over an incoming windowed stream of packets with some fixed window duration. Sonata [10] provides a set of queries that have become a sort of baseline to benchmark dataplane telemetry systems. A fairly typical example of such a query can be seen in Sonata's implementation of the superspreader query: Hosts, identified by their `sourceIP` that send traffic to more than Th distinct destinations in a window of size W seconds are flagged as superspreaders. Executing such a query requires making three decisions: (1) which operator to execute during which period in time, (2) which subset of the traffic (identified by some key, such as `sourceIP`) to process and for which operators, and (3) how much memory to allocate to each operator for the window. In this scenario, imagine a piece of hardware that concurrently runs dataflow operators over sets of packets belonging to different segments of the IP space, and each operator required some amount of memory be allocated to it. The naive approach would be to give the operator running on each segment of the IP space an equal share of memory. This approach works if ToI generating IPs are distributed evenly between every prefix, but on the other hand the system breaks down once any individual prefix outgrows its allocated memory, even if the overall volume of data could easily fit in the memory. A more intelligent system would be aware that the prefixes containing ToI-generating IPs are often not distributed evenly, and would instead have some mechanism to track which prefixes generated ToI for each past query and allocate memory to operators accordingly.

1.2.2 The Case for Leveraging Address Distributions for the Dataplane.

Preliminary empirical evidence points towards source IPs not being distributed uniformly across the IP space [11][4]. [11] consider traffic consisting of uni-directional traffic extracted from bidirectional traffic seen on different links or at different routers within different networks, and [4] examined traffic consisting of internet background radiation traffic collected from an access link to a university or provider network containing unused addresses within its allocated IPs. The work by Kohler et al.[11] was the first to examine the spatial structure of observed IP addresses in measured internet traffic. Analysis of hours of measured packet traces from different network locations indicated that each trace showed signs of *multifractal* scaling behavior in the distribution of source IP addresses. Monofractal scaling defines a scale invariant behavior wherein clusters at a given scale level (like /16) get divided into proportionally sized subclusters at finer scales (like /20), irrespective of where the clusters are situated in the IPv4 space. Multifractal scaling is a generalization of this wherein the scaling behavior can vary for different parts of the IPv4 space, and the parameters dictating this "local" scaling behavior are summarized by a function called the multifractal spectrum. Both give rise to a pronounced qualitative "cluster-within-cluster" structure that results in observed IP addresses being distributed very unevenly across the IPv4 space. Subsequently, Barford et al.[4] analyzed multi-day long datasets that consist of source IP addresses in packet traces collected by network telescopes, firewalls and IDSs distributed throughout the internet and reported characteristics of the observed IP address spaces that are consistent with multifractal scaling. The IP space clustering behavior of multifractal traffic may be of substantial benefit to ToI finding heuristics that filter over the IP space.

CHAPTER II

CONTRIBUTION

The studies discussed were first reported two decades ago, and there has been little progress made to establish a rigorous foundational basis for the empirical finding that the structure of observed IP addresses in measured internet traffic demonstrates multifractal scaling. Compared to the commonly-accepted and well-established empirical finding that the temporal structure of measured internet traffic demonstrates self-similar scaling, the reported multifractal findings have received little attention, are neither widely accepted or known, and lack a generally accepted or carefully validated explanation. This issue is summed up by Kohler et al.[11]: "Without a convincing description of how [multifractal] address structure arises, the results of the explorations [in this paper] must be considered preliminary.". We argue there are additional reasons these early empirical reports have resulted in little to no follow-up work and the suggested multifractal behavior has been largely viewed as a curiosity.

First, these findings are somewhat dated, to put it gently. There is a pressing need to use modern traffic and perform a rigorous analysis for a range of current datasets from a range of environments. Second, many existing approaches to multifractal analysis are subject to technical assumptions that are difficult to prove in practice, and coping with this requires exploiting the large numbers of observations that come with typical network measurement by applying statistical techniques to carry out a robust assessment of the macroscopic aspects of multifractal scaling behavior, even if these techniques are inadequate to examine finer grained details. Third, any convincing physical description of how the multifractal structure emerges should be backed by supporting measurements that illustrate the validity of the mechanisms that underly the provided physical explanation. Ideally, such a description should translate directly into parametric generative models that can be used to obtain synthetic traffic that emulates the observed multifractal structure of the IP addresses in the measured traces. Fourth, we are currently lacking concrete use cases that demonstrate whether or not accounting for the observed spatial characteristics of real-world instances of traffic is important for downstream applications, and if so, for what type of such downstream applications. Lastly, in case the multifractal address structure in measured instances of traffic is shown to matter in practice, we need new ways to synthesize instances of traffic that can be shown to exhibit the full-fledged spatial-temporal characteristics that are consistent with real-world traffic and includes both the temporal and spatial aspects of measured internet traffic.

We focus on addressing the first two items, present preliminary results for items three and four, and briefly discuss possibilities for approaching item four. In particular we make the following contributions:

- We amass a repository of different instances of traffic datasets derived from a range of different measured internet traffic traces. The current instances differ in terms of the location where the traffic was collected, traffic volume, duration, user base, application mix, etc. We are constantly adding new instances of traffic as they become available to us. (§ III)

- To infer the presence or absence of multifractal address structure in each of the existing real-world instances of traffic, we apply a statistical method known as "method of moments". Compared to other, more geometrical methods for inferring multifractal structure, the method of moments is known to be more robust with respect to aspects of the data that may violate certain technical conditions other methods require but are in general difficult if not impossible to verify in practice, computationally more tractable than other methods, and especially well-suited for datasets with large number of observations. We show that each of the traffic samples exhibits a spatial address structure that is consistent with multifractal scaling but the degree of multifractality depends on the type of datasets. (§ IV, § V)

– Building on an earlier attempt at providing a physical explanation or description of how multifractal address structure may arise in real-world traffic, we offer some initial empirical evidence in favor of a generative mechanism in the form of an underlying multiplicative conservative cascade process and demonstrate its potential for producing synthetic traffic with a desired multifractal address structure. (§ VI)

– Prior studies that report on empirical findings of multifractal address structure in measured internet traffic provide or discuss no concrete use cases that demonstrate the importance of adequately accounting for this newly observed characteristic. We review the few studies in the existing literature that report on such concrete uses and provide evidence that the observed nonuniformity in how observed addresses are distributed across the IPv4 space can be put to good use in practice. In particular, these use cases argue for more aggressive future efforts that are aimed at adequately accounting for the full-fledged spatial-temporal characteristics of real-world traffic in guiding telemetry systems to optimize their resource usage by examining less non-ToI traffic. (§ VII)

CHAPTER III

DATASETS

Previous work exploring the multifractal properties of network traffic is decades old [11][4], and the volume and nature of real internet traffic has undergone tectonic shifts since those experiments were run. Consequently, further examinations of multifractal properties benefit tremendously from samples of the “modern” internet. Furthermore, examining how these properties vary over time, in the vein of [18] is valuable, particularly examining how the behavior evolves over hours, days, months, and years. To this end, we assemble a collection of publicly available datasets (CAIDA[1] and MAWILab[8]), alongside a number of privately collected samples for university campus border switches.

3.1 Collected Traffic Traces

A variety of traffic samples were evaluated Table 1, spanning a number of dates, times, and locations. For every dataset, only the first 100,000 observed source IPs are considered, which was found to not substantially impact the results while making head to head comparison of different datasets much easier.

In Table 1, CAIDA refers to the CAIDA 2019 anonymized internet trace, which is recorded from an internet backbone [1], MAWI refers to the MAWILab traffic anomaly archive [8], which is recorded at the border between the WIDE project and its parent ISP. UCSB refers to captures of network traffic at the edge of UC Santa Barbara’s campus network. UO refers to captures of network traffic at the edge of the University of Oregon’s campus network. Data from both campuses are anonymized with CryptoPAn [19] prior to analysis. This prefix-preserving anonymization method exactly preserves the structure being analyzed such that running the analysis before and after anonymization yields an identical result.

These datasets represent network traffic from campus and network backbone environments, which covers two very common network settings. Data spanning seven years is included, and traces were captured on a variety of times of days, days, and months. This all serves to forward the argument that what we observe is not isolated to a specific time, place, or circumstance, but is rather a fundamental property of organic internet traffic.

