Internet Device Graphs

Matthew Malloy
comScore
mmalloy@comscore.com

Paul Barford
comScore, University of Wisconsin
pbarford@comscore.com

Enis Ceyhun Alp
comScore, University of Wisconsin
ecalp@cs.wisc.edu

Jonathan Koller
comScore
jkoller@comscore.com

Adria Jewell
comScore
ajewell@comscore.com

Abstract

Internet device graphs identify relationships between user-centric internet connected devices such as desktops, laptops, smartphones, tablets, gaming consoles, TV's, etc. The ability to create such graphs is compelling for online advertising, content customization, recommendation systems, security, and operations. We begin by describing an algorithm for generating a device graph based on IP-colocation, and then apply the algorithm to a corpus of over 2.5 trillion internet events collected over the period of six weeks in the United States. The resulting graph exhibits immense scale with greater than 7.3 billion edges (pair-wise relationships) between more than 1.2 billion nodes (devices), accounting for the vast majority of internet connected devices in the US. Next, we apply community detection algorithms to the graph resulting in a partitioning of internet devices into 100 million small communities representing physical households. We validate this partition with a unique ground truth dataset. We report on the characteristics of the graph and the communities. Lastly, we discuss the important issues of ethics and privacy that must be considered when creating and studying device graphs, and suggest further opportunities for device graph enrichment and application.

1 INTRODUCTION

Identifying internet users at scale in a persistent fashion (*i.e.,* without requiring an explicit login ID) is important in a variety of contexts including online advertising, content customization, recommendation systems, security, and operations. The standard mechanisms for device identification are web cookies and advertising identifiers. Cookies are unique strings associated with web browsers that are assigned by publishers or 3rd parties when a user visits a web page. Advertising identifiers are unique strings placed on mobile devices by the major operating system vendors (*e.g.,* Apple's Identifier for Advertisers (IDFA), Google's Advertising ID, and Microsoft's Advertising ID). These identifiers are available to advertisers and third parties whose apps are running on the mobile device, and have a similar purpose in terms of enabling persistent user identification.

New challenges in user identification have emerged in the past several years due to the explosion in the number of internet connected devices used by individuals on a daily basis. For example, a single person might use a desktop and a laptop, one or more smartphones or tablets, a gaming console, a smart watch, and a smart TV multiple times throughout the course of a day. Thus, a critical objective is the *capability to identify users across devices*. Addressing this objective requires *i)* massive data gathering capability and *ii)* a method for linking device identifiers.

In this paper we describe *internet device graphs*. A device graph is composed of nodes that represent internet connected devices and edges that represent some relationship between them. Relationships can be identified in a number of ways such as shared IP addresses, operating simultaneously in close physical proximity, entity-based (*i.e.,* owned by the same family or enterprise), or community-based (*i.e.,* part of the same social or interest group). The goal of device graphing is to identify these relationships accurately and at scale, in a way that offers utility in practice (*e.g.,* enabling users to be identified across devices), and conforms to established privacy guidelines and ethical principles. While there are a number of commercial device graph offerings (*e.g.,* [3, 5, 8]), we are aware of no research publications that describe methods for establishing relationships between user-centric devices at scale.

We develop an approach for generating a device graph based on *IP-colocation*. Edges in the graph are formed when pairs of devices share an IP address at the same time, commonplace when connecting to the internet via a home router. Our device graphing method assumes the availability of measurement data with *i)* a device ID (*e.g.,* a web cookie or advertising ID), *ii)* an IP address, and *iii)* a timestamp. Interpreting the IP colocation graph comes from the observation that IP space is often *intimate* – IP addresses are shared by devices that belong to the same person, family members, friends and co-workers *e.g.,* when connecting to the internet via the same access point. Time scales of weeks to months capture habitual behavior and reinforce true *user-level* and *household-level* relationships. Edge weights are designed so that a weak edge corresponds to co-workers, acquaintance, or simply coincidental connections. Moderate weight edges are shared by devices that belong to members of the same household. The strongest edges are shared by devices that belong to the same user (*e.g.,* a mobile phone and a PC belonging to the same person) or multiple IDs for the same device (*e.g.,* a cookie and an advertising ID on the same mobile device).

The IP colocation graph enables clustering of devices into communities of densely connected nodes using *community detection*

KDD'17, August 13–17, 2017, Halifax, NS, Canada.

978-1-4503-4887-4/17/08…$15.00
DOI: http://dx.doi.org/10.1145/3097983.3098114

algorithms. The communities generated from the IP colocation device graph also have real-world interpretations. Tuning the community detection algorithms and culling low weight edges from the graph can produce small groups of devices that belong to the same person, larger groups that correspond to devices in the same physical household, or even larger groups corresponding to the same workplace or broad interest community.

We demonstrate the efficacy of our methods by applying them to a dataset comprised of over 2.5 trillion web measurements gathered over the course of a six weeks in the US from comScore's digital network. The resulting graph is gigantic, including more than than 7.3 billion edges (pair-wise relationships) and 1.2 billion nodes (unique device IDs). We apply the Louvian community detection algorithm [16] with a variety of objective functions, taking additional steps to make the algorithm tractable at scale while improving accuracy. Applying community detection algorithms to the graph with appropriate parameters results in a partitioning of devices into more than 100 million communities corresponding to physical households in the United States. We report on the characteristics of the graph and communities. Lastly, we validate our finding against a unique dataset that provides approximate ground truth on the composition of internet devices in over seven thousand US households. When tuned to produce appropriate sized communities, we show that the graph and community detection algorithms are able to match more than 90% of the ground truth households with an average precision and recall above 85%, which is an overwhelming success given the scope and scale of the graph.

In summary, this paper makes the following contributions. First, we introduce device graphing and IP-colocation based methods to generate a device graph (Section 2). Next, we apply community detection algorithms to the graph to generate groupings of devices. The approach is demonstrated through implementation with a large internet measurement dataset and results are validated through a unique ground truth dataset (Section 3). Lastly, we discuss privacy and ethics (Section 4), and related work (Section 5).

2 METHODOLOGY

In this section we describe our methodology for generation of an IP-colocation device graph and identification of communities from the graph. Our method is a two step process: *i)* generate a graph G that establishes *longitudinal* relationships between devices by observing them on the same IP address in a *short* time frame, and *ii)* use G to create appropriate communities, depending on the application, by employing community detection algorithms.

