Can't See Forest through the Trees? Understanding Mixed Network Traffic Graphs from Application Class Distribution

Yu Jin*, Nick Duffield*, Patrick Haffner*, Subhabrata Sen*, Zhi-Li Zhang†

*AT&T Labs - Research, Florham Park, New Jersey, USA
†Computer Science Dept., University of Minnesota, Minneapolis, Minnesota, USA
*{yjin,duffield,haffner,hsu,guy,sen,shvenk}@research.att.com †zhzhang@cs.umn.edu

ABSTRACT

In this paper we study the interaction patterns among traffic from different application classes, namely, *how they collaboratively form a mixed traffic activity graph* (mixed TAG). Utilizing real traffic traces from a major ISP and a large university network, we show that densely connected subgraphs or clusters are the building blocks for a mixed TAG. These subgraphs can be either dependent or independent for different application classes. In addition, clusters from different application classes exhibit repulsive/attractive relationships while they interconnect to form a mixed TAG. We then propose a variant of the Markov clustering algorithm to extract these clusters. Analysis of the clusters show that though mixed TAGs display similar structures, many core components are time/location specific. The clustering results also motivate us to develop an accurate, semi-supervised learning based traffic classification algorithm.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations

General Terms

Measurement, Management

1. INTRODUCTION

With increasing diversity and complex interactions among network entities (e.g., p2p applications, web services with interweaved webs of content providers and ad networks, social networks, and malicious botnets), characterizing and understanding communication and interaction patterns in network traffic from a *network-wide* perspective is imperative. Such understanding is critical to managing and securing today's ISP networks, from traffic classification [1, 2] to network trouble-shooting and anomaly detection [3–5]. By representing the interactions of hosts engaging in the same application in the network as *traffic activity graphs* (TAGs), recent studies in [6–8] demonstrate the effectiveness of using graphs not only as a visualization tool but also as an analysis tool to characterize the unique communication patterns of individual applications, analyze and extract communities of interests therein, and study their evolution. These prior studies, however, focus primarily on *application-specific* TAGs, and thus presume that traffic from different applications can be separated *a priori* in some fashion.

In this paper, we study the network-wide communication and interaction patterns in network traffic with *mixed* applications, and introduce the notion of *mixed* TAGs (see Section 2 for a formal definition). Unlike the *application-specific* TAGs introduced in [6] where edges represent interactions of hosts belonging to the same application, edges in a mixed TAG can represent interactions of *different* applications. To differentiate these applications, we visualize mixed TAGs as if each edge were annotated with a label (or color), signifying the application class it belongs to. The main focus of this paper is to understand not only how hosts interact with other hosts engaging in the same applications, but how they interact with hosts engaging in different applications, and how they collectively form a mixed TAG. For this purpose, we collect real network traffic traces from a major ISP and a large university network and apply a bottom-up approach to study the formation of mixed TAGs. To describe various application traffic in the network, we define 10 application classes summarizing applications with similar functionality. For example, `P2P` contains p2p applications like BitTorrent, eMule, etc., and `Chat` includes IRC, MSN messenger and Yahoo messenger, and so forth.

At the lowest level (see Section 2), we analyze the spatial distribution of edges from different application classes. Using shortest path distance as a metric, we find that edges from different classes often exhibit strongly connected subgraphs or clusters, and we interpret the interconnection of clusters in application-specific TAGs using two generative block models. The first model, *dependent block model*, describes the generation process of `Mail`, `Chat` and `Web` graphs, where a few central clusters attract more connections than others, resulting in a large and dense kernel in the graph. In contrast, the second model, *independent block model*, explains the formation of `Games`, `Media` and `P2P` graphs, where a number of dense clusters are connected by a few random edges. In addition to this, we further show that edges from different applications are not randomly connected. Instead, they display interesting relationships. For example, `P2P` edges are less likely to connect to `VoIP` edges, thereby exhibiting a *repulsive relationship*. In comparison, `Games`

MLG '11 San Diego, CA, USA

and P2P edges are more likely to be neighbors, and hence showing an *attractive relationship*.

Knowing that mixed TAGs are built on top of interconnected clusters, in Section 3 we propose an algorithm to decompose mixed TAGs. Our algorithm is a variant of the Markov Clustering (MCL) algorithm which incorporates edge properties in the clustering process to obtain clean clusters. Experiment results show that the proposed algorithm can successfully decompose mixed TAGs into clusters with high purity. These results also enable us to analyze the mixed TAGs at the level of clusters. Ranking all the clusters using the *closeness* metric, we identify "core" clusters that contribute most to the global connectivity of mixed TAGs. In addition to the common core clusters shared by different TAGs, such as clusters of popular web/mail servers, CDN and advertising servers, we observe core components that are time/location dependent. For example, in mixed TAGs from the university network in 2006, we find core clusters related to news websites and job hunting websites. In contrast, clusters related to antivirus websites and Youtube show up as core clusters in mixed TAGs from the ISP network in 2009. These results are presented in Section 4.