Name	Date	Duration	Type	IP Count	Packet Count
CAIDA-dir-A	2019-01-17 5:00AM	900s	pcap	2,381,306	566M
CAIDA-dir-B	2019-01-17 5:00AM	900s	pcap	3,863,987	1.23B
MAWI-20150202	2015-02-02 2:00PM	900s	pcap	5,571,625	99M
MAWI-20150710	2015-07-10 2:00PM	900s	pcap	4,415,599	191M
MAWI-20151002	2015-10-08 2:00PM	900s	pcap	4,818,370	135M
MAWI-20180316	2018-03-16 2:00PM	900s	pcap	4,567,614	69M
MAWI-20180807	2018-08-07 2:00PM	900s	pcap	4,635,311	73M
MAWI-20181107	2018-11-07 2:00PM	900s	pcap	4,677,191	80M
MAWI-20190901	2019-09-01 2:00PM	900s	pcap	4,689,835	87M
MAWI-20210110	2021-01-10 2:00PM	900s	pcap	265,794	52M
MAWI-20210614	2021-06-14 2:00PM	900s	pcap	180,532	74M
MAWI-20211212	2021-12-12 2:00PM	900s	pcap	149,572	55M
UCSB-20220428	2022-04-28 12:15	900s	pcap	194,498	1.02B
UCSB-20220921	2022-09-21 19:25	900s	pcap	112,164	1.05B
UCSB-20221205	2022-12-05 20:15	900s	pcap	186,575	1.1B
UO-20181106	2018-11-16 00:00	1 day	netflow	1,805,085	9B
UO-20190829	2019-08-29 00:00	1 day	netflow	1,497,311	12B
UO-20200213	2020-02-13 00:00	1 day	netflow	786,607	36B
UO-20211106	2021-11-06 00:00	1 day	netflow	1,288,775	26B
UO-20220517	2022-05-17 00:00	1 day	netflow	668,817	23B

Table 1. List of traffic samples

CHAPTER IV

METHODOLOGY

Let μ denote the measure (defined on the IPv4 space) that counts the number of observed IP addresses in a given dataset. Determining if this measure shows multifractal properties is relatively straightforwards at a high level: compute the fractal dimension at a variety of scaling factors, and see if it changes between scaling factors. If if does change, then μ is multifractal, and if it doesn't change, it is monofractal; if μ has fractal dimension 1 (same as the underlying IPv4 space), then it is nonfractal. This test is equivalent to computing the multifractal spectra and ensuring that it is not degenerate.

4.1 Technique to Detect Multifractal Properties

In order to evaluate the multifractal characteristics of the set of observed IP addresses in a given trace, we use the **Method of Moments** approach [18] adapted for the discrete IP space, which consists of the following steps:

- Compute **multiresolution quantities** by building a table of how large every subnet is for each prefix length (§ 4.1.1)

- Compute the **structure functions** using the multiresolution quantities table (§ 4.1.2)

- Compute the **partition function** by determining the slope for each of the structure functions(§ 4.1.3)

- Nonlinearity of the partition function indicates the presence of a multifractal structure to the data (§ 4.1.4)

4.1.1 Multiresolution Quantities.

For $1 \leq \lambda \leq 32$, we partition the input IP space into subnets with prefix length λ. We then count the number of observed addresses that fall in each subnet, and indicate this value with the notation $\mu(i, \lambda)$, where $i \in [0..2^{\lambda} - 1]$ indicates the index of the subnet being examined, and λ indicates the prefix length of that subnet. Note that the structure of subnets mirrors the binary tree construction outlined in [15].

4.1.2 Structure Function. For a given "moment" q, $Z(\lambda, q) = \sum_{i} \mu(i, \lambda)^{q}$ defines the structure function across all subnets with prefix length λ.We modify the computation slightly such that if $\mu(i, \lambda) = 0$, $\mu(i, \lambda)^{q} = 0$ for any q (even if $q \leq 0$, which would normally yield 1 or not be defined). Relaxing the computation in this way is a necessary to facilitate translation from a continuous space to a discrete space. An example structure function graph can be seen in Figure 1. In effect, the structure function $Z(\lambda, q)$ is an estimate of the q-th moment of μ, and multifractality presumes that these moment estimates will exhibit some scaling behavior for large λ. Further analysis of the behavior of the structure function can be found in § 5.4.1.

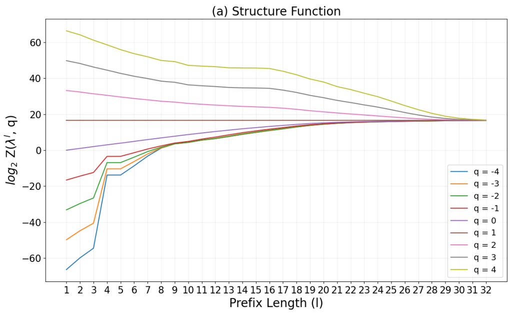


Figure 1. UO-20220517 structure function graph

4.1.3 Partition Function. For a set of outputs $\{Z(\lambda, q) | \lambda \in [s..e]\}$ for a given range of moments q and for a range of prefix lengths s to e in the structure function, let $\tau(q)$ represent the slope of the linear regression of $log_2(Z(\lambda, q))$ vs. $log_2(\lambda)$. For our evaluation, we set our prefix range to work between the 8 and 24 prefixes ($s = 8, e = 24$). We compute $\tau(q)$ for $q \in \mathbb{Z} : -4 \leq q \leq 4$. An example partition function graph can be seen in Figure 2. Formally, if the structure functions serve as estimates of the q-th moments of $Z(\lambda, q)$, then multifractality presumes that $E[\mu^q] \approx Z(\lambda, q) \approx \lambda^{\tau(q)}$ for large values of λ; that is, the partition function attempts to capture the presumed scaling behavior. Further analysis of the behavior of the partition function is provided in § 5.4.2.

4.1.4 Final Analysis.

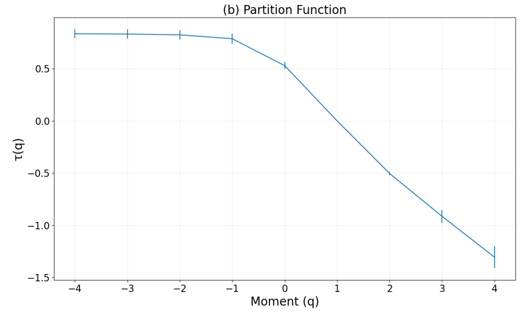


Figure 2. UO-20220517 partition function graph

Following the multifractal formalism outlined by Riedi [15], the partition function $\tau(q)$ would be input in to a Legendre transform $\tau(q) = min(\alpha q - f(\alpha))$ to derive the multifractal spectrum $f(\alpha)$. Thus, monofractal data is expected for a linear input as it results in a degenerate multifractal spectrum. On the other hand, multifractal data results from a nonlinear input as it produces an output with a nondegenerate (i.e. wide) support that indicates a varying fractal dimension, which is the defining characteristic of multifractal structures [16]. In order to observe this, we plot $\tau(q)$ against q and see if there is a substantial nonlinearity. Note that $\tau(q)$ is largely noninformative for nonfractal data, except for showing that the fractal dimension is identical to the Euclidean dimension (i.e., the data fill or cover the entire space).

CHAPTER V

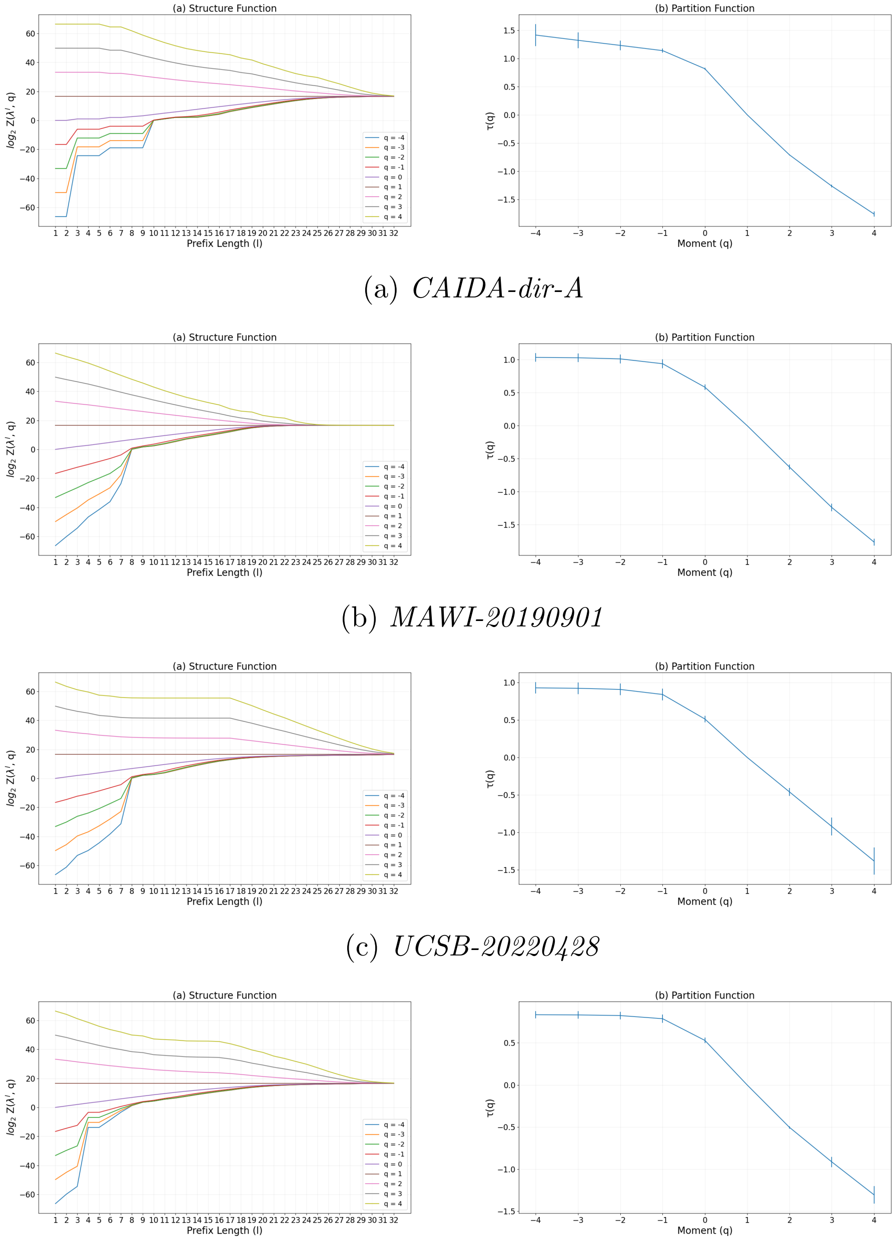
RESULTS

In this chapter, we will first apply the method of moments methodology to uniformly random data in order to get a "baseline" for what nonfractal data looks like. We then apply the same methodology to a single dataset to demonstrate what the results look like for multifractal data. With a fresh understanding of how the analysis works per dataset, we examine aggregates of the results across all datasets. Finally some time is devoted to discussing the practical considerations of the method of moments as well as intuitive interpretations of the functions and results.