There are two primary objectives in device graph generation: accuracy (*i.e.,* a relationship in the graph or a community group reflects a true relationship or community) and scalability (the utility of the graph and community groups is proportional to their scale). Achieving accuracy is complicated by IP address and device ID assignment dynamics, and by user mobility. Achieving scalability depends on developing computationally efficient algorithms and implementations.

Before detailing our methodology for device graph generation, we proceed with common terms in graph analysis. A weighted graph $G = (V, E)$ is a set of nodes V and a set of weighted edges E. A weighted edge $e \in E$ is a two element subset of V and an associated weight: $e = (i, j, w) \in V \times V \times \mathbb{R}$. In a *device* graph, a node $i \in V$ is a device ID (*e.g.,* a web cookie or advertising ID). One of our primary efforts is determining groupings of devices that have strong relationships in the graph. We refer to these groupings as *communities.* When a community is aligned to a physical household, we refer to it as such. A *community* is a set of one or more nodes, $H = \{i, \dots\} \subset V$. The set of communities is denoted $\mathcal{H} = \{H_1, H_2 \dots\}$. $\mathcal{H}$ is a partitioning of V, i.e, $H_i \cap H_j = \{\}$ for any $i \neq j$, and $\bigcup_i H_i = V$.

2.1 IP-Colocation Device Graph

Starting with a dataset consisting of tuples of *(device ID, IP address* [1]*, time)*, generation of the IP-colocation device graph proceeds as follows. First, time is discretized into epochs. On epoch $t = 1$, for each IP address in the dataset, an edge is established between every pair of device IDs that share that IP address. The weight of the edge is equal to the inverse of the total number of distinct device IDs on that IP address during that epoch. IP addresses with more than $N_{\max}$ distinct devices are not considered. The process continues for all epochs in the dataset, and a final weight is assigned to each pair of devices by summing over all epochs and all IP addresses. Only edges with weights greater than γ are included in the final graph. The algorithm is detailed in Algorithm 1.

Algorithm 1 IP-Colocation Device Graph

```
parameters: N_max, γ
input: tuples (device ID, IP address, time)
V = set of unique device IDs
for each time step t, each IP k
    N_{t,k} = number of distinct device IDs on IP k at time t
    if N_{t,k} ≤ N_max
        for all pairs of device IDs (i, j) on IP k at t
            w_{i,j}(t, k) = 1 / N_{t,k}
w_{i,j} = Σ_{t,k} w_{i,j}(t, k) for all (i, j)
E = {(i, j, w_{i,j}) : w_{i,j} > γ}
return G_γ = (V, E)
```

Algorithm 1 requires two parameters. The first parameter, $N_{\max}$, is set to exclude high volume IP addresses, which limits the total number of edges established per IP address. Setting this value too high creates a large edge set, and can make implementation intractable. Moreover, when $N_{\max}$ is sufficiently low, mobile and corporate IP addresses are largely excluded, often a desired property. The second input is the *edge cutoff* parameter γ, which results in discarding of edges below a specified weight.

The algorithm specifies an inverse relationship between the edge weight and the number of devices on an IP address. Roughly speaking, the weight of an edge can be interpreted as the number of epochs two devices share alone on an IP address. For example, if two devices are alone together on an IP address for a single epoch, they receive a weight of 1/2. As weights are additive across epochs, two devices that are alone together for two epochs form an edge with a weight of 1. Likewise, *ten* devices that share an IP for one epoch establish 10 choose 2 edges, each with a weight of 1/10. We

[1] In this paper, we consider IPv4 addresses. IPv6 will be considered in future work.

also note that the graph includes self-edges: a device entirely alone on an IP address builds a self-edge with a weight of 1. Many functions of the distinct number of device IDs on an IP addresses, or other attributes, could be used to define the weight of an edge.

2.2 Community Detection

While an IP-colocation device graph alone is a rich dataset that can provide great utility in a number of applications, one of our main objectives is to group devices into small clusters of strongly related nodes, corresponding to users, households, and larger internet communities. With this goal in mind, we applied community detection algorithms to G_γ, using the parameter γ to control the size of the communities, resulting in various partitionings of V.

Community detection is the process of partitioning a graph into clusters of nodes such that the nodes in the same group are densely connected, while the nodes from different groups are sparsely connected. The problem of community detection has drawn considerable attention from different disciplines, producing many methods. Due to the immense scale of the graph, we focused on the well studied Louvain method [16], using four distinct cost functions on G_γ, for various values of γ. The Louvain approach is a greedy method that proceeds by repeating two steps - first, the algorithm attempts to optimize the cost function by moving nodes to the communities of their neighbors. The second step involves creating a new graph by treating communities as nodes, and repeating, which naturally generates a hierarchy of communities. Each level in the hierarchy, denoted $\ell = 1, 2, \ldots$, corresponds to a different partitioning of the node set. Both decreasing γ and increasing ℓ correspond to increasing the size of the communities.

The communities we generate have real-world interpretations beyond simply sharing an IP address. When γ is large and $\ell = 1$, the communities are smallest, and correspond to groups of devices that belong to the same person – which we refer to as the *user-level*. Decreasing γ to produce larger communities results in groups of devices that belong to the same physical households or small workplace – the *household-level*. Further decreasing γ and increasing ℓ generates *internet communities* that connect extended families, friends and co-workers.

3 EVALUATION

In this section we describe results of the application of Algorithm 1 and the community detection algorithms to a large, unique dataset. Our objectives are to demonstrate implementations and to provide perspective on the scale and characteristics of the resulting graph and communities.

3.1 Data and Implementation

comScore's digital network is one of the largest in the world, collecting information on over 60 billion internet events daily[2]. Data from this network is collected through digital tagging of websites, videos, mobile apps, advertisements, web widgets and distributed content. Tagging is implemented through two means: JavaScript/HTML tags and SDK tag implementations. In both implementations, a client machine executes code locally and reports information directly to a data warehouse in the form of a record.

While the information reported in a record varies depending on the client and the tag implementation, three pieces of information are critical to the device graph *i)* a device ID, *ii)* an IPv4 address, and *iii)* a timestamp. Although the IP address and a timestamp are invariantly included with each record, a device ID may or may not be included depending on the security settings of the client. Device IDs are either: *i)* a 3rd party web browser cookie or *ii)* an advertising ID – including the Apple Identifier for Advertising (IDFA), Google Advertising ID, and Microsoft Advertising ID. A limited number of records were excluded for the following reasons: *i)* the record did not include a 3rd party cookie or an advertising ID, *ii)* the 3rd party cookie was identified as being created less than 24 hours prior to generation of the record, *iii)* the identifier was a known invalid identifier (*i.e.,* device identifiers consisting of a string of zeros, or cookies failing a checksum) and *iv)* the identifier was hyperactive – specifically, devices present on more than a thousand IP addresses.