2. MIXED TRAFFIC ACTIVITY GRAPHS

In this section, we first introduce the datasets and advance the notion of mixed traffic activity graphs (mixed TAGs) as a tool for describing network traffic mix. We next study the formation of mixed TAGs from two aspects: how each application-specific TAG is generated and how these TAGs are interconnected to form a mixed TAG.

2.1 Datasets and Application Classes

There are two datasets used in our study. The first dataset (referred to as the *ISP dataset*) contains network flow records from a major ISP over one month period in 2009. A flow is a sequence of packets with a common key – namely, the standard 5-tuple of IP protocol, source and destination IP addresses, and TCP/UDP ports – that are localized in time. Flow measurements comprise summary statistics that aggregate information derived from a flow's packet headers (including the key, aggregate packet and byte counts for the flow, and timing information) that are exported as IP flow records to a collector. The second dataset (referred to as the *U dataset*) comprises Cisco NetFlow records collected from a large university network over a month period in 2006. One out of 20 flow sampling is applied to the ISP dataset and the U dataset is unsampled. In this paper, we only focus on TCP traffic, however, similar approach can be readily applied for analyzing UDP traffic.

Serving as the *ground truth* for our study, the flow records in the ISP dataset are annotated with a number of broad "application class" *labels*. Motivated by the management tasks of large ISP networks, we define 9 broad application class labels, as shown in Table 1. Similar to [1,9], the labels are generated in an automated way by the measurement devices, using a set of packet-level rules based on combinations of packet signatures that operate on layer-4 packet header information, and layer-7 application protocol signatures. The flow records do not include any application data; neither do they report any user identity information. In comparison, the U dataset is labeled manually based on popular service ports associated with different application classes [10]. We note there is no dominant service port for certain application classes. Therefore, in the U dataset we only label a subset of the application classes. In total, around 30% of the ISP flows and 34% of the U flows cannot be classified, and we assign them to an Unknown class.

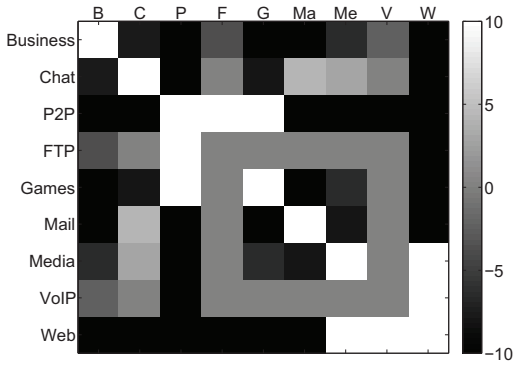


Figure 2: Pearson's residuals.

2.2 Mixed TAGs

Application-specific traffic activity graphs (TAGs) have been applied to characterize traffic from a single application (class) [6]. However, real network traffic typically consists of traffic from various application classes. Here we propose the notion of mixed TAGs as an extension to the original TAGs for describing and analyzing such traffic mix.

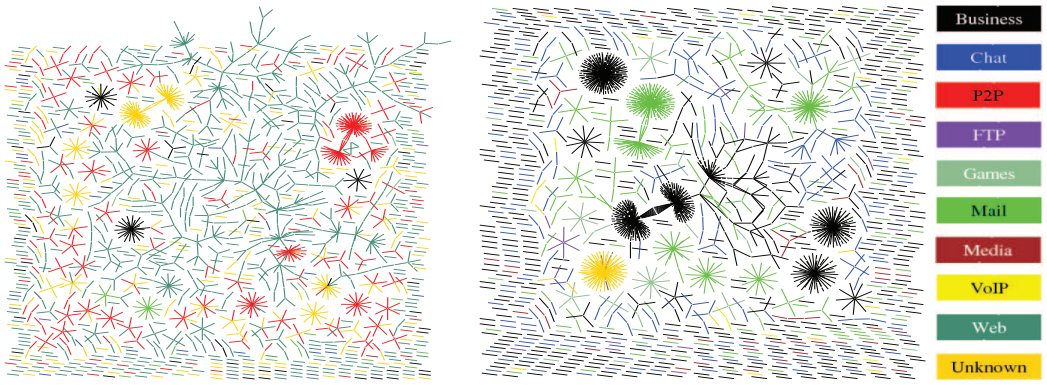
Given time T, let $\mathcal{H}$ denote the set of observed endpoints. A mixed TAG, $\mathcal{G} = \{\mathcal{H}, \mathcal{E}, \mathcal{X}, L\}$, is defined as follows: we include an edge e_i^{pq} in the edge set $\mathcal{E}$ if and only if at least one flow is observed between h_p and h_q. For simplicity, we drop the superscript of e in the remainder of the paper. We then define the *mixed* TAG by labeling each edge of the TAG using the (dominant) application class label of the flows between the two endpoints of the edge. We use $L(e_i)$ as the (dominant) application class label associated with e_i. We note that we assume each communication peer is only involved in one application, which accounts for more than 99.6% of the cases in our datasets, even when the observation period T is extended to a day. As illustrated in Table 1, there are 10 application classes (including Unknown). We further define $\mathbf{x}_i \in \mathcal{X}$ as the traffic features associated with e_i, e.g., number of packets/bytes, packet interarrival rate, etc., depending on application scenarios.