A sample of results covering each data source can be seen in Figure 3. Each row corresponds to a single dataset, and one dataset from each network source was selected. The result of the structure function computation is visible in the left column, and the corresponding partition function computation can be seen in the right column. These are provided here primarily as a visual baseline to show the general behavior common between all the datasets, and to give a rough idea of the particular behavior of each data source. These graphs will be discussed in greater detail later in the document.

5.1 An Example of Nonfractal Data

In order to make concrete what was described in § 4.1.4, we will spend a moment examining data that does not posses any multifractal characteristics. The formal definition of nonfractal that we use is that a set in $[0, 1]$ is nonfractal if its fractal dimension is equal to its dimension as a subset of $[0, 1]$. For the linear IP space, the dimension is 1, so if the computed fractal dimension of a set is 1, then we claim the set is nonfractal. In more intuitive terms, the points of a



(a) *CAIDA-dir-A*

(b) *MAWI-20190901*

(c) *UCSB-20220428*

(d) *UO-20220517*

Figure 3. Structure and partition function graphs for selected datasets

nonfractal set fully fill or cover the space. Graphs of the structure function and partition function for a set of 100K IP addresses that were generated by casting 32 bit integers drawn from a uniformly distributed random source as IP addresses

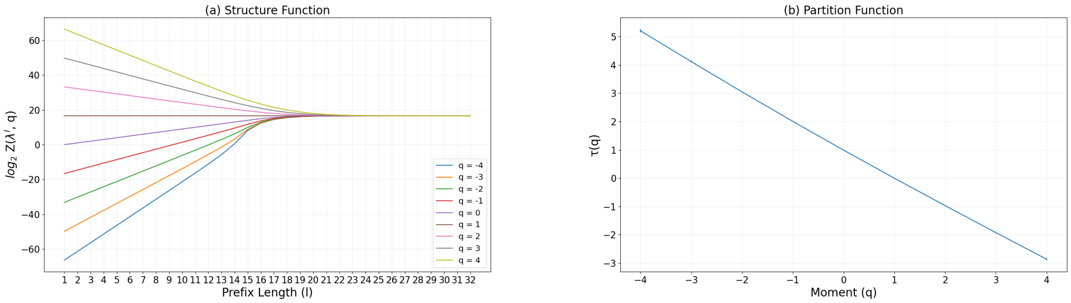


Figure 4. Graph of the structure and partition function for 100K IP addresses generated from uniformly random integers (i.e. nonmultifractal data). Note that the partition function is computed between /0 and /16 instead of /8 and /24

can be seen in Figure 4. A set of values drawn from a uniform distribution by definition are extremely unlikely to posses any multifractal structure. Though this is not monofractal data, the intent is to provide an example of what data that doesn't show the properties we are discussing looks like for the purpose of building intuition. In that capacity uniform random data suffices.

The structure function graphs start at $l = 1$, but in the $l = 0$ case, the y value for each moment would be $log_2(n^q)$ where n is the number of input IPs, which follows from the equation for the structure function, and the fact that we only have one subnet at prefix length 0. This gives a predictable starting point for each moment q in the structure function graph. Additionally, the y value at $l = 32$ will be $log_2(n)$ for every moment, as at $l = 32$, each subnet either has 0 or 1 IPs, so the computation will simply return the number of IPs. We call the horizontal line at $y = log_2(n)$ the **convergence line**, as the lines for every moment will eventually converge to it. Note that the lines for each moment in this structure function plot converge at roughly the same prefix length. As predicted, the partition function for random data is almost perfectly linear, as the data does not posses a varying fractal dimension beyond what little incidental structure emerged from the random number generator.

Figure 4 shows that the input data is uniformly random (i.e. nonfractal), and $\tau(0) \approx 1$, as was predicted for nonfractal data. The partition function here is computed between /0 and /16 rather than /8 and /24 like every other partition function in this document as there is a very distinct nonlinearity in the structure function around /16, and per § 4.1.3, the analysis is only valid over sections of the structure function where every line is linear. Furthermore, there is no "human interference" at small prefix lengths for the random data, so the motivation for excluding the /0 to /8 range does not hold. Note that shifting the range like this is not required for other traces as they are more or less linear between /8 and /24.

5.2 Multifractal Analysis of a Representative Sample

In Figure 5, we see the structure and partition functions for the first 100K IPs of the MAWI 2021-01-10 trace, and the differences from the structure and partition graphs for the random data are stark. Notice that the graph of the structure function has a "knee" between $l = 7$ and $l = 10$ where all the lines for negative moments rapidly approach 0, and the lines for the positive moments don't converge to the expected value of $log_2(100,000) \approx 16.61$ until far later, with the

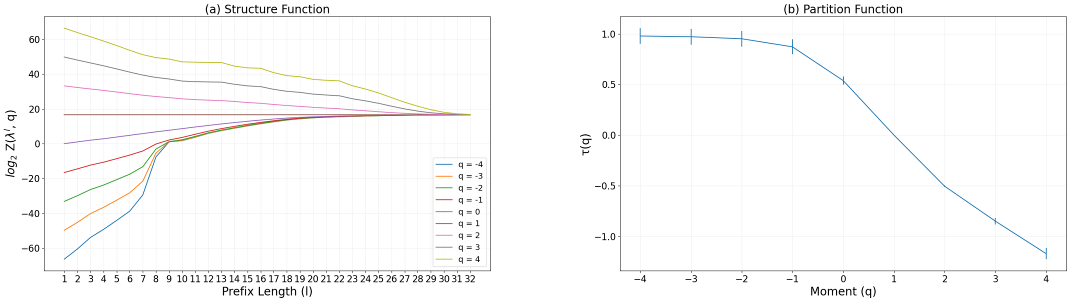


Figure 5. Graph of the structure and partition function for the first 100K IPs of the *MAWI-20210110* trace

line for $q = 4$ not converging until $l = 32$. These observations are represented more numerically in the partition function graph, where the negative moments all have slopes near 1 (recall that this only considers $l \in [8, 24]$, so the "knee" is not examined), and the positive moments trend towards having more negative slopes. The most critical characteristic is the pronounced nonlinearity between the segment for $q \in [-4..-1]$ and $q \in [0..4]$, which is strongly indicative of a varying multifractal dimension, and thus multifractal structure. The partition graph being split in this manner is a common theme in all examined data.

5.3 Aggregate Analysis

As we are using nonlinearity in the partition function to judge the presence of multifractal structure in the data, the simplest analytic approach is to ask "how bent is it" for every trace we are examining. Prior works answered this question by computing the R-Squared value for the partition function output, but we argue that this approach, while more statistically rigorous, is overly general as the nature of the nonlinearity is identical across all samples. Manual inspection of all the data demonstrated that the bend consistently occurred at or adjacent to $q = 0$, so comparing the slope of the graph for $q \in [-4.. - 1]$ with the slope of the graph for $q \in [0..4]$, offers a general approach for evaluating nonlinearity. In order to rigorously compare the pre and post bend slope, a linear regression is computed for each region and plotted. Examining Figure 6, it is clear that the slopes of the "flat part" from $q = -4$ to $q = -1$ are all very shallow, with the steepest slope being -0.08, and most of the slopes clustering around -0.03. In contrast, the shallowest slope in the "bent part" Figure 7 is -0.31. Our metric for "how bent is it" then becomes the slope of the bent part divided by the slope of the flat part. Nonmultifractal data should yield approximately 1 for this metric, as the slope for