Our study uses data collected in a 42 day (6 week) period from December 5, 2016 through January 15, 2017 and is restricted to the US. The epoch used was a UTC day; as such, the UNIX timestamp of each record is rounded to the nearest UTC day. $N_{\max}$, the maximum number of unique device IDs per epoch for which an IP address is considered, was set to 20. Results are reported for $\gamma = 0, 0.2, 0.4, 0.6, 0.8$ and 1.0. Algorithm 1 was run in an Apache Hadoop environment with 500+ worker nodes implemented in 1000 lines of Apache Pig [1]. After collection, aggregation for a single day's data was on the order of an hour. Processing time for generating the graph from the aggregate data, in this deployment, was on the order of a three hours.

Community detection was implemented using open source C++ code [15] on a single machine comprised of Intel(R) Xeon(R) CPU E5-2699 v4 chipsets, totaling 88 cores, and 768GB of memory. Timing results were collected for each γ and cost function, and ranged from less than 2 hours to more than 16 hour depending on the parameters.

3.2 Characteristics of the Graph

The resulting graph, G_0, consists of 1.3B nodes and 7.3B edges (excluding self edges), with an average degree of 11.4 edges per node. Of the 1.3B nodes, 344M corresponded to advertising IDs, and 935M to web cookies. Tables 1 and 2 show statistics of the graph, including breakouts by node type (web cookie or advertising ID).

While G_0 is not fully connected, there is a large connected component to which 85% (1.08B) of nodes belong. Table 3 shows the size of the largest connected component and the approximate average shortest path corresponding to that component, calculated using open source code [38]. The average shortest path is defined as the minimum number of edges connecting two nodes, and was computed by sampling 500 pairs of nodes.

Figure 2 and Figure 3 show the edge weight distribution and the node degree distribution of G_0. The maximum weight of any edge was a cookie to cookie edge with a weight of 274, while the maximum degree of any node was 3090, corresponding to a cookie.

Figure 4 shows a plot of the size of the connected components against the ratio of nodes that belong to such sized or smaller

[2]In an independent web crawl in November of 2013, 18.19% of the Alexa top 100,000 domains placed 3rd party cookies of the analytics company [17]. This is the second largest percent amongst 3rd party hosts.

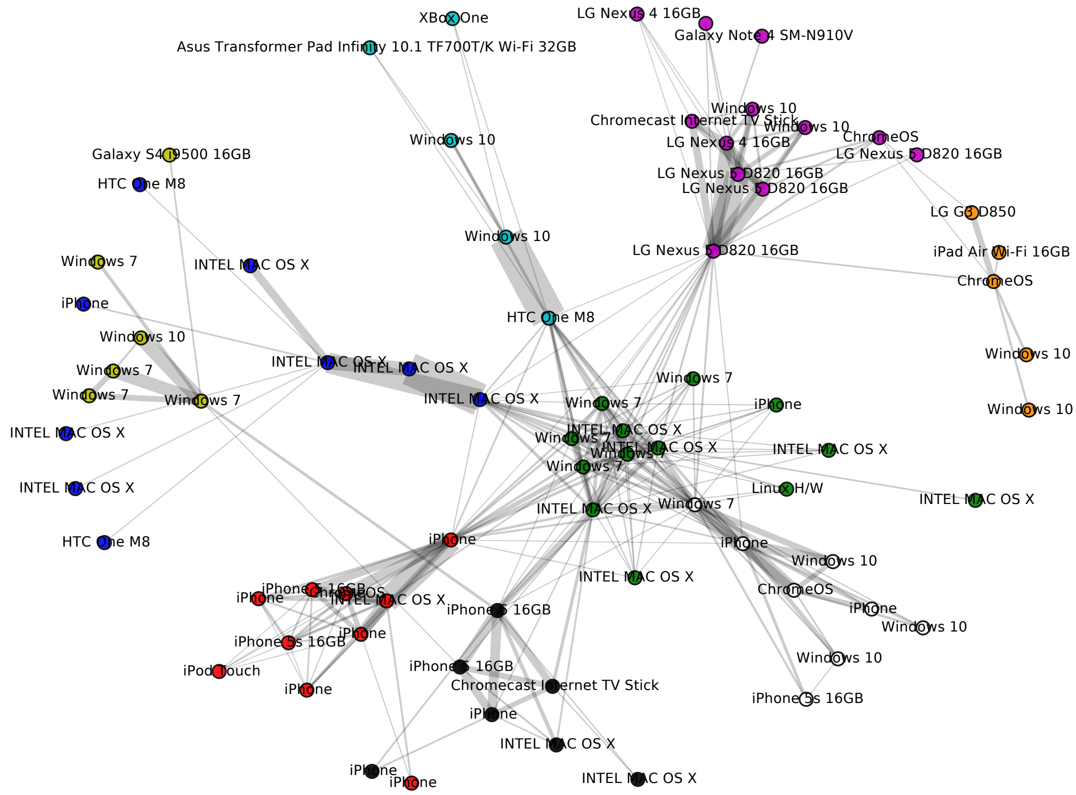

Figure 1: Visualization of a small portion (a 4-hop subgraph) of the IP co-location graph $G_{0.4}$ created by Algorithm 1. Each node represents a device ID (either a cookie or an advertising ID). The thickness of the edge between two nodes is proportional to $w_{i,j}$, defined in Algorithm 1. Node colors indicate communities, corresponding to physical households, as determined by the Louvain Modularity community detection algorithm. Labels are supplemented using the user agent string.

G_0 Statistics	
node count, $\|V\|$	1,278,838,913
edge count, $\|E\|$	7,305,293,292
average degree	11.4

Table 1: Summary statistics of the IP-colocation graph.

G_0 Statistics by Edge Type	
advertising ID – advertising ID	784,626,308
advertising ID – cookie	2,822,643,027
cookie – cookie	3,698,023,957
G_0 Statistics by Node Type	
advertising ID	343,520,426
cookie	935,318,487

Table 2: Summary statistics of the IP-colocation graph by device ID type.

component. A connected component is a portion of the graph such that every pair of vertices is connected by a path. Plots are shown for $\gamma = 0, 0.2, 0.4, 0.6, 0.8$ and 1.0.