Fig. 1[a] illustrates an instance of the mixed TAG constructed using the first 2000 edges in the ISP dataset from 04/06/2009, staring at 7PM. Both plots are drawn using Graphviz tools [11] with default parameters. We annotate different application classes with different colors. For better visualization, we remove the dominant P2P, Web and Unknown traffic and plot other traffic in Fig. 1[b]. We observe that there are many densely connected subgraphs (of different colors) in the mixed TAG, which are distributed at different locations, with varying shapes and density. In addition, most of the clusters are dominated by edges from a single traffic class, though they often contain a few edges from other classes.

When the observation time is extended, the large number of edges in mixed TAGs prevents us from direct visualization of them. In the following, we study the spatial distribution of application classes in mixed TAGs in a quantitative way.

Index	Class/Label	Example Applications	Ports
1	`Business`	Middleware, VPN, etc.	-
2	`Chat`	Messengers, IRC, etc.	5222, 5190, etc.
3	`P2P`	P2P applications	4661, 4662, etc.
4	`FTP`	FTP application	21, 20
5	`Games`	Online gaming applications, e.g., Everquest, WoW, Xbox, etc.	-
6	`Mail`	Email applications, e.g., SMTP and POP	25,993,etc.
7	`Media`	Video/audio streaming applications, e.g., RTSP, MS-Streaming, etc.	-
8	`VoIP`	Voice-over-IP application	-
9	`Web`	HTTP application	80, 443
10	`Unknown`	-	-

Table 1: Broad Application Classes



(a) all applications (b) `P2P`, `Web` and `Unknown` removed

Figure 1: TAGs containing 2000 edges.

2.3 Quantifying Traffic Spatial Distribution

The spatial distribution of a traffic class can be characterized by the distances among pairs of edges from the same traffic class. In particular, we denote $SPD(e_i, e_j)$ as the shortest path distance from e_i to e_j in the TAG. Note a shortest path may contain intermediate edges from other classes.

Using a 10-minute ISP TAG (ISP-10) and a 20-minute ISP TAG (ISP-20) collected on 04/06/2009, both beginning at 7PM, we show the SPD distributions of all traffic classes in terms of the 25% and 75% quantiles in Table 2. For comparison purpose, we also show in the table the same statistics for a 10-minute U TAG (U-10) and a 20-minute U TAG (U-20), both starting from 10:00AM on 02/03/2006.

Class	ISP-10		ISP-20		U-10		U-20	
	.25	.75	.25	.75	.25	.75	.25	.75
`Business`	5	8	5	6	-	-	-	-
`Chat`	7	9	6	7	6	8	5	6
`P2P`	6	8	6	9	6	8	7	8.5
`FTP`	6	9	6	7	4	9	-	-
`Games`	8	10	9	10	-	-	-	-
`Mail`	4	7	3	6	4	7	3	6
`Media`	5	6	5	7	-	-	-	-
`VoIP`	6	8	6	8	-	-	-	-
`Web`	5	6	4	5	4	7	4	6

Table 2: 25% and 75% quantiles of SPD's.

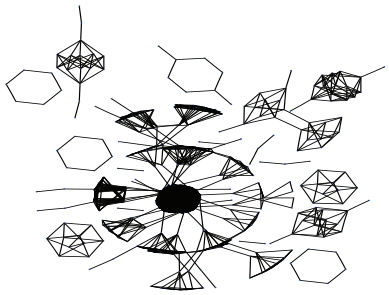
An interesting observation is made when we compare the results of the 10-minute TAGs with the 20-minute TAGs. Notice that when the observation period is extended, the density of TAGs, which is defined as the number of edges divided by the maximum number of allowable edges in a TAG, becomes lower in general. For example, the density for the 10-minute ISP TAG is 7.3e-4 and the density for the 20-minute ISP TAG is only 3.8e-4, nearly halves. If all edges are distributed uniformly in the TAG, we expect an increase in all the SPD's. This assumption holds for `P2P`, `Games` and `Media`. However, for `Chat`, `FTP`, `Mail` and `Web`, the corresponding SPD's decrease instead. In the following, we propose two hypothetic models to explain the generative process of these two types of TAGs.

