The Pennsylvania State University
The Graduate School

NETWORK TRAFFIC ANALYSIS: ANOMALY DETECTION AND SOME IMPLICATIONS OF NEUTRALITY

A Dissertation in
Electrical Engineering
by
Fatih Kocak

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2014

The dissertation of Fatih Kocak was approved* by the following:

David J. Miller
Professor of Electrical Engineering
Dissertation Co-Advisor, Co-Chair of Committee

George Kesidis
Professor of Electrical Engineering, and Computer Science and Engineering
Dissertation Co-Advisor, Co-Chair of Committee

Kenneth Jenkins
Professor of Electrical Engineering

John Doherty
Professor of Electrical Engineering

Anna Squicciarini
Assistant Professor of College of Information Sciences and Technology

Kultegin Aydin
Professor of Electrical Engineering
Head of the Department of Electrical Engineering

*Signatures on file in the Graduate School.

Abstract

This thesis makes contributions to two separate topic areas, namely anomaly detection and network neutrality areas, which are related to each other. In the first part, we focus on detecting samples from anomalous latent classes, buried within a collected batch of known (normal) class samples, where the number of features for each sample is high. We assume and observe to be true that careful feature selection within unsupervised anomaly detection may be needed to achieve the most accurate results (depending on the particular feature representation that is in use). We form pairwise feature tests based on Gaussian mixture models, with one test for every pair of features. The mixtures are estimated using known class samples (null training set). Using these mixture models, p-values are obtained on the test batch samples under the null hypothesis. We use these p-values in basically two different ways. In our first approach, we consider sample-by-sample detection of anomalous class samples amongst the batch of collected samples. We propose a novel sample-wise sequential anomaly detection procedure with growing number of tests. New tests are included only when they are needed, *i.e.*, when their use on currently undetected samples will yield greater aggregate statistical significance of multiple testing corrected detections than obtainable using the existing test set. This approach aims to maximize aggregate statistical significance of all detections made up until a finite horizon. We then approach this anomaly detection problem as a clustering problem. We calculate approximate joint p-values for candidate anomalous clusters, defined by (sample subset, test subset) pairs. Our approach sequentially detects the most significant clusters of samples in a networking context. We use different kinds of feature representations and conditioning contexts and experimented on many datasets for comprehensive performance evaluation purposes. Our p-value clustering algorithm is compared, using ROC curves, with alternative p-value based methods, our sample-by-sample sequential detection, and the one-class SVM. All the competing methods make sample-wise detections, *i.e.*, they do not jointly detect anomalous clusters. The anomalous class was either an HTTP bot (Zeus) or peer-to-peer (P2P) traffic. For certain feature representations, our p-value clustering approach gives promising results for detecting the Zeus bot and P2P traffic amongst Web.

In the second part, we analyze some issues about the network neutrality. We investigate the relations between caching, pricing, and revenues of entities under the light of network neutrality concerns. Firstly, we consider a model with two "eyeball" Internet Service Providers (ISPs) (*i.e.*, those acting as *both* network access and content providers (CP)), with transit pricing of net traffic at their peering point. That is, there is an inter-provider service-level agreement (SLA) involving a revenue based on net transit traffic flow across their peering point(s). We studied the effects of caching remote content via a game between the ISPs on a platform having usage-priced subscribers. We do this for two cases: one is for different congestion points in each ISP (depending traffic origin) leading to tractable Nash equilibria; and the other is for a single congestion point which we herein study numerically. Secondly, we consider a game between an ISP and CP on a platform of end-user demand. A price-convex demand-response is motivated based on the delay-sensitive applications that are expected to be subjected to the assumed usage-priced priority service over best-effort service. Thus, we are considering a two-sided market with multiclass demand wherein one class (that under consideration herein) is delay-sensitive. Both the Internet and proposed Information Centric Network (ICN, encompassing Content Centric Networking (CCN)) scenarios are considered. For our purposes, the ICN case is basically different in the polarity of the side-payment (from ISP to CP in an ICN) and, more importantly here, in that content caching by the ISP is incentivized. A price-convex demand-response model is extended to account for content caching. The corresponding Nash equilibria are derived and studied numerically.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my sincerest gratitude to my thesis advisors Dr. George Kesidis and Dr. David Miller, who were very helpful at every stage of my studies.

I would also like to thank my parents and my friends who were supportive during this long journey.

Finally, I acknowledge National Science Foundation (NSF) for supporting me under grants no. 0915552 and 1116626. I would also like to acknowledge Cisco for supporting me via a Cisco Systems URP Gift.

Dedication

To my parents and my sister

Chapter 1

Introduction

This thesis makes contributions to two separate areas, namely anomaly detection and network neutrality. Although our work in anomaly detection is applicable to many domains, we are particularly interested in application to network intrusion detection problems which aim to detect anomalous behavior among network flows. Our contributions in network neutrality are also related with network flows, since we investigate the effects of caching on the pricing of network flows and revenues for different network models.

Let us elaborate more on how anomaly detection and network neutrality are related to each other. Suppose that there is anomalous traffic originating from particular end-users in a network. Due to security concerns, this may enforce ISPs to take precautions (such as throttling, blocking, etc.) against those users, even though they may not be aware that their device is infected and included in a malicious network such as a botnet. This discrimination against some users will result in violation of network neutrality principles. Furthermore, inclusion in a malicious network may lead to a dramatic increase in the traffic for certain users. This may lead to extra costs for those users, especially when pricing is not flat-rate. This means that the users that are receiving same service may not be priced similarly, since the users that are infected and included in botnets will be consuming more bandwidth than the uninfected users. This is consequence of botnets that violates net neutrality, which underlines the importance of detecting anomalies for the sake of keeping the network neutral.

1.1 Anomaly Detection

Anomaly detection has great practical significance, manifesting in a wide variety of application domains including detection of suspicious/anomalous events in Internet traffic, in human behavior, host-based computer intrusion detection, detection of equipment or complex system failures, as well as of anomalous measurements in scientific experiments. The scenario addressed in this thesis is detection of anomalies of an unknown anomalous class, amongst the N samples in a collected data batch, $\mathcal{X} = \{\underline{x}_i, i = 1, \ldots, N, \underline{x}_i \in \mathcal{R}^D\}$. The batch may consist, *e.g.*, of samples collected over a fixed time window. We assume there is a separate database exclusively containing "normal" examples that can potentially be leveraged for learning the null hypothesis probability model (either on the full D-dimensional space or on lower-dimensional subspaces), used to assess statistical significance of detected anomalies. These statistical significance values are quantified by p-values, which mean the probability of making an observation more extreme than a given observation, under an assumed probability law. For instance, suppose that the assumed probability model is Gaussian with mean 30 and variance 9. Then, the p-value[1] of a sample having value 10 (or 50) can be found by calculating the area shown on Figure 1.1 (red regions on the figure). Notice that the samples having the same distance to the mean have the same p-value, i.e. they are the same in statistical significance for this probability model (10 and 50 in this example). The distance notion will be detailed in Chapter 2.

There are several reasons why D may be large. First, some applications are inherently high-dimensional, with many (raw) features. Second, large D may enable greater anomaly detection power. In supervised classification, it may be possible to discriminate known classes using a small number of (judiciously chosen) features that have good (collective) discrimination power. However, anomaly detection is inherently unsupervised (with respect to the anomalies) – there are generally no anomalous examples and no prior knowledge on which subset of raw (and/or derived) features may best elicit anomalies. This suggests use of more features may increase the likelihood that a sample will manifest a detectable effect. In our experiments, we will consider malicious network traffic packet flows which mimic Web application flows to evade detection – considering more rather than fewer features may typically be required to detect "evasive" anomalies.

There are multiple anomaly detection strategies that can be applied:

[1]This is called two-sided p-value. One can also calculate and use the area under one of the 2 red regions, which would be called one-sided p-value. But, the two-sided p-value notion in 1-D shown in Figure 1.1 gives an idea on how we calculate the p-values using the models with higher dimensions.

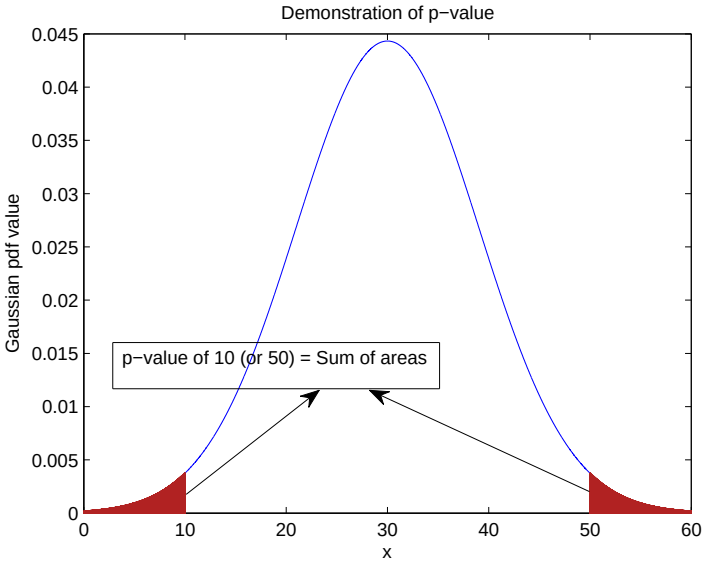

Figure 1.1. Calculation of p-value of 10 (or 50) for $\mathcal{N}(9, 30)$

1) Applying a single test, based on the joint density function defined on the full D-dimensional feature space.