Finally, Figure 1 shows a visualization of a small portion of $G_{0.4}$ (a 4-hop subgraph). Weights of the edges are indicated by the thickness of the line connecting two nodes. Each color denotes a community generated by Louvain Modularity, $\ell = 1$. The labels are supplemented using the user agent string.

3.3 Characteristics of the Communities

Louvain [16] community detection was applied to the IP colocation graph using a variety of cost functions and edge cut-off levels. Specifically, the cost functions employed were modularity, deviation to indetermination, deviation to uniformity and balanced modularity [18]. The edge cutoff levels were $\gamma = 0, 0.2, 0.4, 0.6, 0.8$ and 1.0. The cost functions were selected due to prevalence in literature, and availability in the open source code used to run community detection. In general the cost functions produced very similar communities, with the exception of balanced modularity, which produced significantly smaller communities.

	Largest Connected Component	Mean Shortest Path
G_0	1,084,255,633	9.6
$G_{0.2}$	761,850,044	15.3
$G_{0.4}$	444,211,103	23.3
$G_{0.6}$	132,247,459	48.5
$G_{0.8}$	21,916,077	99.2
$G_{1.0}$	112,191	43.7

Table 3: Number of nodes in the largest connected component as a function of edge weight and approximate mean shortest path between nodes in that component.

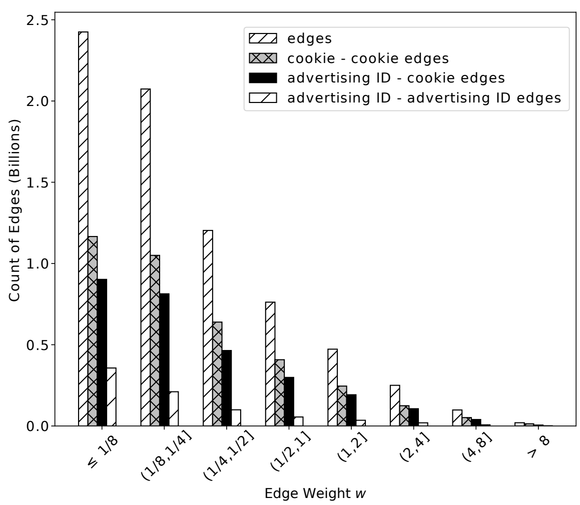

Figure 2: Edge weight distribution of the IP-colocation device graph G_0.

The choice of γ has a strong effect on the properties of the communities generated. As edges below a certain weight are culled from the graph, the number of singleton nodes (*i.e.,* nodes with only self edges above γ) increases and the density of connections both inside and across communities drop. Weaker connections between more loosely associated devices are excluded, which manifests as smaller communities. When γ is sufficiently large, the graph becomes increasingly disconnected (see Table 3), ultimately preventing formation of large communities. As user-level connections (*i.e.,* devices that belong to the same person) correlate with higher edge weight in the graph, communities produced with $\gamma = 1.0$ can be interpreted as groups of devices belonging to an individual. As the edge cutoff is reduce, *household* level connections become prevalent.

Each choice of cost function and cutoff level produces a *hierarchy* of communities as the Louvain approach performs greedy optimization by repeating two steps. First, the cost function is optimized locally by moving nodes into surrounding communities. This generates *small* communities, corresponding to devices that belong to the same *user* or *household.* In the second step, a new graph is built where nodes are the communities from the previous step. The first step is repeated, generating communities of communities.

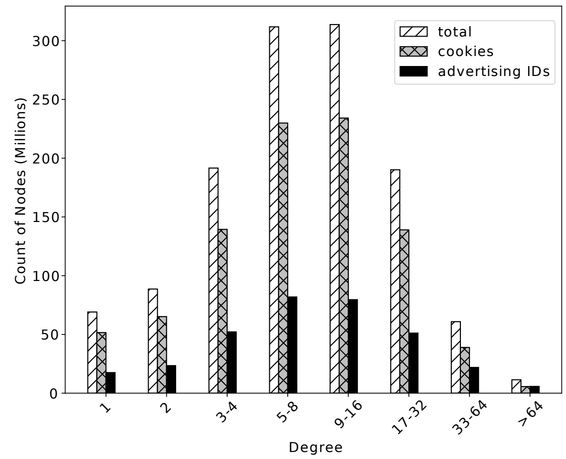

Figure 3: Degree distribution of the IP-colocation device graph G_0.

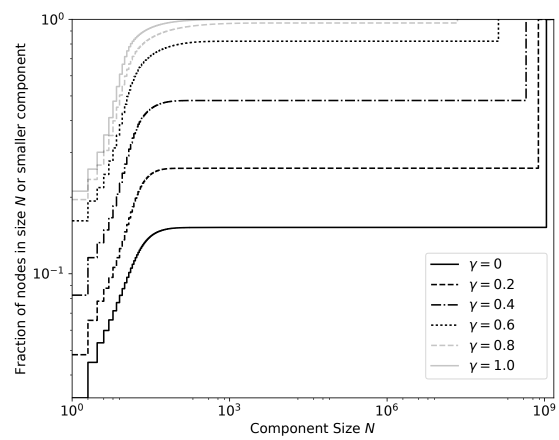

Figure 4: Connected component cumulative distribution. The plot show the fraction of nodes in a connected component of size N or smaller in G_γ.

Figure 5 shows the distribution of household sizes for the Louvain algorithm applied to $G_{0.4}$ with the various cost functions, hierarchy level $\ell = 1$. The cost functions ordered by increasing mean community size were balanced modularity, modularity, deviation to uniformity, and deviation to indetermination, producing mean communities of size 3.39, 4.79, 4.80, and 4.83.

Figure 6 shows the distribution of household sizes when modularity, hierarchy level $\ell = 1$, was run on G_0, $G_{0.2}$, $G_{0.4}$, $G_{0.6}$, $G_{0.8}$ and $G_{1.0}$. The mean community size decreases from 8.3 to 2.8 as γ is increased from zero to one, which is captured in Figure 7.