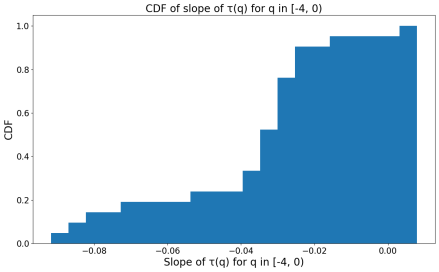


Figure 6. CDF of slopes of partition function for $q \in [-4, -1]$

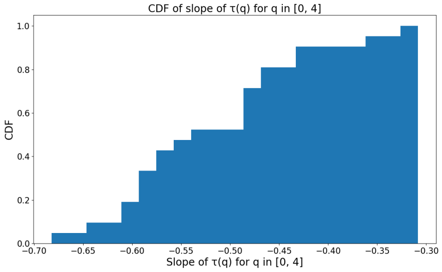


Figure 7. CDF of slopes of partition function for $q \in [0, 4]$

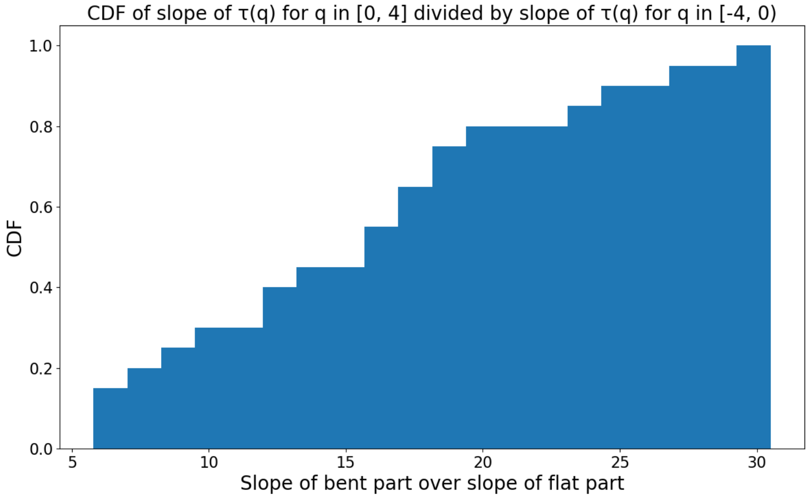


Figure 8. CDF of ratio of slope of "bent part" over slope of "flat part"

$q \in [-4.. - 1]$ and for $q \in [0..4]$ are expected to be the same. Upon examining the collected data, Figure 8 shows that none of the samples has a ratio anywhere near 1, rather the smallest ratio is 6, and the ratios seem roughly evenly distributed between 6 and 31, which indicates that all examined samples have a substantial nonlinearity, which indicates strong multifractal scaling.

5.4 Examination of Method of Moments

Now that these operations have context in the form of results, we take a moment to go back and build up intuition about why each function behaves as it does.

5.4.1 Structure Function. This is a rough translation of the "box counting" method [14] for determining multifractal spectra in continuous input spaces to a discrete input space. Intuitively, one should think of the multiresolution quantities as being similar to finding the mass distribution of the set of input IPs, and the structure function computes a value that correlates with how clustered the mass distribution of the input is. To illustrate the behavior of $Z(\lambda, q)$ consider some input set with enough IPs to fill one subnet with prefix length λ. Now consider how different distributions of those addresses throughout the IP space will impact the value of $Z(\lambda, q)$. If $q = 0$, the function will return the number of subnets that have one or more IPs in them. If $q > 0$, the function will assume its maximum value when every IP is in the same single subnet, and its minimum value when the addresses are distributed evenly across all subnets. Conversely, if $q < 0$, the maximum value occurs when the addresses are spread evenly across all subnets, and the minimum value is occurs when every address is in the same subnet. The value of q changes the size of the subnets that get measured, in a basic sense.

5.4.2 Partition Function.

Selection of the limited prefix range for the partition function is intended to cope with a critical restriction outlined in [7]: When finding the slope for a moment, the line formed by the plotted points must be straight, otherwise the slope found by least squares has no physical meaning. The moment lines for the /8 to /24 prefix range are consistently sufficiently linear that we are comfortable with accepting the result of the linear regression as matching the real slope of the segment being examined. Furthermore, we justify this choice by making the following claims: First, the distribution of subnets up to /8 (and potentially deeper) is dominated by human influence (namely the IANA), so taking measurements here is more akin to measuring the outcomes of negotiations and deals during the early days of the internet rather than more "organic" allocation patterns. Second, prefix lengths greater than 24 tend to run into finite limit effects, as there is a hard scaling limit at /32, and as we approach that limit the structure being analyzed often breaks down.

5.4.3 Interpretation of Moments.

There is a dearth of interpretations of what the moments q actually correspond to in the literature. [16] describes q as a "variable [that] selects different resolutions, with higher values of q selecting a local scaling...of lower order". While nice and general, it is not obvious how to take this description and interpret the shape of the structure and partition graphs, so we lend the intuition we have developed. Intuitively, it is probably most helpful to think about the structure function without the context of multifractals as being a function capable of being "tuned" for certain types/sizes of clusters of addresses by setting q. The structure function converges to the number of IPs in the input set once the type of cluster it is tuned for has been broken up by the increasing prefix length λ (i.e. the prefix length becomes long enough that most of the IPs appear in different subnets).

The $q = 0$ case is a raw estimate of the box-counting dimension of the set of observed IP addresses as a subset of the IPv4 space. This therefore gives information on to what extent the data (IP addresses) is filling out its physical support (the IP space). In measure-centric language, the set of input IPs can be normalized to some subset of $[0, 1]$ (as was done in [11]), and $\tau(0)$ would estimate the fractal dimension of this set. As multifractals tend to not fill out their spaces (versus something like a Hilbert Curve, which does), we would expect $\tau(0) < 1$ for multifractal data. The $\tau(0)$ values for each dataset can be seen in Table 2.

Negative moments ($q < 0$) tune the structure function to ignore subnets with many IPs in them (e.g. for $q = -2$, two subnets each with 50 addresses will contribute eight times the mass as one subnet with 100 addresses), which interestingly makes their presence very apparent since there is a huge gap between the lines for $q < 0$ and the convergence line. Among the sample data, it was common for traffic to concentrate in a couple of /7 or /8 subnets, which contained a large share of the addresses (Figure 9 and Figure 10. Both are generated using the technique outlined in § 5.4.3.1). Structure functions with negative moments would functionally erase these large subnets from the total until they had been broken up sufficiently (i.e. $\lambda = 10$ or $\lambda = 11$), at which point the set of now smaller subnets would start contributing meaningfully to the sum. Positive moments ($q > 0$) conversely tune the structure function to "amplify" the presence of clusters at large prefix lengths by weighting larger subnets much more heavily. For example, for $q = 2$, a subnet with 100 IPs will have double the mass of two 50 IP subnets. Notice that this ratio is much smaller than the $q = -2$ case described earlier, which likely contributes to the relatively rapid rate of change for the negative moments compared to the positive ones.

Further analyzing the behavior of the positive moments, imagine a uniform random distribution of n IP addresses: the expected number of IPs per subnet with prefix length λ is $\frac{n}{2^\lambda}$. If set an arbitrary limit and say the line for a moment is "pretty close" to converged once there are 2 IPs per subnet, then we would expect the values for the positive moments to converge when the expected IPs per subnet is 2. For $n = 100000$, this happens at $\lambda \approx 16$, and if we further restrict our definition of "pretty close" and say the expected value must be 0.5 IPs per subnet, that happens at $\lambda \approx 18$. Checking the structure function graph of random IPs Figure 4 shows that this prediction aligns with the observed outcome. Contraposing this, if the structure function for positive moments has not converged by the time the expected number of IPs per subnet is small (< 1), that means that there are clusters of IPs concentrated in certain subnets at long prefix lengths, and the distribution of IPs is not uniformly random. In this sense, positive moments tune the structure function to be sensitive to the presence of big clusters, and this effect is particularly visible at long prefix lengths.

To connect this behavior back to multifractals, if we imagine a multifractal as a sort of stacking of somewhat self-similar mono-fractals with different fractal dimensions, the moment q selects which fractal dimension we want to be sensitive to. Lower values of q look for larger fractal dimension, meaning it is sensitive to structures at very large scales reappearing at very small scales. Conversely, higher values of q look for lower fractal dimensions, meaning it is sensitive to structures at a given scale reappearing at nearby scales.

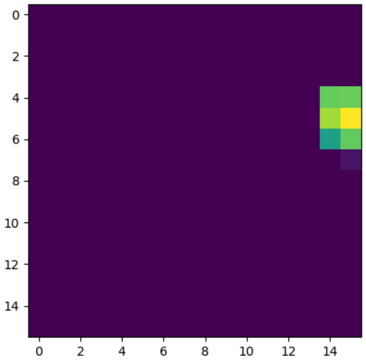


Figure 9. Heatmap showing the portion of addresses in /8 subnets for the *CAIDA-dir-A* trace

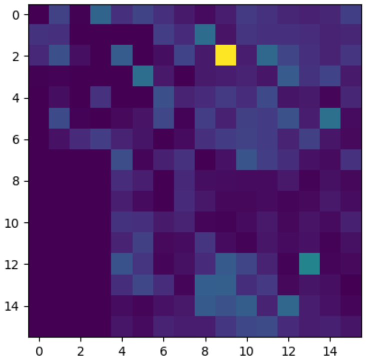


Figure 10. Heatmap showing the portion of addresses in /8 subnets for the *MAWI-20210110* trace

5.4.3.1 Heatmaps. The heatmaps seen in Figure 9 and Figure 10 are generated by dividing the input IP space into subnets with a given prefix length, counting the number of IPs in each subnet, and then mapping the subnets to a grid by taking the linear space of IPs and transforming it into a two dimensional Hilbert Curve of order 8. Colors are then assigned via one of two schemes: In the first scheme, subnets are assigned a color whose brightness linearly corresponds to the number of IPs in that subnet. In the second scheme, a subnet is colored black if there are any IPs at all in it, and white otherwise.