2) Applying multiple tests, *e.g.*, tests on all pairwise feature densities.

   (a) Detecting the sample yielding the smallest p-value over all these tests as the anomaly with the highest priority.

   (b) Detecting the samples according to (some type of) average p-value over all of the tests.

3) Using outlier detection, *e.g.*, one-class SVMs [109].

There are 2 problems with 1). Firstly, if D is large relative to N, the estimation of the joint density will be inaccurate (*i.e.*, there is a curse of dimensionality) [33]. Secondly, suppose that the features are statistically independent and that the anomaly only manifests in one (or a small number) of the features. In this case, the joint log-likelihood is the sum of the marginal (single feature) log-likelihoods, and the effect of a single (anomalous) feature on the joint log-likelihood (which amounts to an average of the marginal log-likelihoods) diminishes with increasing D.

There are also problems with 2). There is the complexity associated with using a number of tests combinatoric (*e.g.*, quadratic) in D. Ignoring complexity, in 2)(a), use of many tests may unduly increase the number of false alarms – suppose there is a single

anomaly in the batch, with the anomaly detectable by only one of the $K = \frac{D(D-1)}{2}$ pairwise tests, with p-value p. Assuming that the tests are independent, the probability that no other sample will have a smaller p-value (and will thus be falsely detected first, prior to detecting the anomaly), given p, is $(1-p)^{K(N-1)}$, *i.e.*, it is exponentially decreasing in KN. Supposing $p = 10^{-5}$, this probability is ~ 0.9 for $KN = 10^4$, and it is vanishing by $KN = 10^6$. In fact, possibility of missing the anomalous behavior that manifests itself in only certain features (tests) exists in all of the methods mentioned above. In 2)(b) this is caused by averaging over all tests, and in 3) by usage of all of the features in one-class SVM.

Here we propose alternatives to these approaches for anomaly detection, inspired by greedy feature selection techniques commonly used in supervised learning. For supervised classification, it is well-known *e.g.* [104] that even if all features have some discrimination power, use of all features may in fact degrade classification accuracy unless there is sufficient training data for learning model parameters accurately enough to exploit this discrimination power. The fact that training data may be limited relative to D motivates feature selection and also *e.g.* decision trees, which may base decisions on only a small complement of the full set of measured features. Also, for unsupervised clustering in high dimensions, feature selection, embedded within the clustering process, has been demonstrated to be crucial for reliable clustering and for accurate estimation of the number of clusters [42]. But, it should be kept in mind that, since the AD problem is unsupervised, it is a priori unknown which subset of features may be informative and there are no ground-truth labeled anomalous class examples to guide selection of these features. Thus, the "feature selection" problem for anomaly detection is much more challenging than in the supervised case.

Conceptually allied to these approaches, but uncommon in an anomaly detection setting, we propose 2 novel approaches for anomaly detection in a batch that are spartan in their use of features/tests.

The first approach is a sample-wise sequential anomaly detection approach, in which new tests are (greedily) included only when they are needed, *i.e.* when their use (on the remaining batch) will yield more statistically significant detections (lower p-values (corrected for multiple testing [19])) than those obtainable using the existing set of tests. More generally, this approach seeks to maximize the aggregate statistical significance of all detections up until a finite horizon. Our approach may be particularly suitable when there is a latent anomalous class present in the data batch, discriminable from the known class using an (albeit unknown) small subspace of the full feature space.

There is prior work, somewhat related, in the statistics literature on sequential thresholding of p-values to ensure a target family-wise error rate [48] or false discovery rate [62] is achieved. However, such schemes do not alter the order in which anomalies are detected – they only determine when to stop making detections. These works also do not sequentially grow the number of tests, in interrogating the batch. There is also work on designing classifiers to maximize the area under the ROC curve, rather than to minimize the classifier's error rate [116]. However, that work addresses a supervised learning scenario, with all classes known a priori and with labeled training exemplars provided for each class. By contrast, we optimize an estimated ROC curve associated with the anomaly detection problem, for which there are no labeled (anomalous) exemplars. There is substantial prior literature on anomaly detection for network intrusion detection, *e.g.*, [23], based on numerous proposed statistical tests and heuristic criteria. Most such approaches will only be effective in detecting specific types of anomalies, within particular networking domains. Our algorithms may provide a robust mechanism for identifying the most suitable such tests to use, adaptive to the networking domain, to the particular anomalies/attacks that may be present, and to temporally changing (nominal) network traffic statistics.

The second approach regards this problem as a clustering problem. Similar to the sample-wise detection approach, we conjecture that (and assess whether) benefits may be achieved by feature selection in an anomaly detection (AD) setting (which is known to be true in supervised setting), *i.e.*, we conjecture that use of many features/tests (most of which have little power to reveal the anomalous class) may mask anomalous classes/clusters that could be well-revealed using only a few (highly discriminating) tests. Many existing AD methods (as well as our sample-wise detection approach) only make separate anomaly detection decisions for each individual sample, *i.e.*, they do not jointly detect clusters of anomalies. In this thesis, however, we make joint detections of clusters of samples, using as the detection criterion an approximate joint p-value. Candidate clusters are jointly defined by their sample subset and the subset of features (tests) with respect to which the cluster exhibits its most extreme deviation from the null – *i.e.*, built into our detection criterion is some intrinsic impetus for feature selection.

In Chapter 2, we formulate the calculation of p-values of samples. P-values are calculated by using GMM models, which are modeled for each feature pair. In this chapter, also the experimental procedure is explained. Although the methods proposed and used in this thesis are applicable to any domain, we experimented on the network intrusion detection domain. Particularly, we aimed to find anomalies disguised amongst Web

(HTTP) traffic. So, the known class is Web traffic. The role of the latent unknown class is played by HTTP bot (Zeus) traffic or peer-to-peer (P2P) traffic. How these flows are obtained is explained. Multiple feature representations are described in this chapter, which will be used in the experiments. The extraction and usage of the features are explained in Chapter 2. Their advantages and disadvantages are discussed. The experimental results for our sample-wise and cluster-wise approaches to anomaly detection are provided in Chapters 3 and 4, respectively.

In Chapter 3, the sample-wise sequential anomaly detection approach is explained by providing the mathematical objective and several sequential detection algorithms are proposed for (approximately) optimizing this objective. Our approach is compared, in area under the ROC curve, with several standard detection strategies for a network intrusion domain, detecting Zeus bot intrusion flows embedded amongst (normal) Web flows. We have also demonstrated the importance of careful feature representation, for supervised discrimination of the Zeus bot from Web traffic. These approaches and the related experimental study also take place in [73].

In Chapter 4, the algorithm that uses p-values for clustering is explained. The comparison results with our sample-wise approach and other methods are provided. Experimental results are provided for several different feature representations. Zeus or P2P traffic is used as anomalous traffic in the experiments. A few of these experimental results (using only the alternating feature representation) are reported in [57]. Extensive analysis including different conditioning contexts are reported in [56].

Contributions:

- We propose two types of approaches that make use of feature selection in different ways both of which basically use tests constructed from each feature pair and assess statistical significance by using p-values.
  - Our first approach in Chapter 3 makes sample-wise detections using growing number of tests by making multiple test corrections for existing and unused tests.
  - Our second approach in Chapter 4 makes cluster detections by using a small subset of tests for each cluster detection.
- We propose multiple types of feature representations, including different conditioning contexts.
- We observe that (test) feature selection is effective for anomaly detection in certain feature spaces.

- We experiment on Zeus-Web and P2P-Web separation problems using Lawrence Berkeley National Laboratory (LBNL) datasets.
- We compare and contrast the performance of our algorithms against methods that do not use feature selection. We provide comparisons in terms of area under ROC and early detection successes of the algorithms.
- We observe the performance differences on many datasets with different sizes and characteristics.
- We investigate the effect of order increase in both our sample-wise and cluster-wise anomaly detection approaches that we have proposed (where the meaning of "order" is different in each approach).

1.2 Network Neutrality

The continuing network (net) neutrality debate (*e.g.*, [107, 79, 75, 13, 113, 45]) involves several different entities, such as ISPs[2], Content Providers (CPs), users, and governments (including partnerships). Although there are many different perceptions for the definition and the coverage of net neutrality, one succinct definition is provided in [45]: "[net neutrality] usually means broadband service providers charge consumers only once for Internet access, do not favor one content provider over another, and do not charge content providers for sending information over broadband lines to end users."