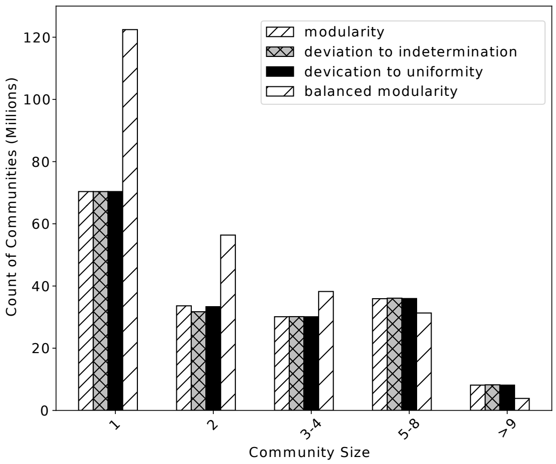

Figure 5: Distribution of community sizes generated by the Louvain algorithm, $\ell = 1$, with various cost functions applied to $G_{0.4}$.

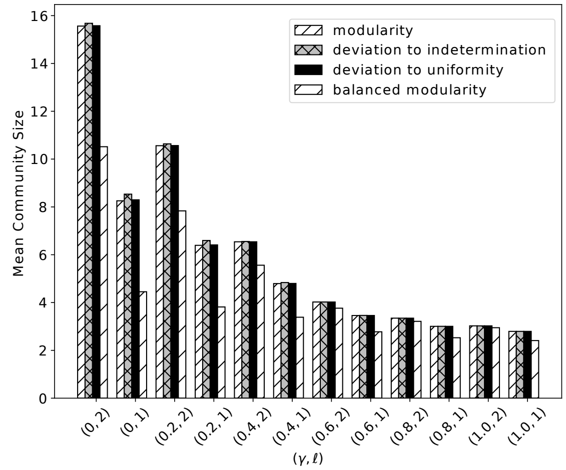

Figure 7: Mean community size at hierarchy levels $\ell = 1$ and $\ell = 2$ for Louvain community detection applied to G_0, $G_{0.2}$, $G_{0.4}$, $G_{0.6}$, $G_{0.8}$ and $G_{1.0}$.

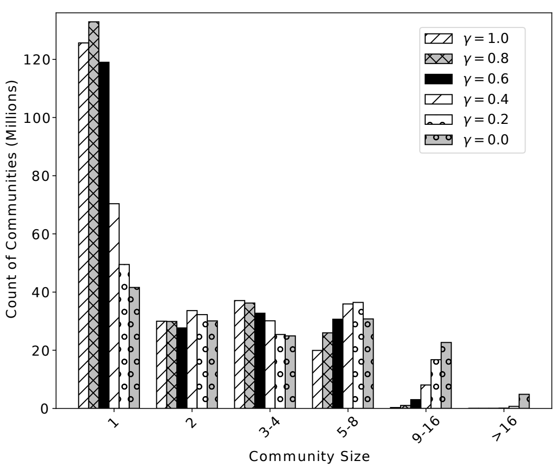

Figure 6: Distribution of community sizes generated by Louvain Modularity, $\ell = 1$, applied to G_0, $G_{0.2}$, $G_{0.4}$, $G_{0.6}$, $G_{0.8}$ and $G_{1.0}$.

3.4 Validation

Validation was facilitated by a ground truth dataset provided by the comScore Total Home Panel (THP), which collects data from customized wireless routers installed in a volunteer's home. The routers act as gateways to the internet capturing browsing statistics for all devices in the household. The router also records both an obfuscated version of the media access control (MAC) address, and the device IDs (3rd party cookies and advertising IDs) associated with any device that connects to the internet via the router.

The device IDs associated with each custom wireless router define the over 7,000 ground truth households, where each household is comprised of one or more device IDs. Only device IDs that correspond to MAC addresses active in the home for more than 2 days are included. This eliminates 'guest' devices that do not have a long term association with the household. The ground truth households were established using the same time frame as the graph: December 5th, 2016 - January 15th 2017.

With ground truth households from the home panel, *precision* and *recall* of the graph communities were calculated. More specifically, let $\mathcal{H}$ be the set of communities produced by the community detection algorithm applied to the graph under test. First, for a ground truth community H' (*i.e.,* a physical household), the corresponding graph community H^* is defined as the community which has the largest intersection of device IDs:

$$H^* := \arg\max_{H \in \mathcal{H}} |H' \cap H|. \tag{1}$$

If more than one such community exists, the smallest one is chosen. Precision and recall for H^* are given by

$$\text{precision} = \frac{|H' \cap H^*|}{|H^*|} \qquad \text{recall} = \frac{|H' \cap H^*|}{|H'|}. \tag{2}$$

A match is declared if precision are recall are both greater than 50%. Both the mean precision and recall, and the match rate are plotted in Figures 8 and 9.

Figure 8 shows the precision-recall curve for the various IP-colocation graphs and community detection algorithms tested. To create the precision-recall curves, Louvain was run on G_γ for γ values of 0 through 1 in increments of 0.2. As increasing γ reduces the number of edges, it reduces the size of the households produced by the Louvain algorithm, creating a precision/recall curve. Precision and recall are reported for the four cost functions considered in addition to hierarchy output levels $\ell = 1$ and $\ell = 2$.

We note a few values from the precision-recall curves. Louvain Modularity applied to $G_{0.4}$ produced a mean precision of 90.8% and recall of 83.8% across 6,401 matched ground truth households, indicating that discarding low weight edges is a viable approach to

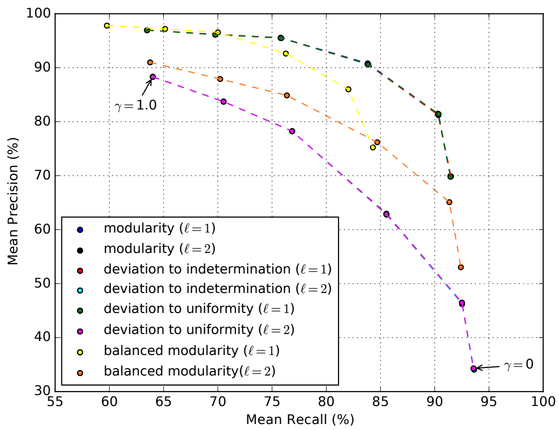

Figure 8: Mean precision-recall curve showing verification of the community detection algorithms against the ground truth communities corresponding to physical households.