2.4 Dependent/Independent Block Models

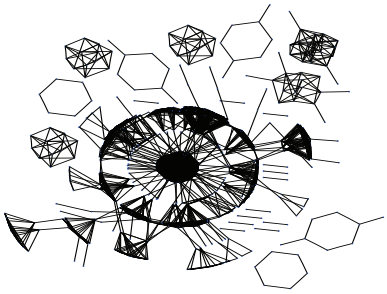
We conjecture there are two major types of formation of (application-specific) TAGs. They both can be considered as special forms of the classical block model [12].

The first model is referred to as the *dependent block model*, with examples like `Mail`, Chat and `Web` TAGs, etc.. The blocks (clusters or dense subgraphs) forming these application TAGs are correlated, with a few "central" blocks attracting more edges than others. Fig. 3 show the `Mail` TAGs from the 10-minute and the 20-minute ISP TAGs as an example. In these TAGs, the probability that an endpoint accesses a central block, given it has accessed a peripheral block, is large. Therefore, these TAGs often exhibit a dense kernel. When more edges are added, we observe the kernel to show stronger connectivity so that SPD's are expected to decrease.

In contrast, the second model (*e.g.* `Games`, `Media` and `P2P` TAGs) is referred to as the *independent block model*, where the corresponding TAGs are formed as a combination of multiple blocks which are independent of each other. Fig. 4[a,b] show the `Games` TAGs from the 10-minute and the 20-minute ISP TAGs as an example. We observe that

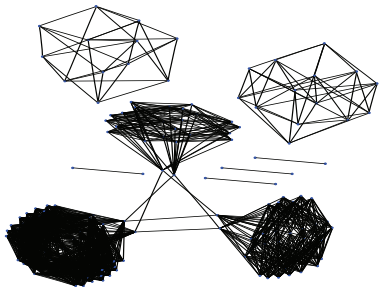


(a) 10-minute `Mail` TAG

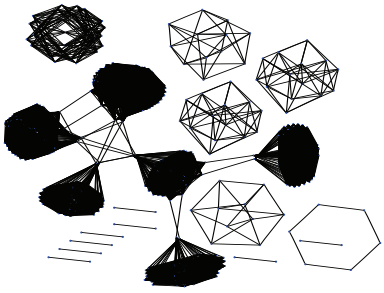


(b) 20-minute `Mail` TAG

Figure 3: TAGs for `Mail` traffic.



(a) 10-minute `Games` TAG



(b) 20-minute `Games` TAG

Figure 4: TAGs for `Games` traffic.

with the increase in the number of edges, new clusters appear. However, only a few random edges connect these clusters and most clusters are isolated (connected through edges from other classes). When the TAG expands, these blocks generally become more dispersed in the TAG, resulting in an increase in the corresponding SPD's.

Knowing that the application-specific TAGs are likely generated by the two block models, the next question we want to answer is: are these application-specific TAGs connected randomly to form the mixed TAG? In the following, we study the correlation of different application classes.

2.5 Application Class Relationships

Let $\pi_i, 1 \leq i \leq 9$ denote the proportions of edges belonging to the 9 application classes, excluding `Unknown`. We measure the correlation between two classes i and j using standardized Pearson's residuals defined below:

$$r_{ij} = \frac{n_{ij} - \mu_{ij}}{\sqrt{\mu_{ij}(1-\pi_i)(1-\pi_j)}},$$

where n_{ij} is the number of occurrence that an edge from class i and an edge from class j are neighbors, and $\mu_{ij} = N\pi_i\pi_j$ is the expected number of neighboring edge pairs from these two classes, under the hypothesis that edges from i and j are randomly distributed in the TAG. The magnitude of r_{ij} stands for the deviation from the hypothesis of random class distribution, and the sign of r_{ij} indicates whether the correlation is positive or negative.

We show the standardized Pearson's residuals in Fig. 2. For better visualization, we bound the residuals within [-10, 10]. A light/dark grid represents a positive/negative residual. We summarize three major types of correlations below.

Clustering Effect. Most of the diagonal grids show a strong positive self-correlation of the corresponding traffic classes. In other words, edges from the same class tend to cluster together, thereby showing dense subgraphs or block structures. This validates the existence of clusters in mixed TAGs. Detailed inspection further shows that many (large) uni-labeled clusters are due to the inherent client/server or p2p structure in the applications. For example, given a `Web` edge, we know that one of the endpoints must be an HTTP server. Since most HTTP servers support `Web` traffic exclusively, other edges connect to the same HTTP server are more likely to be `Web`.