A communication network is "neutral" if it is both application neutral and does not require side-payments for use by remote content providers. Application neutrality means that the network does not handle packet-traffic differently based on the application type, e.g., third party streaming video from Netflix is handled the same as the Internet Service Provider (ISP)'s own "managed" streaming video service over commodity IP. Note that application neutrality allows discrimination based on traffic volume and end-user specified priorities. So, differentiated services (diffserv) among application types is "neutral" if requested by end-users, whereas application diffserv implemented unilaterally by an ISP is not application neutral (even if for altruistic purposes, *e.g.*, to give more bandwidth for putative real-time applications). The focus of the following preliminary study is on premium-access bandwidth, not the content or services delivered. An example of a side payment is Neflix paying remote ISPs for access to their subscribers, rather than only paying its own (local) access provider (Level 3).

[2]Equivalently, network service providers or access providers.

Content providers, such as Amazon, Google, Yahoo!, and eBay, typically support net neutrality because under non-neutral conditions they expect additional access-networking expenses and additional limitations or exclusions on their access to their customers [39]. In contrast to CPs, ISPs (particularly residential ISPs) such as AT&T, Verizon, Comcast, and Deutsche Telekom, typically believe that neutrality regulations threaten the profitability of their enormous infrastructure investments and maintenance costs [39, 107], and that CPs do not pay a fair share of these costs while profiting from advertising that is arguably not requested by consumers[3]. Also, flat-rate pricing frameworks leading to "all-you-can-eat" consumer behavior result in high transport costs and congestion in the ISPs' access networks, *e.g.*, [4], which makes ISPs complain about this and leads them to take blocking (*e.g.*, Comcast blocking P2P applications [1]) or pricing (*e.g.*, [93]) measures. It has been argued that some of these problems can be compensated by side payments between CPs and ISPs [21, 8, 9, 115, 76]. Alternatively, the introduction of premium service classes for applications has been suggested for: critical applications such as health monitoring and home security (which are being increasingly used [45]); streamed spectacle events such as sports activities or newly released movies [39]; and interactive real-time video-conferencing/video-phone sessions. Applications engages in premium services will obviously receive a higher Quality of Service (QoS) than applications under best-effort network-access service, and will need to pay usage-based costs (perhaps after a quota). Such payments are (content/application) neutral in nature [27] due to the willingness of the users to pay for the premium content [21]. Under net neutrality with flat-rate priced access[4], ISPs may not have the incentive to improve their existing infrastructure by increasing capacity [24] (particularly the router/switch infrastructure to drive fiber-to-the-home (FTTH)) or by improved security measures such as virus and spam filtering [39]. (Note that such usage-based costs may need to be authenticated to the human subscriber/end-user.)

Regarding quality-of-service management, the physical location of requested content is obviously important to the goal of decreasing delay experienced by the users [41]. This in turn underscores the importance of caching data proximal to the users, including by their ISP. Some large content providers, such as eBay and Google, cache their content around the world on their own servers, while smaller content providers often use intermediary content distributors, such as Akamai, who have caching agreements with local ISPs at different locations [39]. Such agreements or more dedicated partnerships

[3]Note that neutrality regulations for wireless access in the United States have not been instituted at the time of writing of this thesis; see [4] for discussions of the mobile wireless access scenario.

[4]Limited only by uplink and downlink bandwidth of the service agreement.

between ISPs and CPs (*i.e.*, "eyeball" ISPs) lead to scenarios wherein ISPs may cache each other's content, which raises issues of transit pricing between them. See Figure 1.2. To achieve end-to-end QoS, [18] argued that a sending ISP should pay for the transport traffic over an interconnection between ISPs.

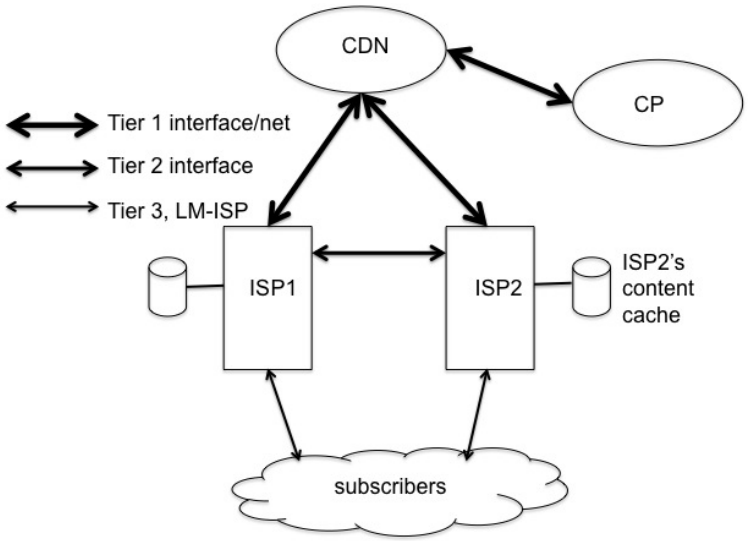

Figure 1.2. A CP may use a Content Distribution Network (CDN) as depicted, or may have a local caching agreement with a last-mile (LM) ISP, or neither

Notwithstanding arguments for and against side-payments, the necessity of providing a single interface (single contract including mutual services) to the end user is emphasized by several presenters in [4]. Product offers to the end users are assumed to be made in mainly two different ways: pull (on-demand) or push. Product offers can be prepared in distributed (among ISPs), partially centralized (by any of the ISPs), or fully centralized (by an external single facilitator entity) ways [32]. We herein primarily consider the "pull" demand model for content product where content requested in the recent past is cached in anticipation of similar demand locally.

In Chapter 6, we consider a model involving two different eyeball ISPs connected at peering point(s) where revenue is generated corresponding to net traffic transmitted [53]. Initially, we consider a crude caching model captured by a single parameter, Φ, affecting the revenue generated by transit traffic. For a more dynamic caching strategy, we model user/customer migration among such ISPs (as in [21]) due to delay-dissatisfaction, and so develop a game where the caching factors Φ are control variables, rather than fixed parameters, of the players of the game.

Since the onset of the net neutrality debate, researchers have studied parsimonious

models of the Internet marketplace to gain insight into the macroeconomic forces in play. Performance is often assessed based on the Bertrand-Nash equilibria of noncooperative, decentralized games, and in terms of dynamical convergence to these equilibria, often considering limited resources (particularly bandwidth [110]) as in classical Cournot games.

For example, games involving end-users and content providers on an ISP platform were studied in [79, 75]. Shapley values, indicating fair division of revenue with a coalition (or cooperative game), are used to argue for side-payments between ISPs and CPs in [68, 69].

In Chapter 7 of this thesis, we consider a noncooperative game between a single (or cooperating collective) CP and a single ISP on a platform of end-users served by both, *i.e.*, a two-sided market . We assume that the applications under consideration are delay-sensitive. Applications only ever requiring best-effort service, and the revenue they generate for the ISP and CP players, are not considered herein.

In Chapter 7, in addition to the current Internet setting, we are also interested in that of proposed Information-Centric Networks (ICNs, generalizing Content-Centric Networks (CCNs)), *e.g.*, [40, 103, 84]. In a related discussion, different scenarios for transit networks and content distribution networks (CDNs) [5, 71] were considered in [6], including those in which the CDN (or individual CP) is incented to compensate the transit network (ISP) to cache its content. In this thesis, the principle difference between the Internet and ICN settings is the direction of the side-payment between ISP and CP, similar to the difference between content-centric and access-centric networking as described in [103] (see their Figures 7.5 and 7.6). Also, the ISP is incentivized to cache in the ICN setting [54, 55].

Contributions:

- We studied the effects of caching remote content to access pricing and revenues under different network models.
  - In Chapter 6, we considered a game between eyeball ISPs with transit pricing of network traffic at their peering point. Also, the scenario where multiple ISPs are competing for the same group of end-users is also among the cases that are investigated.
  - In Chapter 7, we studied a game between an ISP and a CP, considering Internet and ICN scenarios.
- We found the Nash equilibrium points in both of the network models.

- In Chapter 7, we showed how a fractional caching factor could be optimal at Nash equilibrium.
- In Chapter 7, we also observed cases where optimal caching factors take values 0 or 1.
- We compared utilities in Internet and ICN settings in Chapter 7. Caching incentives and direction of the side-payment are the important details that differ in these two scenarios.
- We investigated how caching incentives for ISPs are beneficial for utilities in Chapter 7.
- In Chapter 6, we analyzed effects of content caching in a network of 2 ISPs for two cases.
  - Different congestion points in each ISP, leading to tractable Nash equilibrium analyses.
  - Single congestion point in each ISP, which is studied numerically in this thesis. This imposed a throughput limit downstream to the end-users.
  - Different congestion points in each ISP, where for there are multiple ISPs competing for a group of end-users.