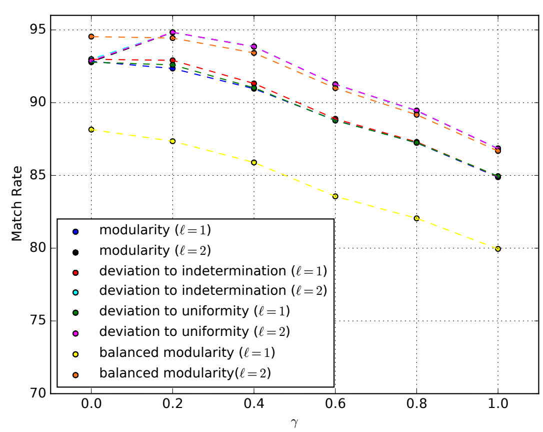

Figure 9: Percent of ground truth communities (physical households) that were successfully matched to the graph based communities. A graph community is considered to be successfully matched if there exists a ground truth household with greater than 50% precision and recall. The ground truth households are composed of device identifiers that connect to wireless routers in 7,037 physical households.

generating accurate and appropriately sized communities. Deviation to indetermination and uniformity produced similar precision recall curves. The results indicate overwhelming success.

4 PRIVACY CONSIDERATIONS

While any dataset that contains device identifiers can pose privacy questions, the scale and user-centric focus of device graphs has the potential to raise new privacy concerns. At the forefront of these concerns is association with personally identifiable information (PII), and special consideration must be given to preclude such associations. To that end, graph data structures are attractive from a privacy perspective since they capture pair-wise relationships, and do not disclose the information used to generate the relationship. In other words, even if a device graph is constructed from information that may be considered sensitive (*e.g.,* shared login or shared PII), it can be stored and analyzed without any of the sensitive information.

Specific to our IP colocation-based device graph, additional measures mitigate potential disclosure of sensitive information. All of the IP addresses used to generate graphs for this paper were obfuscated. The obfuscation is based on time dependent one way hash of the IP address that allows devices to be associated with each other, but not a physical IP. Furthermore, device identifiers in our dataset, both cookies and advertising IDs, are stored as a one way hash. All device IDs are further assigned an integer value before processing, and the mapping back to the one way hashed version of the ID is not stored for the purposes of this study. While these obfuscations of the device IDs can limit the utility of a graph in practical applications, the assumption is that these measures in full would not be necessary for proprietary application. We refer the reader to comScore's privacy policy for further discussion [2] and the policies specified by device graph vendors (*e.g.,* [3, 5, 8]).

There are practical steps that users can take to limit the chances that their devices are included in a device graph. In particular, browser privacy settings that reject third party cookies prevent inclusion of a browser cookie in our device graph. Moreover, *incognito* browsing, which creates session cookies that are deleted when the browsing session is complete, largely precludes inclusion, as the IP colocation device graph only includes cookies that have a lifetime greater than 24 hours. In regards to advertising IDs, OS privacy settings allow users to opt out and preclude inclusion of their advertising ID in the graph. More broadly, we appeal to the large literature on user privacy in the web (*e.g.,* [36]) and methods to limit information leakage (*e.g.,* [45]).

Lastly, The Menlo Report [23] specifies core ethical principles, provides context for these in computing and communication research, and provides guidance on Institutional Review Board (IRB) and self evaluation. This report and subsequent discussions in [34] form the basis for our consideration of ethics in device graphing.

5 RELATED WORK

To the best of our knowledge there are no prior research papers on algorithms or systems for internet device graphing. While there are several companies that have device graph offerings (*e.g.,* [3, 5, 8]), they publish very little or nothing about their data or methods. One of the objectives of our work is define the problem, and describe and demonstrate how it might be addressed. Related work largely falls into three areas: large graphs and community detection (*e.g.,* social networks), IP and dynamics, and internet measurement which includes mapping home networks and mapping internet topology.

Large Graphs and Community Detection. Large graphs have been extensively studied in recent years. Many of the publicly available graph datasets that have been extensively studied are orders of magnitude smaller than the IP colocation graph (see the online databases of publicly available graph datasets [7] and [4]). Nonethe-less, analysis of graphs of similar size do exists, and include

[12], [52]. A broad overview of community detection is provided by Fortunato in [28]. Classical methods for community detection in graphs include graph partitioning (*e.g.,* [21]), hierarchical clustering (*e.g.,* [41]) and spectral clustering (*e.g.,* [53]). Greedy techniques (*e.g.,* [20, 42]) are favorites for large graphs since they have low computational complexity. The Louvain [16] approach is a popular greedy algorithm, since it is reported to detect communities with high accuracy and low computational complexity at immense scale. An inherent shortcoming of the modularity based techniques is the *resolution limit* [29], which prevents small communities from being accurately detected in large networks.

It is important to highlight the immense size of our device graph in comparison to much of the community detection literature, with some exceptions. The WebBase 2001 graph, which is used in the original Louvain Modularity paper has 118 million nodes and 1 billion edges [16]. While other studies propose parallel algorithms for similarly sized graphs [44], we are unaware of any implementations on graphs larger than G_0. Many of the studies in the literature [30–32, 43] that employ and compare Louvain Modularity are conducted on graphs that are an order of magnitude smaller than our device graph. To the best of our knowledge, the IP-colocation graph is the largest graph to which Louvain algorithms have been applied [3].

Community structures specific to the internet have been considered for some time, and are prolific in the large scale community detection literature. Examples include, Flake *et al.* which describes how self-organization in web link structures enables identification of interest communities [27]. Backstrom *et al.* provided one of the earlier studies on communities in social networks by characterizing relationships in LiveJournal and DBLP [13]. Other studies have examined how graph structures of social connections evolve over time [37], inferring user profiles based on community connections [40] and social structures in specific networks [51].

IP Dynamics. IP address assignments to internet-connected devices can be assigned statically or dynamically. Static assignments may be used in small networks but are difficult to manage at scale. Thus, dynamic assignment *e.g.,* via DHCP are the norm.

Prior studies have examined the stability of IP address assignments to individual devices. Central to these studies is the need for persistent identification of the same device. Xie *et al.* use the UDmap to to identify dynamically assigned IP addresses in [54]. The key to persistent device identification is using Hotmail user login names. Casado and Freedman use web cookies to track clients accessing CDNs in [19]. We are informed by their client tracking method, however the objectives of their study differ significantly from ours. Indeed, the main advantage to our approach (over use of the IP itself to define a household) is resilience to IP dynamics.

Network address translation (NAT) enables generation of the IP colocation device graph by allowing multiple devices to share a single IP address. Prolific use of NATed IP addresses by wireless access points are paramount to the utility of the proposed graph. A useful recent assessment of the prevalence of NAT in the internet was presented by Akamai [6]. There have been many studies that attempt to estimate the number of devices behind NATs *e.g.,* [14, 35, 39]. All of these rely on clients revealing themselves through a feature such as IPIDs, a hardware fingerprint or a combination of features, which might be useful in future device graphing studies.