5.4.4 Influence of Sparsely Populated Subnets on Negative Moments. Across the structure function graphs for every real world dataset, a “knee” was observed somewhere between the /6 and /10 prefix, where the negative moments abruptly shot up towards 0. This is caused by the properties of $Z(\lambda, q)$ when $q < 0$: As soon as one of the subnets being examined only has one IP, $Z(\lambda, q) > 1$, which trivially follows the definition of Z. On the structure function log plot, this effect manifests as the line crossing the x-axis. It is well known that the distribution of network traffic by prefix is long tailed, so the distribution of active IPs by prefix likely has the same property, so it is reasonable to expect that at relatively early prefix lengths there will be a subnets with only one IP. Recall that due to the relaxation in the computation described in § 4.1.2, subnets with no IPs are not considered in the computation. Checking Table 2, one can actually match up the prefix length where each dataset has a subnet with one active IP with the prefix length where the knee is in the dataset. Weaknesses in the mapping of the box counting technique from continuous spaces to discrete spaces are likely to blame for this behavior.

Some traces, like the *CAIDA-dir-A* trace show a sort of “stairstepping” in the stucture function graph Figure 3 that isn’t a proper knee. Examining the heatmap for *CAIDA-dir-A* Figure 9 shows how the effect described in § 5.4.3 for negative moments could cause this: when the overwhelming majority of the addresses are concentrated in six /8 subnets, each time an increase in resolution divides that set of addresses it has a profound impact, but occasionally the division caused by a zoom in will have no impact if the IPs happen to coincide in the same subnet. This gives rise to the stair-stepping effect, as some increases in prefix resolution split the IPs, and others don’t.

Dataset	/6	/7	/8	$\tau(0)$
CAIDA-dir-A	0	0	0	0.816
CAIDA-dir-B	$\underline{5}$	11	30	0.499
MAWI-20150202	0	$\underline{1}$	8	0.577
MAWI-20150710	0	0	$\underline{7}$	0.580
MAWI-20151002	0	0	$\underline{3}$	0.579
MAWI-20180316	0	$\underline{2}$	7	0.583
MAWI-20180807	0	$\underline{2}$	4	0.585
MAWI-20181107	0	$\underline{1}$	1	0.585
MAWI-20190901	0	$\underline{1}$	3	0.583
MAWI-20210110	0	0	$\underline{2}$	0.538
MAWI-20210614	0	0	$\underline{5}$	0.524
MAWI-20211212	0	0	$\underline{1}$	0.514
UCSB-20220428	0	$\underline{1}$	4	0.512
UCSB-20220921	0	0	$\underline{1}$	0.503
UCSB-20221205	0	0	$\underline{3}$	0.508
UO-20181106	0	$\underline{1}$	1	0.552
UO-20190829	0	0	0	0.548
UO-20200213	0	0	0	0.544
UO-20211106	0	0	$\underline{4}$	0.541
UO-20220517	0	$\underline{2}$	11	0.530

Table 2. Number of subnets containing 1 IP address for a set of prefix lengths, underlines indicate the shortest prefix length where a subnet had a single active IP in it. $\tau(0)$ is included to illustrate the relationship between scaling factor and IP dispersion and provides an estimate of the fractal dimension of the observed set of IPs.

CHAPTER VI

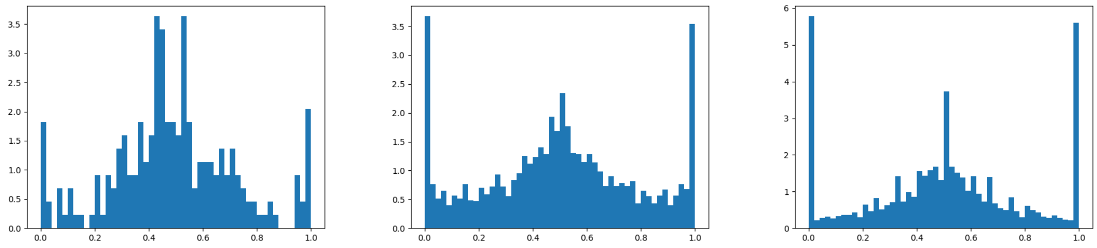
PHYSICAL EXPLANATION

In this chapter, we dig in to the lower level statistical mechanics that describe how the multifractal structures are generated locally. We then take this analysis and postulate an mechanism by which real-world phenomena could give rise to the observed statistical mechanics. Finally we translate the observed mechanisms into a generative model that creates realistic multifractal IP distributions, and spend some time analyzing its output.

6.1 Preliminary Findings

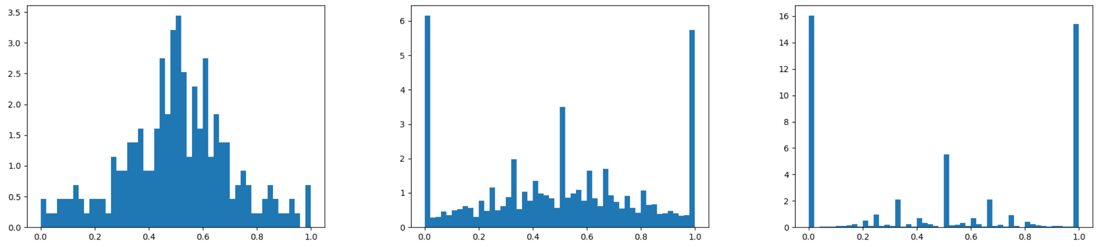
In order to peek at the statistical mechanics at work within the data, we ran each set of input IPs through the level weight procedure outlined in § 6.1.2 for prefix lengths of 8, 12, and 16; and generated a histogram, a selection of which can be found in Figure 11, Figure 13, and Figure 12. This procedure examines every subnet at the given prefix length, finds the probability that an IP in that subnet has a 1 as the next bit after the current prefix, and plots a histogram of those values. Note that these are recorded across the duration of the trace rather than the first 100K IPs, so larger traces like CAIDA will be smoothed out relative to the smaller traces. The /8 prefix is typically noisy due to a small number of possible samples (256), but shows something that very clearly resembles a normal curve or beta function, excluding the peaks at 0 and 1 which we choose to ignore for reasons covered in § 6.2.2. This observation drove the development of our generative cascade model.

6.1.1 Physical Interpretation. Following Kohler et al.[11], we concur that the multifractal structure is likely due to organizations receiving address allocations, partitioning them up, and redistributing the partitions to



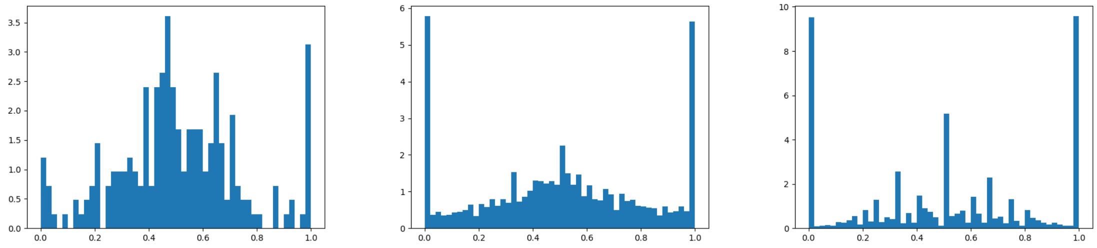
(a) /8 (b) /12 (c) /16

Figure 11. Level weights for *CAIDA-dir-A*



(a) /8 (b) /12 (c) /16

Figure 12. Level weights for *UCSB-20220921*



(a) /8 (b) /12 (c) /16

Figure 13. Level weights for *MAWI-20210110*

customers, which aligns rather closely with the cascade model. We further propose that there are "phases" to the allocation behavior as prefix length increases. The first phase from /0 to /8 − /10 is dominated by IANA allocations[9][2] The second phase from /8 to /15 or so is dominated primarily by internet service providers and secondarily by "legacy organization" that got their /8 allocations back before that was a substantial commitment and have been hanging on to them (the DOD being a prime example of this). For ISPs, a quick examination of the address blocks allocated to Comcast and Spectrum indicates that they split their allocations up into large /9s-/15s that they then allocate internally on a geographic basis, often by city. The third phase is the allocation of either single addresses or small blocks to customers, and the final phase is the customer distributing their address(es) among their local machines. Each of these phases has a tendency to create clusters: for the first phase addresses are divided up geographically to different RIRs (though this is pretty messy), which creates natural clusters along geographic borders. In the second phase, service providers do another layer of geographical partitioning, which creates clusters whose size and activity likely correlate to the population of the city they are assigned to. In the third phase, more clusters are created as geographic blocks are chopped up and handed out, and the fourth phase likely happens on such a small scale that it would be challenging to classify anything as a substantial cluster.