Chapter 2

Background – P-value Calculations, Experimental Setup, and Feature Representations for Anomaly Detection

In this chapter, the background to understand Chapters 3 and 4 is provided. The algorithms that we propose in Chapters 3 and 4 use p-values to make statistical significance assessment of the samples. Tests are constructed from feature pairs. And p-values are calculated for each of these tests. Bivariate or univariate GMM modeling is used, depending on the existence of any categorical features. Starting with a brief discussion on supervised and unsupervised learning concepts, all of these stages, including experimental setup, feature extraction, and different feature representations are explained in this chapter.

2.1 Anomaly Detection

An anomaly can be defined at a high level as a pattern that does not conform to the expected behavior, which is considered as "normal". Anomalous behavior can occur in different forms depending on its nature. There can be point anomalies, where individual data instances are considered anomalous [23]. Another form can be multiple anomalies originating from the same source, which are similarly behaving to each other (in some sense), although their collective behavior is not considered normal.

Anomaly detection has great importance in many application domains, *e.g.*, network traffic, image processing, sensor networks, even in text where sometimes there may be interest in finding the novelties in a given context [23].

The output of anomaly detection can be scores or labels on the test data samples. With some sort of scoring system, one can understand how anomalous a given sample is. Another option is making a hard decision and labeling the samples as normal or anomalous. [23]

Supervised anomaly detection techniques can be used, but they suffer from a fundamental limitation. This issue is due to the basic difference between supervised and unsupervised learning. In supervised learning, the training set has labels for each sample. For example, if one aims to train a supervised classification algorithm, then the class that each training sample belongs is known. On the contrary, in unsupervised learning, labels for the training set samples are not known. This in general makes unsupervised learning much harder, if we compare similar types of problems [33, 15]. But, semi-supervised or unsupervised anomaly detection algorithms have more wide spread applications, since they do not need the existence of the targeted anomalous samples in the training set. In unsupervised anomaly detection, the aim is to detect unknown (new) behavior. Unknown behavior may be nominal (unknown knowns) or attack (unknown unknowns, as typically assumed). Known behavior is typically assumed to include known knowns, but may also include known unknowns (attack) and "natural outliers". The null hypothesis is based on the known behavior. In other words, the training set consists of samples that belong to known behavior. The alternative hypothesis is the unknown behavior, which is our aim to detect herein. No example of alternative hypothesis are present in the training set. So, in this way, an unsupervised anomaly detector will be able to recognize unknowns, which are anomalous or suspicious behavior even without encountering such behavior before. This is an important problem in network intrusion detection: detection of zero-day attacks.

2.1.1 Prior Work

As mentioned before, there are many application areas of anomaly detection. In this thesis, the particular focus is on network intrusion detection, where most of the following literature survey originates.

Depending on the aim of the anomaly detector, there are studies that employ supervised or unsupervised anomaly detection. Again, the basic difference between these two approaches is the availability of anomalous samples in the training phase. In a supervised

approach, anomalous samples are available in the training phase. There are many prior studies on supervised anomaly detection, such as [121], [106], [120], [80], [65], [20], [61], [94] and [91]. Detection systems based on known signatures, such as [83], [89] are widely used in supervised methods.

There are also examples of semi-supervised [118] and hybrid approaches which utilize both unsupervised and supervised techniques in anomaly detection, *e.g.*, in [97] hybrid of one-class SVM (unsupervised) and soft-margin SVM (supervised) is employed. Another such work is [35], where anomaly detection is attempted after signature based detection.

But, as also mentioned in [98], where current problems about network intrusion detection are examined, the aim in network intrusion detection should be to find previously unseen anomalous activity, without the need to define the anomalous activity upfront. (Also, attention is drawn to the fact that there is not enough publicly available recent dataset, which is an obstacle for experimental studies). Another work highlighting the importance of unsupervised anomaly detection is [67], where it is mentioned that the possibility of detecting formerly unknown intrusions makes anomaly detection interesting to attempt. As also mentioned in [74], signature-based "anomaly" detection approaches, which are supervised, are unable to catch the unseen anomalies, since they require availability of samples regarding to the targeted anomalies in the training phase.

Anomalous activity in the networks can be of various forms. A botnet is a widespread anomalous activity source, which consists of a network comprised of compromised machines participating various kinds of malicious behaviors, such as fraud, distributed denial-of-service attacks, etc. P2P activity can be also be viewed as an anomalous behavior when it spoofs Web activity on port 80. There are many works in the literature on network intrusion detection that are about detection of botnet traffic [14] and P2P activity.

But there are different interpretations of anomalies in networks. For instance, [7] targets anomalous time intervals that might be caused by various reasons such as large data transfers and untimely congestion. In [26], traffic volume anomaly detection is performed, where a volume anomaly is defined as an unusually small or large volume of traffic occurring within a time period. [31] is on anomaly detection in high dimensional data under the null hypothesis of no anomaly. Their test statistic is the magnitude of the residuals of a Principal Component Analysis (PCA) analysis. An experimental study targets volume anomalies in Internet traffic data.

There are also works that aim to detect machines where malicious activities take place. In [44], a system called botminer is proposed which exploits behavioral anomalies.

It aims to find the groups of already compromised machines in a network by utilizing the existence of communication among the members of the botnet and analyzing certain malicious activities of bots in the compromised machines. [94] proposes a two step approach for online malware detection, which consists of clustering and detection engines. Malicious operating system objects (e.g. processes and files) are detected in this work. Firstly, they are clustered. Then, the detection engine detects the cluster as malicious if the behaviors of the clusters match a predefined behavior template formed by a set of behaviors. A set of malware software and a set of benign software are used to train the template database (which makes the approach supervised). [108] attempts botnet detection based on DNS query periodicity of bots. It targets to find bot-relevant domain names and IP addresses. Common property of these works is that they utilize protocol behavior anomalies.

There is also a significant research effort in finding anomalous flows in networks. For instance, in [119], which is another work that utilizes protocol behavior anomalies, discrimination of P2P botnet traffic from legitimate P2P traffic is performed by a two-phase system. In the first phase, the P2P clients are detected. Then, flows are clustered by a two-step clustering approach. The first step is K-means clustering where K is number of expected clusters, and the second step is hierarchical clustering. The flow features used here are number of packets and bytes (sent and received). The destination IP addresses of the flows in these clusters are considered. The clusters whose distinct Border Gateway Protocol (BGP) prefix count for these destination IPs is smaller than a selected threshold are discarded. Second phase is the detection of P2P bots. The similarity between active time of the bots and the active time of the underlying compromised system is used. Also, the fact that overlap of peers contacted by two P2P bots belonging to the same P2P botnet is much larger than that contacted by two clients in a legitimate P2P network is used. After clustering, hosts in dense clusters are classified as P2P bots. In the experimental part, they analyze results for different K values (in K-means clustering), but the selection criterion is not mentioned. Another issue about this work is that the specific properties of P2P botnets are utilized in this work, making it harder to apply for other types of botnets and impossible to apply in other domains.

Another approach for detecting anomalous flows is by using flow-based features, which has received interest recently. There are many studies that use network flows to detect botnets [63], [38], including supervised and unsupervised settings. Some of these works use payload information [66], [117], [114]. But, these approaches have drawbacks. One issue is that payload information might be unavailable due to some reason such as

encryption. Also, polymorphic or metamorphic malware can avoid payload-information based detection systems.

Features that can be obtained from the packet headers are not prone to these kinds of problems about payload usage. One such work is [36], where anomaly detection is made among unlabeled data with the assumption that normal samples dominate the dataset. Three methods that are used in this work are a fixed width clustering method based on the distance between points, clustering by using distances to the k nearest neighbors, and one-class SVM. In the experimental part, anomalous flows are detected. Flow features such as total duration and total bytes transferred are used. In this approach, the selection of the slackness (ν) parameter that is crucial for one-class SVM is not provided. This work aims to provide an unsupervised framework, which is preferred in anomaly detection settings as mentioned above. But, good hyperparameter selection in this work seems is challenging, which is very important in especially unsupervised approaches. [85] has a similar approach. Here, using a small part of data to select the best parameter value is proposed. But, this is counter to the unsupervised nature of the approach.

In [60] network wide anomalies are found by aggregating IP-level traffic data into origin-destination flows. The traffic between an origin-destination pair consists of the flows that enter the network at the origin and exit from the destination. The drawback of this approach is that this aggregation loses the flow-level resolution and is only able to have a rough, high-level notion about the anomalies in the network.

Apart from works concerned about anomaly detection applications in open networks, there are also studies targeting specific networks for anomaly detection. One such work is given in [72], where the aim is perform anomaly detection in industrial control system networks, which are deterministic and more behaviorally restricted compared to "open" networks, such as Internet.