Internet Measurement. Measurement and characterization of internet structure and topology has been an active area of research for many years. An objective in these studies is to establish accurate (*i.e.,* unique identification of nodes and links) network-layer maps of service provider networks (*e.g.*, [47, 50]). A number of studies have developed new techniques for data collection that improve accuracy and mapping coverage, *e.g.*, [10, 11, 24, 48]. Other efforts have considered the inherent accuracies in probe-based network mapping [26, 46, 49, 55] and how Internet Exchange Points (IXPs) have had an impact on the accuracy of network-layer maps, *e.g.*, [9, 11]. Our work differs from these studies in that it is based on data that is typically collected directly from devices *e.g.,* via Javascript tags or passive network traces, and our goal is in understanding relations between devices.

Finally, there are several studies that are more closely related to ours in their goals and approach. Heidemann *et al.* report on the population of devices that respond to ping-style measurements in [33]. We believe that enhanced active probe-based measurements are a possible starting point for future device graph studies. The problem of measuring and characterizing home networks is also related to our work. For example, DiCioccio *et al.* develop a tool called HomeNet Profiler, which when deployed within a home network can collect information about WiFi connected devices [22]. HomeNet Profiler is similar in some respects to the home panel that we use in validating our device graph. The limitation of many of those methods to generation of device graphs is that they are not applicable at scale. However, prior studies that use probe-based methods to gather device information broadly in the internet (*e.g.,* [25, 33]) coupled with methods for device identification behind NATs (*e.g.,* [35, 39]) do suggest possible approaches to assembling datasets suitable for IP-colocation device graphing.

6 SUMMARY

This paper introduces device graphing. We describe a method for device graphing based on IP-colocation, which enables generation of graphs of immense scale. We complement this with Louvian community detection to generate groupings of devices that have utility in practical applications. We implement our device graphing approach and community detection methods to 2.5 trillion records, resulting in a graph with over 7 billion edges and over 1.2 billion nodes, and communities corresponding to 100 million households in the US. The accuracy of the grouping of devices into households is validated through the use of a unique, ground truth dataset.

While the graph and communities exhibit immense scale and strong accuracy, there are a handful of opportunities for improvement. Devices that connect to the internet exclusively via mobile carrier will not be identified in the IP-colocation graph. Future complications, including IPv6, and changes in residential service provider IP allocation, must be addressed. Additionally, it is common place that multiple device IDs are transmitted by a single device. This occurs not only when a user clears cookies or resets an advertising ID, but simply when a device presents an advertising ID *and* one or more browser cookies. Special consideration can be given to these scenarios and we leave this to future work.

[3] While tuning the algorithms for graphs of this size presents practical challenges, one could argue this is not particularity important since running the algorithm simply required a machine with sufficient memory.