6.1.2 Level Weights. We use "level weights" to describe a list of probabilities generated using the following technique (as seen in Figure 11, Figure 13 and Figure 12):

- Initialize a list lw with no entries

- Divide the set of input IPs in to subnets with prefix length λ

- For each subnet, count how many of its member IPs have a 1 for bit $\lambda + 1$
- Divide the above count by the total number of IPs in the set, and append this portion to lw

These are graphed as a histogram where each bar corresponds to the portion of a subnets child IPs that have a 1 for the next bit (i.e. the probability that an IP in that subnet has a 1 as the next bit after the subnet prefix), and the height of the bar is the relative occurrence of that probability.

6.2 Generative Cascade Model

In order to synthesize lists of IP addresses showing multifractal properties, we use the following procedure:

Given a number n indicating how many IP addresses should be generated, a number l indicating the current prefix length of the working pool of addresses, and a prefix p indicating the current IP prefix, and beta distribution parameters a and b:

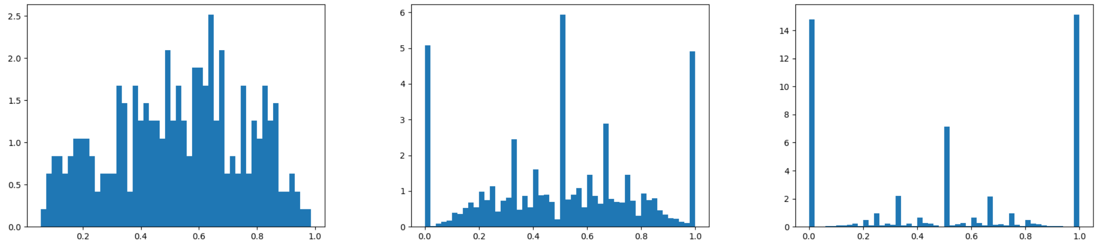
- If p has all 32 bits filled out and $n = 1$, add it to the list of output addresses and return. If $n > 1$, indicate that there was a duplicate.
- Sample the distribution $\beta(a, b)$ to get a number x between 0 and 1.
- Generate a new prefix $p0$ by setting bit l of p to 0.
- Generate a new prefix $p1$ by setting bit l of p to 1.
- Recursively call this operation with $p = p0$, $n = floor(x * n)$, $l = l + 1$
- Recursively call this operation with $p = p1$, $n = n - floor(x * n)$, $l = l + 1$

In plainer language, this procedure works by recursively partitioning a set of size n in to two parts, where one part has $\beta(a, b)$ portion of the input set, and the other part has the rest. Members of the first set get 0's the bit corresponding to the current recursion depth, and the other set gets 1's. By adjusting the a and b parameters of the Beta Function, it is possible to adjust the distribution of IPs. In the experimental data, a and b tended to be equal, however setting a and b to be smaller tended to create more clustered sets of IPs, while setting them larger created smoother and more evenly distributed sets of IPs.

6.2.1 Cascade Model Results.

The level weights computed for the cascade model 14 align quite well with those computed for the samples. The structure functions for the synthetic data Figure 15, Figure 16, Figure 17 have the key features found in the structure functions for the experimental data: the more clustered it is, the later the positive moments converge and the earlier the negative moments converge. There is one key problem: the knee is missing for Figure 15 and Figure 16 (whose a and b parameters are most similar to the observed data). As outlined in § 5.4.4, we believe that at short prefix lengths in real data, there is some not-so-organic clustering as a consequence of how the IP space is allocated, and absent this initial perturbation in the synthetic data, the knee evaporates away on the less clustered data.

The partition functions for the synthetic data seen in Figure 18, Figure 19, and Figure 20 also demonstrates the nonlinearity characteristic of multifractal structure in the data, and furthermore the nonlinearity becomes more pronounced as the beta function parameters are reduced. This aligns completely with the expectations laid out above.



(a) /8

(b) /12

(c) /16

Figure 14. Level weights for cascade model synthetic data. $a = 2, b = 2$

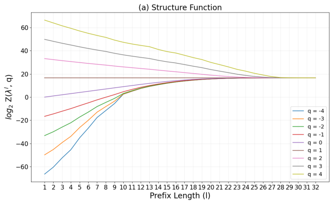


Figure 15. Structure function for 100K synthetic IPs (a=2, b=2)

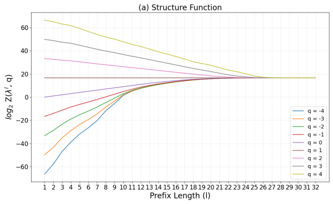


Figure 16. Structure function for 100K synthetic IPs (a=3, b=3)

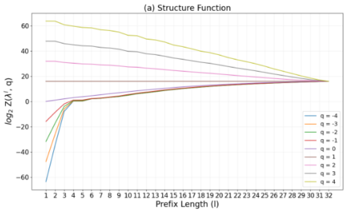


Figure 17. Structure function for 100K synthetic IPs (a=0.5, b=0.5)

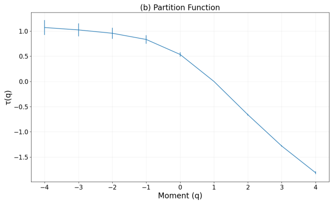


Figure 18. Partition function for 100K synthetic IPs (a=2, b=2)

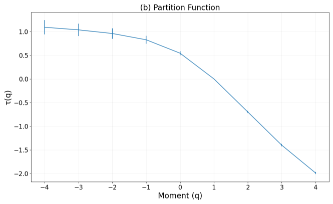


Figure 19. Partition function for 100K synthetic IPs (a=3, b=3)

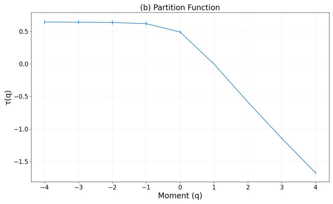


Figure 20. Partition function for 100K synthetic IPs (a=0.5, b=0.5)

6.2.2 Cascade Model Limitations.

A key deficiency of the model is that the beta function parameters do not vary with respect to bit depth, while in empirical testing seen in Figure 21, Figure 22, and Figure 23, there appears to be a trend towards higher coefficients at deeper bit depths. Implementing this as part of the model would be trivial, however the observed trend is likely heavily influenced by insufficient sample size at long prefix lengths. As the prefix length grows and the number of IPs per subnet shrinks, the samples taken of the beta distribution (as described in § 6.2.3) become quantized. When a subnet has four IPs, for example, getting the portion of IPs with a 1 in the next bit is going to yield $0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$ or 1, regardless of the underlying distribution. This effect becomes particularly pronounced among subnets with one or two addresses, and at longer prefix lengths $0, \frac{1}{2}$, and 1 occur with overwhelming frequency. This is likely to lead to increasingly incorrect regression values as prefix length increases, so experimentation with much larger datasets that can fill out subnets with enough IPs at much deeper prefix lengths is likely needed to conclusively determine the beta coefficients at those prefix lengths. Additionally, techniques to dequantize data by injecting noise in to the samples may actually aid the regression system.

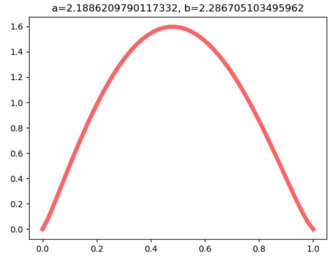


Figure 21. Regression of beta function parameters for *MAWI-20210110* at $\lambda = 8$

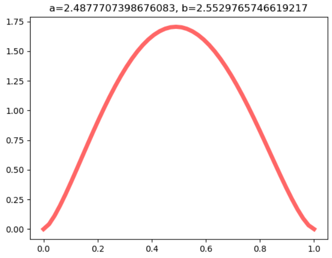


Figure 22. Regression of beta function parameters for *MAWI-20210110* at $\lambda = 16$

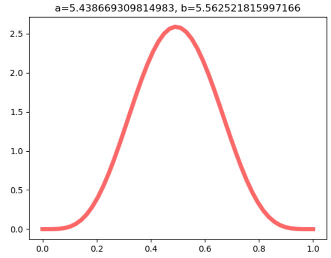


Figure 23. Regression of beta function parameters for *MAWI-20210110* at $\lambda = 24$

6.2.3 Beta Regression. To generate Figure 21, Figure 22 and Figure 23, we use the process for generating level weights to get a list of probabilities that, in each subnet with prefix length λ, the $(\lambda + 1)$th bit of a child IP from that subnet is 1. We treat these probabilities as samples from a beta distribution, and use regression techniques to compute the a and b parameters of the distribution. Due to properties of the level weights outlined in § 6.2.2, the samples have strong probabilities of assuming values of 0 and 1 (and to a lesser extent, simple fractions), but the peaks from these are placed almost optimally to interfere with the beta regression system and they represent heavily distorted samples, so during regression values of 0 and 1 are ignored.