There are also outlier detection approaches that are applied to anomaly detection problems. For example, [87] aims to find the outliers by ranking the samples based on the distance of a sample from its kth nearest neighbor and declares the top n samples to be outliers in this ranking. Here, it is assumed that anomalies are rare in the data. [99] uses one-class SVM.

Among the aforementioned works, some of them perform sample-wise detections such as [99], [87], [7]; whereas [36], [119], [94], [44] make cluster-wise detections. But none of them use a statistical significance assessment using p-values. However, in a different domain [82] where ecological adaptation of honey bees are studied, p-values are used to represent the significance of relative differences of on protein levels. P-values are then

clustered by using a hierarchical clustering approach.

2.2 GMM Modeling

Here, how the GMM modeling in this thesis is performed is explained. To give an overview; in order to fit GMMs onto the training set data, firstly, K-means clustering (Section 2.2.1) is run for a given number of components. Then, the result of K-means is used in the initialization step of the EM algorithm (Section 2.2.2). Due to random initialization in K-means clustering, this (running K-means and then EM) is repeated multiple times. Best trial and best component count are picked according to BIC criterion (Section 2.2.3). The details below will clarify this process.

2.2.1 K-means Clustering

K-means clustering [70, 15] is a widely used unsupervised algorithm that aims to minimize the sum of Euclidean distance of each sample to the centroid of the cluster that it belongs to. Aim is to find the values for $\{r_{jl}\}$ and $\{\underline{m}_l\}$ that will minimize

$$\sum_{l=1}^{L}\sum_{j=1}^{V} r_{jl}\|\underline{x}_j - \underline{m}_l\|_2^2 \tag{2.1}$$

where $\underline{x}_j$ denotes the jth sample in a training set of size V, L is the number of clusters, m_l is the center of the lth cluster, and $r_{jl} \in \{0, 1\}$ is an indicator variable taking value 1 if $\underline{x}_j$ belongs to the lth cluster.

The algorithm steps are as follows:

1) Randomly assign the centroids ($\underline{m}_l$) for each cluster.

2) Assign each sample to the cluster whose centroid is closest.

$$r_{jl} = \begin{cases} 1, & \text{if } l = \arg\min_i \|\underline{x}_j - \underline{m}_i\|_2^2 \\ 0, & \text{otherwise.} \end{cases}$$

3) Update cluster centroids:

$$m_l = \frac{\sum_{j=1}^{V} r_{jl}x_j}{\sum_{j=1}^{V} r_{jl}}$$

4) Repeat steps 2) and 3) until the samples no more change the clusters that they belong to.

K-means clustering is guaranteed to converge, although it may converge to a local minimum. (Convergence properties are studied in [70].) The initialization of the cluster centroids plays an important role in this respect. Running the algorithm more than one time helps to avoid unlucky random initialization.

Also, it is worth noting that the samples are hard-assigned to the clusters, which will not be the case in Expectation-Maximization algorithm presented in the next section.

2.2.2 Expectation-Maximization (EM) Algorithm

First, a general overview of the EM algorithm will be seen below. Then, how EM can be used to fit GMM models on a given data will be discussed.

2.2.2.1 General Overview of EM

EM algorithm [29, 15] aims to find the maximum likelihood estimate when latent variables are used in the model. Let $\mathbf{X}$ denote the observed data, $\mathbf{Y}$ the latent variables, and $\mathbf{\Theta}$ the model parameters. Then, the log-likelihood is given by

$$\ln P(\mathbf{X}|\mathbf{\Theta}) = \ln \left\{ \sum_{\mathbf{Y}} P(\mathbf{X}, \mathbf{Y}|\mathbf{\Theta}) \right\}. \tag{2.2}$$

Here, Y being latent makes X incomplete data. (X and Y together are called complete data.) Therefore, (2.2) is in fact called incomplete log-likelihood function. EM algorithm maximizes this iteratively as below:

1) Initialize $\mathbf{\Theta}^{(t)}$.

2) *Expectation (E) Step:*
Find $P(\mathbf{Y}|\mathbf{X}, \mathbf{\Theta^{(t)}})$.

3) *Maximization (M) Step:*

$$\mathbf{\Theta}^{(t+1)} = \arg\max_{\mathbf{\Theta}} \sum_{\mathbf{Y}} P(\mathbf{Y}|\mathbf{X}, \mathbf{\Theta}^{(t)}) \ln P(\mathbf{X}, \mathbf{Y}|\mathbf{\Theta})$$

4) Stop, if convergence happens.
Assign $\mathbf{\Theta}^{(t)} \leftarrow \mathbf{\Theta}^{(t+1)}$ and return to 2), if convergence criterion is not met.

At each cycle, EM is guaranteed to increase the incomplete log-likelihood. Similar to the case in K-means clustering, EM is also not guaranteed to find the global optimum, if there are multiple local optima. In this case, it may converge to one of these local optima. On the other hand, if the log-likelihood function is concave, then EM will converge to the optimum point. Convergence properties of EM algorithm are studied in [112].

2.2.2.2 EM for GMM Modeling

In order to find the Gaussian components that generated the observed data, we use EM algorithm. The latent variables here are the variables that denote the source component generating each sample in the given data set. Let y_{jl} denote these latent variables. $y_{jl} = 1$, if jth sample is generated by the lth Gaussian component and 0 otherwise. The model parameters are means ($\underline{\mu}_l$), covariance matrices (Σ_l), and mixing probabilities (α_l) for each component.

1) Initialize $\underline{\mu}_l$, Σ_l, and α_l for each component and calculate log-likelihood:

$$\sum_{j=1}^{V} \ln \left(\sum_{l=1}^{L} \alpha_l f_{\underline{X}_j|l}(\underline{x}_j|\theta_l) \right) \tag{2.3}$$

2) *Expectation (E) Step:* Find the posterior probabilities for each sample.

$$P(y_{jl} = 1) = \frac{\alpha_l f_{\underline{X}_j|l}(\underline{x}_j|\theta_l)}{\sum_{i=1}^{L} \alpha_i f_{\underline{X}_j|i}(\underline{x}_j|\theta_i)}$$

3) *Maximization (M) Step:* Update the model parameters.

$$\begin{aligned}
\underline{\mu}_l &= \frac{1}{V} \sum_{j=1}^{V} P(y_{jl} = 1) \underline{x}_j \\
\Sigma_l &= \frac{1}{V} \sum_{j=1}^{V} P(y_{jl} = 1)(\underline{x}_j - \underline{\mu}_l)(\underline{x}_j - \underline{\mu}_l)^T \\
\alpha_l &= \frac{1}{V} \sum_{j=1}^{V} P(y_{jl} = 1)
\end{aligned}$$

4) Calculate the log-likelihood

$$\sum_{j=1}^{V} \ln \left(\sum_{l=1}^{L} \alpha_l f_{\underline{X}_j|l}(\underline{x}_j|\theta_l) \right) \tag{2.4}$$

Stop, if convergence criterion is met.

Return to 2), if convergence criterion is not met.

2.2.3 Selecting the Number of Components

The training Web flows (represented by D-dimensional feature vectors) were fit using GMMs (building both a D-dimensional GMM and all D choose 2 bivariate GMM models), with the Bayesian Information Criterion (BIC) [92] used to select the number of components in any given model. The reason in using pairwise feature tests is that this captures the possible dependence in the behavior of each feature pair. In addition to this, it also keeps the training set size requirement low, avoiding curse of dimensionality.

The BIC cost that is tried to minimize in the GMM training step is provided below. The first term uses the number of parameters that are present in the model (d(=2) is the dimension and L is the number of GMM components) and the number of samples in the training set (V). (For each GMM component, there are d parameters in the mean vector and $d(d+1)/2$ different parameters in the covariance matrix. There are $L-1$ number of free model parameters for the priors.) The second term in (2.5) stands for the log-likelihood.

$$\begin{aligned}
\text{BIC Cost} &= \left(L \left(d + \frac{d(d+1)}{2} \right) + L - 1 \right) \frac{\ln(V)}{2} - \ln \left(\prod_{j=1}^{V} f_{\underline{X}_j}(\underline{x}_j) \right) \\
&= \left(L \left(d + \frac{d(d+1)}{2} \right) + L - 1 \right) \frac{\ln(V)}{2} - \ln \left(\prod_{j=1}^{V} \sum_{l=1}^{L} \alpha_l f_{\underline{X}_j|l}(\underline{x}_j|\theta_l) \right) \\
&= \left(L \left(d + \frac{d(d+1)}{2} \right) + L - 1 \right) \frac{\ln(V)}{2} - \sum_{j=1}^{V} \ln \left(\sum_{l=1}^{L} \alpha_l f_{\underline{X}_j|l}(\underline{x}_j|\theta_l) \right)
\end{aligned} \tag{2.5}$$

Figure 2.1 shows a training set for 2 arbitrary features (out of 20). Figure 2.2 depicts the GMM components that are fit onto the training set (Gaussian models are shown with the contours that pass over half of the maximum pdf value for that component) and the test set. (File 1[1] is used here.)