REFERENCES

[1] Apache Pig. https://pig.apache.org. Accessed: 2016-02-10.
[2] comScore Privacy Policy. https://www.scorecardresearch.com/privacy.aspx. Accessed: 2017-02-17.
[3] Drawbridge. http://www.drawbridge.com/c/graph/. Accessed: 2017-02-17.
[4] The Koblenz Network Collection. http://konect.uni-koblenz.de/networks/. Accessed: 2017-02-15.
[5] Lotame Cross Device. http://www.lotamecrossdevice.com. Accessed: 2017-02-17.
[6] Prevalence of Network Address Port Translation (NAPT). https://www.ietf.org/proceedings/93/slides/slides-93-hopsrg-1.pdf. Accessed: 2016-09-19.
[7] Stanford network analysis project. http://snap.stanford.edu/data/. Accessed: 2017-02-15.
[8] The Tapad Device Graph. http://www.tapad.com/device-graph/, 2016.
[9] Ager, B., Chatzis, N., Feldmann, A., Sarrar, N., Uhlig, S., and Willinger, W. Anatomy of a Large European IXP. In *Proceedings of ACM SIGCOMM conference* (2012).
[10] Augustin, B., Cuvellier, X., Orgogozo, B., Viger, F., Friedman, T., Latapy, M., Magnien, C., and Teixeira, R. Avoiding Traceroute Anomalies with Paris Traceroute. In *Proceedings of ACM SIGCOMM Internet measurement conference* (2006).
[11] Augustin, B., Krishnamurthy, B., and Willinger, W. IXPs: Mapped? In *Proceedings of ACM Internet measurement conference* (2009).
[12] Backstrom, L., Boldi, P., Rosa, M., Ugander, J., and Vigna, S. Four degrees of separation. In *Proceedings of the 4th Annual ACM Web Science Conference* (2012), ACM, pp. 33–42.
[13] Backstrom, L., Huttenlocher, D., Kleinberg, J., and Lan, X. Group Formation in Large Social Networks: Membership, Growth, and Evolution. In *Proceedings of ACM SIGKDD* (2006).
[14] Bellovin, S. A Technique for Counting NATted Hosts. In *Proceedings of ACM Internet Measurement Workshop* (November 2002).
[15] Blondel, V., Guillaume, J.-L., and Aynaud, T. http://sourceforge.net/projects/louvain/.
[16] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. Fast Unfolding of Communities in Large Networks. *Journal of Statistical Mechanics: Theory and Experiment 2008*, 10 (2008), P10008.
[17] Cahn, A., Alfeld, S., Barford, P., and Muthukrishnan, S. An Empirical Study of Web Cookies. In *Proceedings of the 25th International Conference on World Wide Web* (2016), International World Wide Web Conferences Steering Committee, pp. 891–901.
[18] Campigotto, R., Céspedes, P. C., and Guillaume, J.-L. A generalized and adaptive method for community detection. *arXiv preprint arXiv:1406.2518* (2014).
[19] Casado, M., and Freedman, M. Peering Through the Shroud: The Effect of Edge Opacity on IP-based Client Identification. In *Proceedings of the USENIX NSDI* (April 2007).
[20] Clauset, A., Newman, M. E., and Moore, C. Finding Community Structure in Very Large Networks. *Physical review E 70*, 6 (2004), 066111.
[21] D. Spielman, S. T. Nearly-linear Time Algorithms for Graph Partitioning, Graph Sparsification, and Solving Linear Systems. In *Proceedings of ACM Symposium on Theory of Computing* (2004).
[22] DiCioccio, L., Teixeira, R., and Rosenberg, C. Measuring Home Networks with HomeNet Profiler. In *Proceedings of the Passive and Active Measurement Conference* (March 2013).
[23] Dittrich, D., and Kenneally, E. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. In *US Department of Homeland Security Tech Report* (August 2012).
[24] Durairajan, R., Sommers, J., and Barford, P. Layer 1-Informed Internet Topology Measurement. In *Proceedings of ACM Internet Measurement Conference* (November 2014).
[25] Durumeric, Z., Wustrow, E., and Halderman, A. ZMap: Fast Internet-Wide Scanning and its Security Applications. In *Proceedings of the USENIX Security Symposium* (August 2013).
[26] Eriksson, B., Barford, P., Sommers, J., and Nowak, R. Inferring Unseen Components of the Internet Core. *IEEE Journal on Selected Areas in Communications* (2011).
[27] Flake, G., Lawrence, S., Giles, C., and Coetzee, F. Self-organization and Identification of Web Communities. *IEEE Computer 35*, 3 (August 2002).
[28] Fortunato, S. Community Detection in Graphs. *Physics Reports 486*, 3 (February 2010).
[29] Fortunato, S., and Barthelemy, M. Resolution Limit in Community Detection. *Proceedings of the National Academy of Sciences 104*, 1 (2007), 36–41.
[30] Greene, D., Doyle, D., and Cunningham, P. Tracking the Evolution of Communities in Dynamic Social Networks. In *Advances in social networks analysis and mining (ASONAM), 2010 international conference on* (2010), IEEE, pp. 176–183.
[31] Harenberg, S., Bello, G., Gjeltema, L., Ranshous, S., Harlalka, J., Seay, R., Padmanabhan, K., and Samatova, N. Community Detection in Large-scale Networks: a Survey and Empirical Evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics 6*, 6 (2014), 426–439.
[32] Haynes, J., and Perisic, I. Mapping Search Relevance to Social Networks. In *Proceedings of the 3rd Workshop on Social Network Mining and Analysis* (2009), ACM, p. 2.
[33] Heidemann, J., Pradkin, Y., Govindan, R., Papadopoulos, C., Bartlett, G., and Bannister, J. Census and Survey of the Visible Internet. In *Proceedings of ACM Internet Measurement Conference* (October 2008).
[34] Kenneally, E., and Bailey, M. Cyber-security Research Ethics Dialogue and Strategy Workshop. *ACM SIGCOMM Computer Communication Review 4*, 2 (2013).
[35] Kohno, Y., Broido, A., and Claffy, K. Remote Physical Device Fingerprinting. *IEEE Transactions on Dependable and Secure Computing 2*, 2 (April 2005).
[36] Krishnamurthy, B., and Wills, C. Privacy Diffusion on the Web: A Longitudinal Perspective. In *Proceedings of the 18th International Conference on World Wide Web* (New York, NY, USA, 2009), WWW '09, ACM, pp. 541–550.
[37] Kumar, R., Novak, J., and Tomkind, A. *Structure and Evolution of Online Social Networks*. Springer, 2010.
[38] Leskovec, J., and Sosič, R. Snap: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST) 8*, 1 (2016), 1.
[39] Maier, G., Schneider, F., and Feldmann, A. NAT Usage in Residential Broadband Networks. In *Proceedings of the Passive and Active Measurement Conference* (March 2011).
[40] Mislove, A., Viswanath, B., Bummadi, K., and Druschel, P. You are Who You Know: Inferring User Profiles in Online Social Networks. In *Proceedings of ACM International Conference on Web Search and Data Mining* (February 2010).
[41] Murtagh, F. A Survey of Recent Advances in Hierarchical Clustering Algorithms. *Computer Journal 26*, 4 (1983).
[42] Newman, M. E. J. Fast Algorithm for Detecting Community Structure in Networks. *Phys. Rev. E 69* (Jun 2004), 066133.
[43] Pujol, J. M., Erramilli, V., and Rodriguez, P. Divide and Conquer: Partitioning Online Social Networks. *arXiv preprint arXiv:0905.4918* (2009).
[44] Riedy, E. J., Meyerhenke, H., Ediger, D., and Bader, D. A. Parallel Community Detection for Massive Graphs. In *International Conference on Parallel Processing and Applied Mathematics* (2011), Springer, pp. 286–296.
[45] Roesner, F., Kohno, T., and Wetherall, D. Detecting and Defending Against Third-party Tracking on the Web. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation* (Berkeley, CA, USA, 2012), NSDI'12, USENIX Association, pp. 12–12.
[46] Roughan, M., Tuke, S. J., and Maennel, O. Bigfoot, Sasquatch, the Yeti and other Missing Links: what we don't know about the AS Graph. In *Proceedings of ACM Internet measurement conference* (2008).
[47] Sherwood, R., Bender, A., and Spring, N. Discarte: a Disjunctive Internet Cartographer. In *Proceedings of ACM SIGCOMM conference* (2008).
[48] Sherwood, R., and Spring, N. Touring the Internet in a TCP Sidecar. In *Proceedings of ACM Internet Measurement Conference* (2006).
[49] Spinelli, L., Crovella, M., and Eriksson, B. AliasCluster: A Lightweight Approach to Interface Disambiguation. In *Proceedings of the Global Internet Symposium* (2013).
[50] Spring, N., Mahajan, R., and Wetherall, D. Measuring ISP Topologies with Rocketfuel. *ACM SIGCOMM conference* (2002).
[51] Traud, A., Mucha, P., and Porter, M. Social Structure of Facebook Networks. *Physica A: Statistical Mechanics and its Applications 391*, 16 (August 2012).
[52] Ugander, J., Karrer, B., Backstrom, L., and Marlow, C. The anatomy of the facebook social graph. *arXiv preprint arXiv:1111.4503* (2011).
[53] von Luxburg, U. A Tutorial on Spectral Clustering. *Statistics and Computing 17*, 4 (December 2007).
[54] Xie, Y., Yu, F., Achan, K., Gillum, E., Goldszmidt, M., and Wobber, T. How Dynamic are IP Addresses? In *ACM SIGCOMM Computer Communication Review* (2007), vol. 37, ACM, pp. 301–312.
[55] Zhang, Y., Oliveira, R., Wang, Y., Su, S., Zhang, B., Bi, J., Zhang, H., and Zhang, L. A Framework to Quantify the Pitfalls of using Traceroute in AS-level Topology Measurement. *IEEE Journal on Selected Areas in Communications* (2011).