CHAPTER VII

USE CASES FOR MULTIFRACTAL ADDRESS STRUCTURE

In this chapter we will start by looking at a method by which the clustering property of multifractal data can be taken advantage of to optimize resource usage of dataplane telemetry systems via a filtering scheme. Next we will examine some possibilities of using the multifractal spectrum as a “signature” for certain phenomenon or using it as a sort of alarm to indicate something about the nature of the network traffic changed. Finally we will look at the viability of deploying the method of moments technique to a real dataplane telemetry system.

7.1 Clustered Prefix Tracking

Current efforts are finding success with using the “clusters-of-clusters” property that comes with multifractal behavior to potentially reduce workloads on IDS and network telemetry systems that depend on knowing things about the source of the packets they examine [3]. Soldo et al. [17] describe a technique for coming up with compact filter rules, though they do not explore how clustering behavior benefits these filtering rules. Chen et al. [5] found that the overwhelming majority of malicious traffic tended to be generated by addresses from a fixed subset of the IP space, and similarly Collins et al. [6] found that malicious hosts tended to cluster in the address space. A number of papers deal with iteratively refining their queries over the IP space [12][13][10][20], which carries an implicit assumption that the features they are refining towards tend to cluster in the IP space. Our work justifies this assumption. With the knowledge that sources generating traffic of interest tend to cluster, it becomes possible to handle those clusters in aggregate rather than individually.

Consider some task where a dataplane telemetry system is running a query to detect and monitor some set of addresses that are interesting (i.e. they are generating ToI). The naive approach is to monitor all traffic and test to see if every address is interesting. This is simple to implement and works well if the telemetry system has enough memory to handle the entire space of traffic, but it quickly falls apart if the traffic exceeds the resources of the telemetry system. More sophisticated approaches may do some kind of time domain multiplexing, where they cycle through monitoring subsets of the traffic such that the subset fits on the telemetry hardware. This incurs penalties to accruacy as each address will only be monitored for some portion of the time, and sophisticated statistical devices are likely required on the backend in order to make the partial data usable. An even more sophisticated model may do time domain multiplexing, but also individually track addresses that were found to be interesting, so it doesn't suffer from the loss of observation time. Unfortunately, this requires a filter per address to capture interesting traffic, and these are typically implemented as extremely expensive TCAM in dataplane hardware, so this approach fails to scale beyond the availability of TCAM rows.

Multifractal properties to the rescue! Since we know that ToI generating (i.e. interesting) addresses tend to cluster, what if it was possible to just identify the prefixes with only interesting addresses and track those? Luckily a procedure capable of doing this is quite straightforwards assuming the input has a query that is able to flag addresses as "interesting" or "uninsteresting" at runtime:

– Given a subnet with prefix length l, divide the prefix in to 2^i smaller subnets with prefix length $l + i$

- Initialize an empty list e that will contain the subnets with only interesting addresses
- See if each smaller subnet s contains interesting addresses
- If $l + i$ is 32, see if the address in s is interersting, and if it is add it to e and continue to the next smaller subnet
- If s consists of only interesting addresses, add s to e
- If s consists of a mix of interesting and uninteresting addresses, recursively run these instructions on s, and concatenate each resulting list to e
- Otherwise, if s contains only uninteresting addresses ignore it
- Once every smaller subnet is processed, return e

If $i = 1$, this will compute the shortest possible list of prefixes that captures all interesting addresses and nothing else. These prefixes would be the optimal set of things to monitor in order to only see attack traffic while using the minimum number of prefix filters. If $i > 1$, this will compute the shortest possible list of prefixes where prefix lengths are multiples of i that capture all interesting addresses and nothing else. In this sense i works as a knob that trades an increased number of prefix filters for fewer recursive iterations. Additionally, if monitoring some uninteresting traffic is permissible, limits can be placed on the recursion depth which will reduce the maximum number of filters needed. In a real deployment, this implementation would likely take the form of dataplane telemetry hardware that is running two tasks concurrently. The first task is to run the query in question on every prefix that has been flagged as containing only interesting addresses continuously. The second task is to periodically run the recursive process, starting with 0.0.0.0/0 (multiplexing subnets over time if memory is constrained), and adding the resulting list of prefixes to the set of prefixes with only interesting addresses. In the event that a prefix containing only interesting sources suddenly gains an uninteresting source, that prefix can be “evicted”, and have the recursive refinement process run on it in order to reestablish a list of finer grained prefixes that only contain interesting traffic.

This approach makes a very firm guarantee: if the system has enough memory to track all the sources generating ToI for a given query, plus some constrained overhead for the recursive scanning process, it will be capable of satisfying the query regardless of how much uninteresting traffic there is as long as it doesn't run out of room for filters. Additionally, given the clustering properties of multifractal ToI, the number of prefixes that need to be tracked (and thus filters needed) will be minimal compared to the number of addresses being tracked, so for many types of ToI, this approach is completely viable on hardware systems available today, and there is a tunable parameter that can be used to trade off the number of filters needed and the amount of uninteresting traffic that ends up being mistakenly tracked, which makes this approach viable for a wide range of capabilities.

7.2 Spectral Signature

Kohler et al.[11] describe measuring the multifractal spectra of a laboratory network before and during a pair of worm attacks in the summer of 2001, and demonstrate that spectra varied significantly both when the worm was introduced, and when a revised version of the worm changed the spreading behavior from targeting random IPs to attempting to spread by stochastically selecting IPs from similar prefixes. This indicates that there may be some value in actually tracking the multifractal spectra of addresses on a network in real time: it provides a decent set of aggregates that appear mostly time-invariant[18] on any scale between seconds and days, but it is sensitive to changes in the structure of the traffic it is built from. A rapid shift in the spectra or deviation from historical norms may actually serve as a decent alarm bell indicating that something fundamental about who is using the network changed, for example maybe there is a DDoS attack or a worm, or maybe a misconfigured router is dumping packets on the network that don't belong there. It is doubtful that the multifractal spectra is useful in its own right, but it correlates with something that every network operator cares deeply about: who is on the network? Further work in this direction might yield "spectral signatures" of different types of events, or even something as specific as a certain botnet. This idea is executed on by Barlow et al.[4], where they compute multifractal spectra of the internet background radiation, which could be construed as the signature of the background radiation, but much work remains in characterizing signatures and seeing if they are even distinct enough to draw meaning from, and further taxonomizing the signatures for different types of traffic (HTTP, UDP, etc.) to see if there is significance there. While our work does not actually compute the spectra, the partition function can be converted in to a multifractal spectra using the Legendre transform.

7.3 Implementation and Viability of Deployment to Dataplane

We implement the method of moments technique as Rust program that computes the multiresolution quantities and structure function which emits the structure function values for a python program to run a linear regression on. The entire computation takes under a second for 10M IPs on a relatively old Intel Xeon E5-2650v4 while using a single thread, and the complexity is linear with respect to the number of input IPs. The implementation can be trivially multithreaded using a thread safe hashmap as the backing structure. IP addresses are extracted from netflow and PCAP data using a pair of custom parsers written in Rust.

The dataplane side of the computation just requires collecting a list of all source or destination IP addresses (depending on the analysis), which is one of the simplest possible operations to run on modern dataplanes. Examination of Table 1 shows that even relatively large and long datasets do not accumulate more than 10M IP addresses, and in the common windowed telemetry system, the number of IPs is very manageable.

As computation of the structure and partition functions in software is relatively quick, averaging 10ms of computation time on the mentioned CPU for 100K IPs, the door to actually running the structure and partition function as part of a realtime dataplane telemetry system is wide open. Additionally, the multiresolution quantities and structure function steps are extremely amenable to being run on a stream processor as generating the base multiresolution quantity from a list of addresses is a single `reduce`, and each subsequent multiresolution quantity is a `map` and `reduce` on a dataset half as large. Computing $Z(\lambda', q)$ is a single `map` and `sum` per moment. The linear regression is a simple operation on a fixed set of values numbering less than 1000. We believe that these requirements are extremely modest for modern dataplane telemetry deployments, and the benefits outlined in § 7.1 absolutely justify the processing cost.

CHAPTER VIII

DISCUSSION AND OUTLOOK

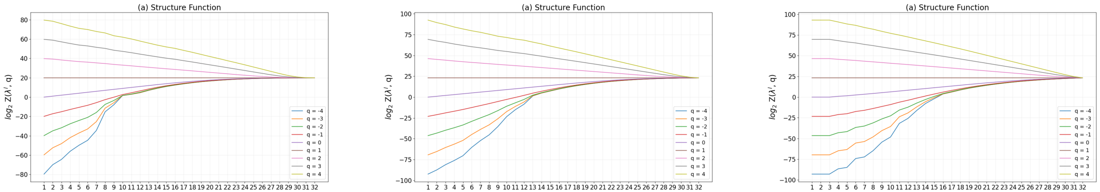
In this chapter we will first discuss some of the limitations of the method of moments, particularly as the input dataset grows to be large. Then we will outline an initial approach for generalizing our cascade model to work in both the address space and time dimensions, as well as discuss future work in characterizing the physical phenomenon that creates the observed multifractal structure.