[1]For File numberings, see Table 2.1 in Section 4.3.

Figure 2.1. An example of a training set that is used to fit GMM

2.3 Calculation of P-values of Samples

In this work, reference/null densities for low-dimensional collections of features are modeled by multivariate Gaussian mixture models (GMMs), *i.e.*

$$f_{\underline{V}}(\underline{v}) = \sum_{l=1}^{L} \alpha_l f_{\underline{V}|l}(\underline{v}|\theta_l) \tag{2.6}$$

where α_l ($0 \leq \alpha_l \leq 1$, $\sum_{l=1}^{L} \alpha_l = 1$) is the mass for each component density $f_{\underline{V}|l}(\underline{v}|\theta_l)$, and the parameter set $\theta_l = (\underline{\mu}_l, \Sigma_l)$. We would like to calculate the p-value – the probability that a feature vector will be more extreme than the given observed vector $\underline{x}$. For a single multivariate Gaussian density $\mathcal{N}(\underline{\mu}, \Sigma)$, the corresponding multivariate integral (over the exterior of the ellipse defined by the squared Mahalanobis distance from $\underline{x}$ to $\underline{\mu}$) needs to be calculated. Then, this can be extended to multiple Gaussian components case. The next 2 subsections provide the formulations on how to find the necessary integral and the extension to GMM case (*i.e.*, with multiple Gaussian densities), for univariate and bivariate Gaussian cases, respectively.

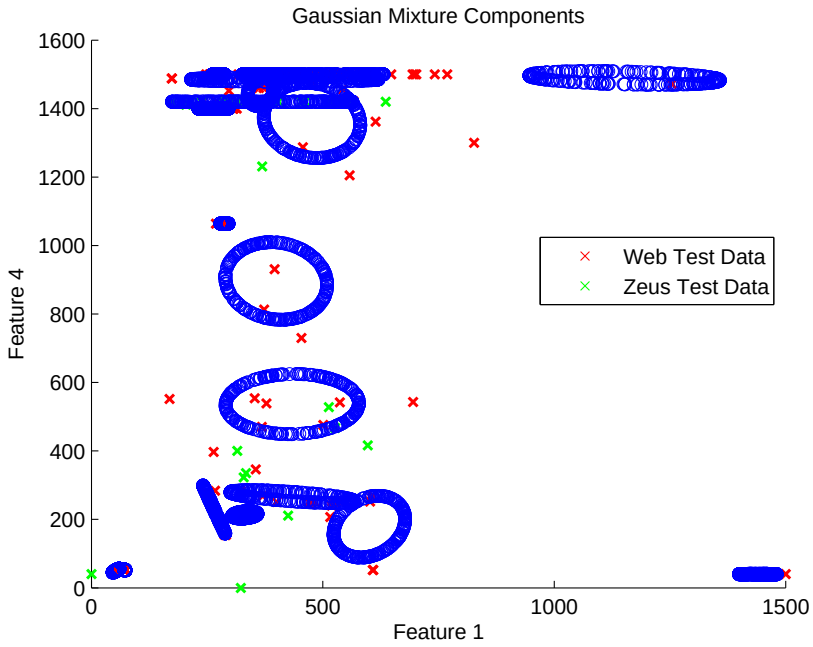

Figure 2.2. GMM components on the test set

2.3.1 Univariate Gaussian Case

For this case, we can replace $\underline{x}$ with x, $\underline{\mu}$ with μ, and Σ with σ^2, since we are dealing with 1-dimensional vectors. The p-value for a single Gaussian component can be found as follows:

$$\begin{aligned}
\text{P-value} = & \int_{|x-\mu|>r} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \mathrm{d}x \\
= & 1 - \int_{|x-\mu|<r} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \mathrm{d}x \\
= & 1 - \left| \frac{2}{\sqrt{2\pi\sigma^2}} \int_{\mu}^{x} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \mathrm{d}z \right| \\
= & 1 - \left| \operatorname{erf}\left(\frac{x-\mu}{\sqrt{2\sigma^2}} \right) \right| \qquad (2.7)
\end{aligned}$$

since error function is defined as

$$\operatorname{erf}(z) = \frac{2}{\sqrt{\pi}} \int_{0}^{z} e^{-t^2} \mathrm{d}t. \qquad (2.8)$$

Let $E \in \{0,1\}$ be a random variable, where 1 indicates an extreme value and 0 otherwise. Then, for a GMM, we obtain the following p-value result:

$$\begin{aligned} P[E=1|x] =& \sum_{l=1}^{L} P[C=l|x]P[E=1|x,C=l] \\ =& \sum_{l=1}^{L} P[C=l|x]\left(1-\left|\mathrm{erf}\left(\frac{x-\mu}{\sqrt{2\sigma^2}}\right)\right|\right), \end{aligned} \tag{2.9}$$

where C is the component of origin and $P[C=l|x]$ is the mixture component posterior which can be found as follows:

$$P[C=l|x] = \frac{P[x|C=l]P[C=l]}{\sum_{k=1}^{L} P[x|C=k]P[C=k]} \tag{2.10}$$

Although packet size pairs are modeled in this thesis, the univariate GMM p-value given in (2.9) is also used in the network anomaly detection experiments when only one of the packet sizes of the pair is used due to a certain conditioning context. These conditioning contexts are defined in detail in Section 2.5.

2.3.2 Bivariate Gaussian Case

For this case, the corresponding multivariate integral can be exactly calculated by applying a whitening transformation, leading to the result that the p-value is 1 minus the Rayleigh cdf $F_R(r^2(\underline{x}))$, where $r^2(\underline{x}) = (\underline{x}-\underline{\mu})'\Sigma^{-1}(\underline{x}-\underline{\mu})$.

$$\begin{aligned} \text{P-value} =& \iint\limits_{(\underline{x}-\underline{\mu})'\Sigma^{-1}(\underline{x}-\underline{\mu})>r^2} \frac{1}{2\pi|\Sigma|^{1/2}} e^{-\frac{1}{2}(\underline{x}-\underline{\mu})'\Sigma^{-1}(\underline{x}-\underline{\mu})} \mathrm{d}x_1 \mathrm{d}x_2 \\ =& 1 - \iint\limits_{(\underline{x}-\underline{\mu})'\Sigma^{-1}(\underline{x}-\underline{\mu})\leq r^2} \frac{1}{2\pi|\Sigma|^{1/2}} e^{-\frac{1}{2}(\underline{x}-\underline{\mu})'\Sigma^{-1}(\underline{x}-\underline{\mu})} \mathrm{d}x_1 \mathrm{d}x_2 \end{aligned} \tag{2.11}$$

$$\begin{aligned} r^2(\underline{x}) =& (\underline{x}-\underline{\mu})'\Sigma^{-1}(\underline{x}-\underline{\mu}) \\ =& (\underline{x}-\underline{\mu})'G\Lambda^{-1}G'(\underline{x}-\underline{\mu}) \\ =& (\underline{x}-\underline{\mu})'G\Lambda^{-\frac{1}{2}}\Lambda^{-\frac{1}{2}}G'(\underline{x}-\underline{\mu}) \end{aligned}$$

where G is the matrix whose columns are the eigenvectors of Σ. Let

$$\underline{Y} = \Lambda^{-\frac{1}{2}} G'(\underline{x} - \underline{\mu}) \qquad (\|\underline{Y}\|^2 = r^2)$$

Then,

$$\begin{aligned} \mathrm{d}\underline{Y} =& |\Lambda|^{-\frac{1}{2}} \mathrm{d}\underline{x} \\ \mathrm{d}\underline{x} =& |\Lambda|^{\frac{1}{2}} \mathrm{d}\underline{Y} \end{aligned} \tag{2.12}$$

So, the p-value in (2.11) becomes

$$\text{P-value} = 1 - \iint\limits_{Y_1^2+Y_2^2 \le r^2} \frac{1}{2\pi|\Sigma|^{1/2}} e^{-\frac{1}{2}(Y_1^2+Y_2^2)} |\Lambda|^{\frac{1}{2}} \mathrm{d}Y_1 \mathrm{d}Y_2 \tag{2.13}$$

$$= 1 - \int_{-r}^{r} \int_{-\sqrt{r^2-Y_1^2}}^{\sqrt{r^2-Y_1^2}} \frac{1}{2\pi} e^{-\frac{1}{2}(Y_1^2+Y_2^2)} \mathrm{d}Y_2 \mathrm{d}Y_1 \tag{2.14}$$