Repulsive Relationship. In addition to the clustering effect, there are also strong *repulsive* and *attractive* effects appearing in the colored TAGs. By the *repulsive* relationship, we mean that given edges from two application classes, the presence of one significantly *reduces* the chance the other appears among the edges incident on one of the endpoints (or equivalently, increases the probability that the other class is *absent* among those edges). The repulsive relationship is likely due to that servers typically provide only one particular type of service exclusively. For instance, `Web`, `Games` and `Mail` are likely served by different servers (whether they belong to the same or different content providers). As another example, `VoIP` has a strong negative correlation with `P2P`. This might be due to frequent use of `VoIP` for business related teleconferencing, often using a laptop and sometimes in a more mobile environment. These endpoints are thus unlikely to run `P2P` applications.

Attractive Relationship. In contrast to the repulsive relationship, by the *attractive* relationship, we mean that given edges from two classes, the presence of one significantly *increases* the chance the other appears among the edges incident on one of the endpoints. For example, `Games` and `P2P` exhibit an attractive relationship. We speculate that endpoints generating online `Games` traffic are likely client desktop machines with high-speed connections, a configuration

also well suited to downloading files through P2P.

In summary, we have demonstrated that mixed TAGs are formed by application-specific TAGs, each of which contains strongly connected clusters (subgraphs), that are linked together in a variety of manners. Some of them exhibit a strong attractive relationship, and thus closer to each other, while others show a repulsive relationship, thus farther apart. Some clusters appear to be independent of each other, but linked together randomly; while others are linked together via a non-random, dependent structure. Next, we first propose a variant of the Markov clustering (MCL) algorithm for decomposing mixed TAGs (Section 3), and then analyze and characterize mixed TAGs at the level of clusters (Section 4).

3. EXTRACTING CLUSTERS WITH MCL

In this section, we propose a variant of the Markov clustering (MCL) algorithm for decomposing mixed TAGs, and then analyze and characterize mixed TAGs at the level of clusters in Section 4.

3.1 MCL Algorithm

The MCL algorithm [13] is developed for graph partitioning, based on the assumption that random walks tend to stay in the same cluster for a longer time rather than traversing across clusters. In order to apply MCL to decompose mixed TAGs, we define the Markovian matrix A as follows. Each row/column represents an edge in the TAG, and an entry a_{ij} is non-zero if edge i and edge j share a common endpoint. Rows in A are normalized so that each a_{ij} corresponds to the probability of a random walk from edge i to j. As we shall see later, this definition enables us to compute a_{ij} from traffic properties so as to emphasize the difference of various application traffic during the clustering process.

MCL iterates two processes: expansion and inflation. Expansion takes the power of the Markovian matrix using the normal matrix product. For instance, taking the square of the matrix will compute random walks of length two. Since higher length paths are more common within clusters than between different clusters, expansion will increase the probabilities of intra-cluster walks. Inflation is the element-wise power to α (we use $\alpha = 1.8$ in our experiment) followed by a diagonal scaling (to make the resulting matrix Markovian). Inflation changes the probabilities associated with the collection of random walks departing from one particular edge by favoring more probable walks. MCL terminates when two processes converge. Cluster memberships can be identified by extracting connected components from the MCL result.

In preliminary experiments, we implemented the *basic MCL* algorithm, where the edge properties are not used and thus $a_{ij} = 1$. In this case, the adjacency matrix of $\mathcal{G}$ can be treated as a symmetric binary matrix. Therefore, direct clustering on the defined TAG implicitly allows host participating in multiple applications. This could enable us to use more scalable hard-clustering algorithms (instead of expensive soft co-clustering algorithms, e.g., [6, 14, 15]) to extract clusters from the graph.

Due to the fact that endpoints may participate in multiple applications, application of the basic MCL to mixed TAGs often lead to polluted clusters, i.e., each cluster consists of a large portion of traffic from non-dominant classes, which is not a satisfactory result in term of decomposing mixed TAGs. However, we can remedy this by augmenting MCL with traffic attributes into the clustering process, given the fact that traffic from different application classes often exhibit distinguishable properties.