8.1 Limitations

As the method of moments operates over a discrete space, there is a hard limit to how small the structures that can be examined are (i.e. the minimum structure size is 1 IP). Consequently, as the number of input IPs grows very large, there is a sort of "saturation", where the input space fills up with enough addresses that it is no longer possible to tell where one cluster ends and another begins. We control for saturation by only considering the first 100K IPs in each dataset, which was selected because (i) it was a round number, (ii) it was the largest round number every dataset would be able to provide for, and (iii) it was found that the structure and partition graphs at 100K IPs for each dataset were qualitatively very similar to those for every IP in the same dataset. This is a nonissue in continuous domains as it is possible to increase resolution endlessly, but in the discrete space we operate in, the size of the input actually has an impact on the result, even if the underlying structure is identical. We used the cascade model to generate 1M, and 10M, and 42M IPs (selecting 42M as it is $\approx$ 1% of the IPv4 space), all with beta parameters $a = 2$ and $b = 2$. The structure functions can be seen in Figure 24, and the partition functions in Figure 25. The computed structure and partition functions for the 1M sample are similar to the 100K sample seen in Figure 15, but those are very different from the 10M and 42M sample both of which show clear perturbances in their structure function and decay of multifractal properties in the progressively more linear partition function. We postulate that the number of inputs that causes "saturation" is highly dependent on the fractal dimension of the data, but leave exploring these properties as future work.

8.2 Discussion

The work laid out here is limited in that we only set out to examine multifractal properties in the spatial dimension, but there is a large existing body of work establishing that internet traffic shows self similarity in the temporal axis as well. Future efforts will likely need to unify these understandings, particularly if they intend to build a full-fledged synthetic dataset generator, rather than something that just generates sets of IP addresses. One simple approach for this could likely consist of generating a set of addresses using our earlier described approach, allocating packet counts to each source in an independent and identically distributed (iid) manner (using some heavy-tailed distribution), and then using some scheme (e.g., on/off behavior) to distribute each addresses' allocation of packets over time. A more elaborate but also more realistic approach would be to generate a genuine spatial-temporal traffic dynamics by avoiding the above iid assumption and replacing it by a packet allocation process that is derived from

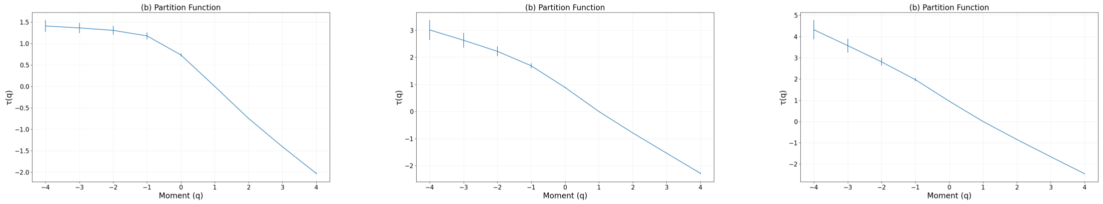


(a) 1M IPs

(b) 10M IPs

(c) 42M IPs

Figure 24. Structure functions for cascade model data with $a = 2$ and $b = 2$ for different numbers of IPs



(a) 1M IPs (b) 10M IPs (c) 42M IPs

Figure 25. Partition functions for cascade model data with $a = 2$ and $b = 2$ for different numbers of IPs

the same kind of multifractal analysis that is described in this thesis but where the measure μ counts the number of encountered packets per observed IP address rather than the number of observed IP addresses. Similar to the first approach, this approach also requires a scheme to distribute each IP's allocation of packets over time, but the allocated packet counts in this case are now tightly correlated with the multifractal scaling behavior that we would expect to observe for the measure that counts the number of observed packets per observed IP and that could also be synthetically generated by means of an appropriately defined cascade construction.

Characterizing the physical explanation for the driving force behind the cluster behavior is an open problem. We pitched one possible interpretation here § 6.1.1, but all the hard labor of digging up statistics on who owns what allocations, where they have been allocated, and how many hosts are using these allocations is left as future work. In general, we believe that the multifractal construction of the internet is caused by features and practices inherent to subnets themselves, as entities are provided some allocation of address space, which they then allocate portions of to sub-entities or clients. This is handled recursively until all addresses are distributed. As all entities that assigned addresses are not equally "active", this gives rise to a clustering behavior. The geographic interpretation discussed in § 6.1.1 is just one dimension that the clustering can manifest in, but many others are possible and likely (e.g. a cloud provider may have a single subnet assigned to“popular” servers, so these servers would be clustered in this subnet).

Furthermore, there is a nuanced discussion to be had about how the multifractal effects are observed in samples of internet traffic, but since there isn’t a big trace of all the traffic on the internet to look at, what we characterized here may not actually be a property of the internet itself but rather a behavior that rises out of how clients of the network environments that we analyzed contact the rest of the world. This opens a dizzying number of questions about how different usage patterns and user bases shape the fractal structure, and we leave answering these as future work.

CHAPTER IX

CONCLUSION

In this thesis we collected a repository of traffic of interest from a range of real-world environments and measured the multifractal properties of each dataset using the method of moments technique. We ran a bulk evaluation that found that every dataset was consistent with exhibiting multifractal scaling behavior that manifests in a pronounced cluster-within-cluster property. We then offered a possible explanation for how this clustering property may emerge, and demonstrated a parametric cascade model IP address generator that was able to generate sets of addresses that matched the multifractal scaling properties of the datasets. Finally we proposed several concrete use cases for multifractal properties and discuss the viability of computing it in a live network setting.

REFERENCES CITED

[1] The CAIDA UCSD anonymized Internet traces dataset - 2019. `https://www.caida.org/data/monitors/passive-equinix-nyc.xml`, 2019.

[2] IANA IPv4 address space registry, February 2022.

[3] Albert Gran Alcoz, Martin Strohmeier, Vincent Lenders, and Laurent Vanbever. Aggregate-based congestion control for pulse-wave ddos defense. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 693–706, 2022.

[4] Paul Barford, Rob Nowak, Rebecca Willett, and Vinod Yegneswaran. Toward a model for source address of internet background radiation. In *In Proceedings of Passive and Active Measurement Conference (PAM 2006*, 2006.

[5] Z. Chen, C. Ji, and P. Barford. Spatial-temporal characteristics of internet malicious sources. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 2306–2314, 2008.

[6] Michael Patrick Collins, Timothy J. Shimeall, Sid Faber, Jeff Janies, Rhiannon Weaver, Markus De Shon, and Joseph (Jay) B. Kadane. Using uncleanliness to predict future botnet addresses. In *ACM/SIGCOMM Internet Measurement Conference*, 2007.

[7] Carl JG Evertsz and Benoit B Mandelbrot. Multifractal measures. *Chaos and fractals*, 1992:921–953, 1992.

[8] Romain Fontugne, Pierre Borgnat, Patrice Abry, and Kensuke Fukuda. MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking. In *Proceedings of the ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2010.

[9] Elise P. Gerich. Guidelines for Management of IP Address Space. RFC 1466, May 1993.

[10] Arpit Gupta, Rob Harrison, Marco Canini, Nick Feamster, Jennifer Rexford, and Walter Willinger. Sonata: Query-driven streaming network telemetry. In *Proceedings of the conference of the ACM Special Interest Group on Data Communication (SIGCOMM)*, pages 357–371, 2018.

[11] Eddie Kohler, Jinyang Li, Vern Paxson, and Scott Shenker. Observed structure of addresses in ip traffic. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment*, IMW '02, page 253–266, New York, NY, USA, 2002. Association for Computing Machinery.

[12] Masoud Moshref, Minlan Yu, Ramesh Govindan, and Amin Vahdat. DREAM: Dynamic resource allocation for software-defined measurement. *ACM SIGCOMM Computer Communication Review*, 44(4):419–430, 2014.

[13] Masoud Moshref, Minlan Yu, Ramesh Govindan, and Amin Vahdat. SCREAM: Sketch resource allocation for software-defined measurement. In *Proceedings of the ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, page 14, 2015.

[14] Heinz-Otto Peitgen, Hartmut Jürgens, and Dietmar Saupe. Chaos and fractals - new frontiers of science. 1992.

[15] Rudolf Riedi. An introduction to multifractals. 01 1999.

[16] Hadrien Salat, Roberto Murcio, and Elsa Arcaute. Multifractal methodology. *Physica A: Statistical Mechanics and its Applications*, 473:467–487, 2017.

[17] Fabio Soldo, Karim Eldefrawy, Athina Markopoulou, Balachander Krishnamurthy, and Jacobus Merwe. Filtering sources of unwanted traffic. pages 199 – 208, 01 2008.

[18] Megan Walter. On the multifractal structure of observed internet addresses, 6 2022. Available at `https://www.cs.uoregon.edu/Reports/UG-202206-Walter.pdf`.

[19] Jun Xu, Jinliang Fan, Mostafa Ammar, and Sue B. Moon. On the design and performance of prefix-preserving ip traffic trace anonymization. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, IMW '01, page 263–266, New York, NY, USA, 2001. Association for Computing Machinery.

[20] Lihua Yuan, Chen-Nee Chuah, and Prasant Mohapatra. ProgME: towards programmable network measurement. *IEEE/ACM Transactions on Networking*, 19(1):115–128, 2011.