By using the transformation

$$\begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} = \begin{pmatrix} \rho\cos\theta \\ \rho\sin\theta \end{pmatrix}, \tag{2.15}$$

we obtain the following result:

$$\text{P-value} = 1 - \int_0^r \int_0^{2\pi} \frac{\rho}{2\pi} e^{-\frac{1}{2}(\rho^2)} \mathrm{d}\theta \mathrm{d}\rho \tag{2.16}$$

$$= \int_0^r \rho e^{-\frac{\rho^2}{2}} \mathrm{d}\rho \tag{2.17}$$

$$= 1 - (1 - e^{-\frac{r^2}{2}}) \tag{2.18}$$

$$= e^{-\frac{r^2}{2}}, \quad r \ge 0. \tag{2.19}$$

As in Section 2.3.1, we obtain the following p-value result for a GMM:

$$P[E = 1|\underline{x}] = \sum_{l=1}^{L} P[C = l|\underline{x}] P[E = 1|\underline{x}, C = l]$$

$$= \sum_{l=1}^{L} P[C = l|\underline{x}] e^{-r_l^2(\underline{x})/2}, \tag{2.20}$$

where C is the component of origin and $P[C = l|\underline{x}]$ is the mixture component posterior which is:

$$P[C = l|\underline{x}] = \frac{P[\underline{x}|C = l]P[C = l]}{\sum_{k=1}^{L} P[\underline{x}|C = k]P[C = k]} \tag{2.21}$$

Bivariate GMM p-value in (2.20) will be used in most of the network anomaly detection experiments. Given a feature vector $\underline{x} \in \mathcal{R}^D$, we can define such p-values for all $M \equiv \binom{D}{2}$ feature pairs.

2.4 Experimental Setup – Internet Flows

The experiments in this section focus on anomaly detection in networks for Web traffic (TCP port 80 flows). The Web flows are obtained from the LBNL repository [58]. Each dataset includes packets captured from the same port and time of day. The experimental results obtained in this thesis are based on the datasets that provide Web flows at least 10 times more than the number of Zeus flows (39). There are 53 such files, all of which are used in the experiments. The file numbers provided in Table 2.1 are for short-hand notations to some of these files. The number of Web flows having at least 10 packets after the 3-way handshake is also given for each file in the table.

There are multiple anomalous classes that could be embedded amongst Web traffic. One such is Zeus botnet traffic [59], which tries to disguise itself amongst the Web traffic. Another is P2P traffic. For P2P-Web discrimination, we were able to find P2P and Web flows from the same domain. The same dataset (File 2) obtained from the LBNL repository is used to extract both types of flows. The Web flows from this data set were also used in the Zeus experiments. Since LBNL datasets do not specify which are the P2P flows, we used the port-mapper ([122]) obtained from the Cambridge dataset [43]. The Cambridge dataset includes and explicitly annotates the P2P flows, which enabled us to acquire the source and destination ports[2] that define the P2P flows. Based on these port-supervised examples, we learned a C4.5 decision tree to distinguish P2P from non-P2P flows. We then applied this decision tree to the LBNL dataset (File 2) to

[2]We could have used only destination ports, but the port-mapper accuracy is higher when both source and destination ports are used.

File Number	File Name	Number of Web Flows
File 1	20041215-0510.port008	2925
File 2	20041215-1343.port008	5413
File 3	20041215-1443.port010	1634
File 4	20041215-1242.port006	4543
File 5	20041215-1142.port003	5472
File 6	20050106-1423.port026	4375
File 7	20050106-1727.port006	1716
File 8	20041215-0711.port015	18409
File 9	20041004-1326.port006	667
File 10	20041215-0410.port006	447
File 11	20041216-1518.port006	2875
File 12	20050106-1827.port006	1551
File 13	20050107-1323.port026	4933
Combined File	All port006 files combined	11838

Table 2.1. LBNL files used in this thesis

identify the LBNL file's P2P flows[3].

In this thesis, in the anomaly detection experiments, 10-fold cross-validation is used for the training-test split of a dataset. Web flows (from a single dataset) were partitioned into 10 folds (with approximately the same number of flows). The Web flows in 9 of these folds form the training set and the remaining fold combined with all of the anomalous (Zeus or P2P) flows makes up the test fold (Zeus flows are obtained from [59]). This is repeated 10 times for a dataset, selecting a different fold of Web flows to be included in the test set at each time. An ROC plot is obtained for each of these. ROC plots shown in this thesis show the average of these 10 ROC plots.

The number of Zeus flows having at least 10 packets after the 3-way handshake is 39. File 2 is used for P2P experiments. There are 271 P2P flows in this dataset.

These flows are obtained by using Tshark, which is terminal based Wireshark [3]. Packet fields (such as IP packet size, TCP port number, etc.) can be extracted from pcap (packet capture) files, by using the proper field in Tshark. A complete list of fields can be found in [3].

[3]Web (HTTP) flows' TCP destination port is 80 and the decision tree does not assign port 80 to the P2P class. Thus, the Web and P2P classes are disjoint, removing any possibility for ambiguity concerning our flow labeling process for the LBNL datasets.

2.5 Feature Space Representations

Detecting Zeus flows as Web anomalies is not a trivial task – the Zeus bot was constructed to mimic Web flows, and in previous work [22], using a set of features including those proposed in [64], we found that several standard *supervised* classifiers (given labeled training examples of both Web and the bot) were unable to reliably discriminate Web from Zeus. (Unsupervised) anomaly detection of bot flows amongst Web flows is even harder than supervised classification. However, unlike the features proposed in [64], we propose use of a feature space that preserves the bidirectional packet size *sequence* information. This feature representation is provided in Section 2.5.2.

All of the representations use the first 10 packets after the 3-way handshake flow start ([102]) of each (TCP) flow (in order to have a system with low latency). The sizes and directions of these packets are used. Also, ACK packets are crucial for TCP flows and these are packets that have no payload, but only header. A minimal IP packet with no payload has an IP packet size of 40. But there is the possibility of using Selective ACK (SACK), which is used to acknowledge only certain portions of the traffic. This modified usage of ACKs may lead to having 52 as an ACK packet size. Actually, 64 is also a possible ACK packet size when SACK is used, but this is not frequently seen. Therefore, the approaches that use whether there is an ACK packet or not as conditioning context can use 40 or 52 packet sizes as an indicator for an ACK. A more careful approach can treat the packets that have payload size 0 as ACK packets, which will make little or no difference relative to choosing the packets with IP packet size 40 or 52.

But, there are different possible ways to utilize packet size, direction, and ACK information, each having its own pros and cons. These are explained in the following subsections. Considering the tradeoffs and the experimental results, we arrived at the conclusion that the best feature representation among these is the "alternating feature representation" that is provided in Section 2.5.2.

2.5.1 Lossless feature representation

Given 10 packet sizes and directions for each flow, the first well-grounded approach is using ACK and direction information as conditioning context in GMM modeling and treating them as categoricals.

In this approach, we have a 10-dimensional feature vector that is comprised of the IP packet sizes of the 10 packets. The bivariate models are generated for each pair of features, which makes $\binom{10}{2}$ models in total. But, one should keep in mind that the

possibility of encountering categorical features (ACK) in this approach, leads to 4 cases for building models:

1. No feature is categorical: In this case, the bivariate GMM models need to be obtained.
2. First feature is categorical: The univariate GMM models are obtained for the second feature of the pair conditioned on the first feature is categorical.
3. Second feature is categorical: The univariate GMM models are obtained for the first feature of the pair conditioned on the second feature is categorical.
4. Both features are categorical: No GMM models are obtained. Just the relevant probabilities are calculated.

Each case can use the samples that fall into that category, *e.g.*, bivariate models for a pair of features can only be obtained by training on the samples that have non-categorical values for those particular features. Further division of the training set is caused by conditioning on the directions of the packets (client-to-server (C) or server-to-client (S)). In fact, the situation is even worse than it seems, because the size of the subgroups of the training set will not have equal sizes. So, some cases will have much less than $R/16$ (or even 0) samples, if there are R samples in total. Hence, the sample size available to each case becomes very limited compared to the total available training set. This division of the training set into subgroups corresponding to each conditioning context is depicted in Figure 2.3. To remind, in the figure, C and S at the third level denote the directions of each of the packets.

After building the models, the next step is to calculate the p-values. The component posteriors of the GMM for univariate and bivariate modeling are calculated as in (2.10) and (2.21), respectively. As mentioned before, $E \in \{0, 1\}$ is a random variable, where 1 indicates an extreme value and 0 otherwise. Consider these random variables for the packet information for the derivations of p-values:

- Packet sizes: $\underline{X} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$
- Packet directions: $\underline{U} = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$. $U_i \in \{C, S\}$ where C denotes client-to-server direction and S is the server-to-client direction.