Algorithm 1 MCL with edge features.

```
Input: a mixed TAG G, α = 1.8, β = 1;
//Construct weighted adjacency matrix A := {a_ij};
Initialize a_ij = 0 for all entries in A;
for each pair of edges e_i and e_j in G do
    if e_i and e_j share a common endpoint then
        a_ij := exp(−β||x_i − x_j||_2^2);
    end if
end for
repeat
    Normalize rows in A;
    A := A^2; //expansion
    a_ij := a_ij^α, for all entries in A;//inflation
until A converges
Extract connected components in A as the clustering result;
```

Name	Type	Name	Type
min_duration	numeric	max_duration	numeric
min_pkt_size	numeric	max_pkt_size	numeric
min_pkt_rate	numeric	max_pkt_rate	numeric
symmetry	numeric		

Table 3: Edge features derived from basic flow features.

Our basic idea is to assign weights to the Markovian matrix A, and conduct the basic MCL on the weighted matrix. We define the weight between two adjacent edges i and j as $a_{ij} := exp(-\beta||\mathbf{x}_i - \mathbf{x}_j||_2^2)$, where two edges have a higher weight (or a higher probability for a random walk to traverse between e_i and e_j) if they have similar traffic properties. We summarize the augmented MCL algorithm with edge features in Alg. 1.

We consider the worst case scenario while selecting edge features, where transport layer headers are missing, e.g., due to IPv6 tunneling or IPSec. The edge features are listed in Table 3, which are derived from the remaining basic flow features, namely, IP addresses, flow packets, bytes and duration (see [16] for detailed definition of these features).

4. ANALYZING MCL RESULTS

Using the decomposition results from the MCL algorithm, in this section, we study mixed TAGs at the level of clusters.

4.1 Analysis of Clusters

After applying MCL to the 10-minute ISP TAG (with 22,176 edges), we obtain 4,098 clusters. To access the cleanliness of a cluster, we define the *purity* of a cluster as the percentage of edges belonging to the dominant application class in the cluster. We show the cluster size (x-axis) vs. purity (y-axis) for these clusters in Fig. 5. We can see that a majority of the clusters have purity greater than 0.5, and for those large clusters (with more than 100 edges), the purity is more than 0.9. We also see that there are many small clusters. This observation indicates that the proposed MCL algorithm, by incorporating the edge properties, successfully separates the mixed clusters into smaller but clean clusters.

To characterize the shapes of the clusters from different

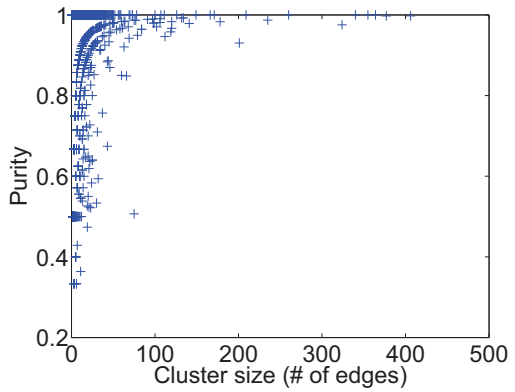


Figure 5: Cluster size vs. purity.

application classes, we use the diameter metric. The diameter of a cluster c is defined as:

$$diameter(c) := argmax_{i,j} SPD(e_i, e_j),$$

i.e., the longest shortest path distance among edges in c. We focus on clusters with more than 10 edges. We see that `Chat`, `FTP`, `Mail` and `Web` clusters tend to have larger diameters (with the median ranges from 3 to 5). In comparison, `Business`, `P2P`, and `Media` clusters have median diameters less than 3. This observation can also be explained by the proposed two block models. For example, strongly connected clusters in the dependent block model are expected to be extracted as a single cluster, and thereby showing a larger diameter. In contrast, blocks in the independent block model form separate clusters, due to their independent nature (only a few links connect these blocks, thereby random walks tend to stay within the block). Hence their diameters are generally small.

4.2 Analysis of Core Clusters

So far our analysis of mixed TAGs has focused primarily on the aspect of graph structures. In this section, we investigate the semantics behind the formation of mixed TAGs, i.e., what are the core clusters and how do they contribute to the global connectivity in different mixed TAGs.

ID	10-minute ISP TAG	10-minute U TAG
1	P2P	Yahoo, Level3
2	Yahoo, PhotoBucket	P2P
3	LLNW	Yahoo Mail, HotMail, MSN
4	McAfee	CareerCast, HotJobs
5	Google, Akamai	DoubleClick, LLNW
	Microsoft, Live	AOL, ImageShack
6	Yahoo Mail	MktLadder, LawLadder
7	Youtube, LLNW	CNN, MCI
8	P2P	Yahoo, Amazon
9	AOL, DoubleClick	Google
10	EBay, Amazon	ChicagoSunTimes, Bullz-Eye

Table 4: Top core clusters ranked by closeness.

To identify core clusters, we rank all the clusters from a mixed TAG using the *closeness* metric, which is defined as:

$$closeness(c) := \frac{1}{\sum_j SPD(c, c_j)}.$$

A high closeness value means the corresponding cluster is located more centrally in a mixed TAG, and hence is more important to the global connectivity. Table 4 lists the top 10 core clusters (ranked by closeness) from the 10-minute ISP TAG and the 10-minute U TAG.

Though two TAGs are from different geolocations with a time gap of 3 years, we see that these two TAGs share many core clusters. These clusters include popular `Web/Mail` servers (Yahoo/Yahoo Mail, Google, AOL), E-commerce websites (Ebay, Amazon) and photo sharing websites (ImageShack, PhotoBucket). Internet users generally have a high chance of visiting these servers, and hence they become centers of the TAGs. In addition, clusters consisting of CDN and advertising servers, e.g., LLNW, Akamai and DoubleClick, also appear in the centers of TAGs. This is not surprising, because users accessing other websites often obliviously connect these servers to retrieve data content, making them more central compared with the other clusters.

Despite these similarities, these two TAGs contain core clusters that are location/time specific. For example, in the U dataset, we observe clusters related to news websites (ChicagoSunTimes, Bullz-Eye, CNN) and job hunting websites (CareerCast, HotJobs, MktLadder, LawLadder). In comparison, McAfee and Youtube show up as central clusters in the ISP dataset (note that the U dataset was collected in 2006, and Youtube was not as popular at that time).

4.3 Application of Traffic Classification

Based on the observation that clusters extracted by the MCL algorithm have generally high purity, we can utilize the MCL results to conduct traffic classification in a semi-supervised setting. The basic idea is to decompose a TAG into clusters using MCL, and then manually classify individual clusters. Recall that the accuracy (or purity) of a cluster is defined as the proportion of edges from the dominant application class in the cluster. The rest of the non-dominant edges in a cluster are treated as classification errors. Therefore, the accuracy of a particular application class is defined as one minus the proportion of erroneous edges from that class. We show the traffic classification accuracy of the 10-minute and the 20-minute ISP datasets in Fig. 6. We see that the overall accuracy is around 97% for both datasets. In terms of the per-class accuracy, the two largest classes, `Web` and `P2P`, have an accuracy of 98%. Even for classes with less than 50 edges, such as `FTP`, `Games` and `VoIP`, we obtain an accuracy of more than 50%. Such accuracy is quite impressive given the best reported accuracy in a supervised setting using basic flow features is only 90% [16].

We argue that our method is not an ultimate solution for the traffic classification problem, but rather a way to "compress" the data before other sophisticated classification methods can be applied. For example, in the 10-minute ISP dataset, instead of classifying 22K edges, we can simply classify 4K clusters by randomly selecting an edge from each cluster to determine the labels for other edges in the same cluster. In this way, our method can provide a compression rate of 20% (4K/22K). This result can be further improved as follows. Based on the observation that each pair of endpoints is only involved in a single type of application over a long duration (e.g., a day), we can use the clusters from other time periods to calibrate the current clustering results. For example, if edges from two separate clusters are observed to be within the same cluster in other time slots, we merge these two clusters. In our experiment, using a whole

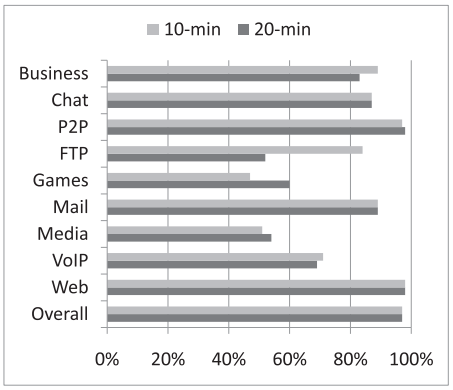


Figure 6: Accuracy.

day data (144 time periods), we can improve the compression rate to 8:1 by only increasing the error rate from 4.8% to 5.6%. As a salient feature, our method still works even when a portion of the edges cannot be classified by traditional methods based solely on traffic features. For example, we find that 9% of the `Web` edges in our datasets do not use port 80 or 443 as service ports, thus are typically misclassified by port-based methods. However, our method can classify them accurately based on their neighboring edges in the same clusters.

5. RELATED WORK

Our work is motivated by the analysis and modeling of application-specific traffic activity graphs [6–8, 17]. Our focus in this paper is primarily on explaining the formation of mixed TAGs as a combination of various application-specific TAGs. Our work is also related to BLINC [1]. However, BLINC emphasizes more on the description of various types of network activities, not the formation of traffic graphs. From the perspective of graph decomposition, various algorithms have been proposed [13–15, 18]. Our MCL method differs in that we take into account the traffic properties associated with individual edges to obtain unpolluted subgraphs. In terms of traffic classification, our method differs from other techniques [2, 16, 19–22] in that we use a semi-supervised learning approach and the clusters are formed by both the traffic properties and edge interaction patterns.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we studied the formation of mixed traffic activity graphs (mixed TAGs). A mixed TAG comprises a number of densely connected subgraphs or clusters from various application classes, which are either dependent or independent. We found that different application classes exhibited repulsive or attractive relationships when they interconnect to form a mixed TAG. We also proposed a variant of the Markov clustering algorithm to decompose mixed TAGs. Analysis on these resulting clusters showed that though mixed TAGs display similar structures, some core components in these TAGs are time/location dependent. Our work in the future will concentrate on mathematical modeling of the generation process of mixed TAGs. In addition, we will develop more effective way of classifying network traffic by combining both network traffic statistics and network structure information.

Acknowledgement

Zhi-Li Zhang is supported in part by the NSF grants CNS-0905037 and CNS-1017647, and an AT&T VURI gift grant.

7. REFERENCES

[1] T. Karagiannis, K. Papagiannaki and M. Faloutsos. BLINC: Multilevel traffic classification in the dark. In *Proc. of ACM SIGCOMM'05*, August 2005.

[2] A. Moore and D. Zuev. Internet traffic classification using bayesian analysis. In *Proc. of Sigmetrics'05*, 2005.

[3] A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. In *Proc. of SIGCOMM '04*, pages 219–230, 2004.

[4] A. Lakhina, M. Crovella, and C. Diot. Mining anomalies using traffic feature distributions. In *Proc. of SIGCOMM '05*, pages 217–228, 2005.

[5] K. Xu, Z.-L. Zhang and S. Bhattacharyya. Profiling Internet backbone traffic: behavior models and applications. In *Proc. of ACM SIGCOMM'05*, 2005.

[6] Y. Jin, E. Sharafuddin, and Z-L. Zhang. Unveiling core network-wide communication patterns through application traffic activity graph decomposition. In *Proc. of SIGMETRICS '09*, pages 49–60, 2009.

[7] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese. Network monitoring using traffic dispersion graphs (tdgs). In *Proc. of IMC'07*, 2007.

[8] M. Iliofotou, M. Faloutsos, and M. Mitzenmacher. Exploiting dynamicity in graph-based traffic analysis: techniques and applications. In *Proc. of CoNext'09*, 2009.

[9] P. Haffner, S. Sen, O. Spatscheck, and D. Wang. ACAS: Automated construction of application signatures. In *Proc. of MineNet*, 2005.

[10] Port numbers, http://www.iana.org/assignments/port-numbers.

[11] Graphviz. http://www.graphviz.org/.

[12] P. Bickel and A. Chen. A nonparametric view of network models and newman-girvan and other modularities. *Proc. of National Academy of Sciences*, 106(50):21068–73, 2009.

[13] S. Dongen. *Graph clustering by flow simulation*. PhD thesis, University of Utrecht, 2000.

[14] H. Shan and A. Banerjee. Bayesian co-clustering. In *ICDM '08*, pages 530–539, 2008.

[15] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *J. Mach. Learn. Res.*, 8:1919–1986, 2007.

[16] Y. Jin, N. Duffield, P. Haffner, S. Sen, and Z.-L. Zhang. Inferring applications at the network layer using collective traffic statistics. In *Proc. of ITC 22nd International Teletraffic Congress*, 2010.

[17] M. Iliofotou, B. Gallagher, T. Eliassi-Rad, G. Xie, and M. Faloutsos. Profiling-by-association: a resilient traffic profiling solution for the internet backbone. In *Proceedings of the 6th International COnference*, Co-NEXT '10, pages 2:1–2:12, 2010.

[18] I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. of*

KDD'01, 2001.

[19] J. Erman, A. Mahanti, M. F. Arlitt, I. Cohen, and C. L. Williamson. Offline/Realtime traffic classification using semi-supervised learning. *Perform. Eval.*, 64(9–12):1194–1213, 2007.

[20] H. Jiang, A. W. Moore, Z. Ge, S. Jin, and J. Wang. Lightweight application classification for network management. In *Proc. of INM '07*, 2007.

[21] Andrew W. Moore and Denis Zuev. Internet traffic classification using bayesian analysis techniques. In *Proc. of ACM SIGMETRICS'05*, Banff, Canada, 2005.

[22] B. Gallagher, M. Iliofotou, T. Eliassi-Rad, and M. Faloutsos. Link homophily in the application layer and its usage in traffic classification. In *Proceedings of the 29th conference on Information communications*, INFOCOM'10, pages 221–225, 2010.