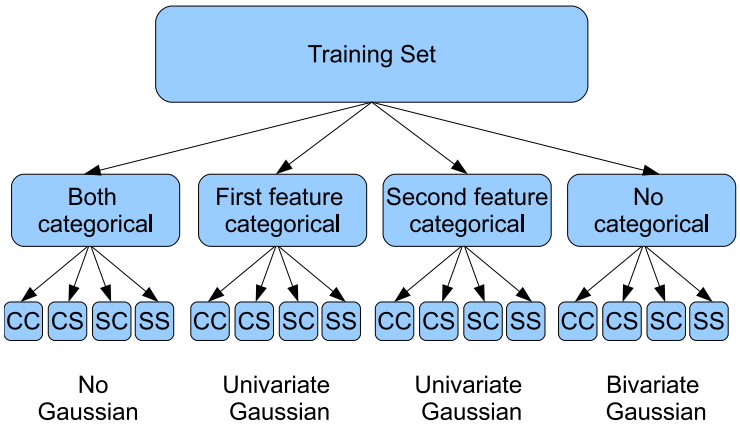

Figure 2.3. Division of the training based on conditioning contexts in lossless feature representation

- ACK indicators: $\underline{A} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$. $A_i = 1$ indicates that the packet is an ACK packet and $A_i = 0$ indicates that it is not an ACK. ($i \in \{1, 2\}$).

For the cases where one of the features is categorical, p-value is found as follows, which is obtained by modifying (2.9):

$$\begin{aligned}
\text{P-value} =& P[E = 1, A_1 = 1, A_2 = 0, \underline{U} = \underline{u} | \underline{X} = \underline{x}] \\
=& P[A_1 = 1, A_2 = 0, \underline{U} = \underline{u} | \underline{X} = \underline{x}] \cdot \\
& P[E = 1 | \underline{X} = \underline{x}, A_1 = 1, A_2 = 0, \underline{U} = \underline{u}] \\
=& P[A_1 = 1, A_2 = 0, \underline{U} = \underline{u}] \cdot \\
& \sum_{l=1}^{L} \{P[C = l | \underline{X} = \underline{x}, A_1 = 1, A_2 = 0, \underline{U} = \underline{u}] \cdot \\
& P[E = 1 | \underline{X} = \underline{x}, C = l, A_1 = 1, A_2 = 0, \underline{U} = \underline{u}]\} \\
=& P[A_1 = 1, A_2 = 0, \underline{U} = \underline{u}] \cdot \\
& \sum_{l=1}^{L} P[C = l | \underline{X} = \underline{x}, A_1 = 1, A_2 = 0, \underline{U} = \underline{u}] \left(1 - \left|\text{erf}\left(\frac{x_2 - \mu}{\sqrt{2\sigma^2}}\right)\right|\right)
\end{aligned} \tag{2.22}$$

where it is assumed that only the 1st packet is an ACK. If only the 2nd packet is an ACK, then x_1 needs to be swapped with x_2 in (2.22).

As mentioned above, when both of the features in the feature pair are not categorical, the p-values need to be calculated by using bivariate GMMs. This is done by modifying

(2.20) in the following way:

$$\begin{aligned}
\text{P-value} =& P[E=1, A_1=0, A_2=0, \underline{U}=\underline{u}|\underline{X}=\underline{x}] \\
=& P[A_1=0, A_2=0, \underline{U}=\underline{u}|\underline{X}=\underline{x}]\cdot \\
& P[E=1|\underline{X}=\underline{x}, A_1=0, A_2=0, \underline{U}=\underline{u}] \\
=& P[A_1=0, A_2=0, \underline{U}=\underline{u}]\cdot \\
& \sum_{l=1}^{L}\{P[C=l|\underline{X}=\underline{x}, A_1=0, A_2=0, \underline{U}=\underline{u}]\cdot \\
& P[E=1|\underline{X}=\underline{x}, C=l, A_1=0, A_2=0, \underline{U}=\underline{u}]\} \\
=& P[A_1=0, A_2=0, \underline{U}=\underline{u}]\cdot \\
& \sum_{l=1}^{L} P[C=l|\underline{X}=\underline{x}, A_1=0, A_2=0, \underline{U}=\underline{u}]e^{-r_l^2(\underline{x})/2}
\end{aligned} \tag{2.23}$$

The other case, in which both of the features are categorical, only the relevant probabilities are in effect for the p-value calculations (of course, (2.24) can be calculated in different ways, *e.g.*, by conditioning the packets being ACKs on their directions).

$$\text{P-value} = P[A_1=1, A_2=1, \underline{U}=\underline{u}] \tag{2.24}$$

The disadvantage of this feature representation is that it may require large null training set sample sizes to accurately model densities under every conditioning context shown in Figure 2.3, since each case can build model using only the available part of the training set to that case.

2.5.2 Alternating feature representation

In order to both use the packet direction information without giving up full use of the available training set, we propose a different feature representation. We define a 20-dimensional feature vector consisting of the sizes of these 10 packets, assuming packets strictly alternate client-to-server (C) and server-to-client (S). A zero packet size is inserted between 2 consecutive packets having the same direction, indicating the absence of a packet in the reverse direction between these 2 packets. Hence, sizes of the first 10 packets (after the 3-way handshake) are deterministically placed in locations within this 20-dimensional feature vector, given the particular packet direction sequence (for the first ten packets). Zeros are placed in the remaining ten locations. Notice that

this feature vector preserves bidirectional packet size sequence information. But, value of each feature in this 20-dimensional feature vector depends on the directions of the previous packets of the flow. This is not the case in lossless representation, since packet sizes are conditioned on the directions of each packet in that representation. Examples of alternatin feature representation are provided in Table 2.2. Suppose that the packet sizes of the 10 packets (after 3-way) are S1, S2, ..., S10. In the table, how the feature representation changes due to the different packet directions is visualized.

Packet Directions	Feature Representation
C S C S C S C S C S	S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 0 0 0 0 0 0 0 0 0 0
C S C S C C C C C C	S1 S2 S3 S4 S5 0 S6 0 S7 0 S8 0 S9 0 S10 0 0 0 0 0
S S S S S S S S S S	0 S1 0 S2 0 S3 0 S4 0 S5 0 S6 0 S7 0 S8 0 S9 0 S10

Table 2.2. Alternating Feature Representation

For Zeus-Web discrimination, since the Zeus flows and the Web flows are captured from different domains, packet interarrival time information is not exploited due to a lack of realistic timing information for the Zeus flows. For P2P-Web discrimination, since the source of both types of flows is the same domain, the timing information could be used. However, timing is not exploited in this thesis. In addition, no payload information is exploited (it is unavailable to us; moreover, encryption can easily defeat deep packet inspection).

Notice that in this representation, ACK packets (of size either 40 or 52) and the lack of alternating packet (represented by a zero) are heuristically treated just like any other packet size value, modelled by Gaussian mixture models (GMMs).

Also, it is worth mentioning that the smallest IP packet size is 40. This IP packet size value is used by ACK packets, which are used frequently in TCP flows. This will lead to a sharp (with very low variance) Gaussian component with mean 40 (corresponding to that feature). This Gaussian component will have negligible value at 0, which is why we can comfortably place 0's to obtain a 20-dimensional feature vector by using 10 packets as explained above. Insertion of 0's will lead to another Gaussian component with mean 0 (for the corresponding feature), which will again be very narrow, not affecting the closest packet size value (40).

Since all of the packet sizes are regarded as continuous, only bivariate GMM modeling is employed in p-value calculations. P-value of each sample is calculated as follows;

$$\text{P-value} = P[E = 1 | \underline{X} = \underline{x}]$$

$$= \sum_{l=1}^{L} P[C = l|\underline{X} = \underline{x}]P[E = 1|\underline{X} = \underline{x}, C = l]$$
$$= \sum_{l=1}^{L} P[C = l|\underline{X} = \underline{x}]e^{-r_l^2(\underline{x})/2},$$

which is derived in Section 2.3.2.

2.5.3 Alternating feature representation - with categorical feature 0

We have achieved finding a feature representation by proposing the approach in Section 2.5.2. But, the question arises about the concerns on categorical features that is mentioned in Section 2.5.1. In fact, since the alternating feature representation introduces 0's in places of missing packets for certain places in the 20-dimensional feature space, this is another source of categorical feature presence.

This leads us to regard the artificial 0's as categorical features and use them as conditioning context embedded in the alternating feature representation given in Section 2.5.2.

Similar to the representation in Section 2.5.1, we condition based on the categorical features here. But unlike that, we do not have conditioning based on the direction of the packets, since the alternating feature space has already taken into account direction information.

The p-value calculation for each case is provided below:

1. No feature is categorical: None of the features in the feature pair is 0.

$$\begin{aligned}
\text{P-value} =& P[E = 1, X_1 \neq 0, X_2 \neq 0|\underline{X} = \underline{x}] \\
=& P[X_1 \neq 0, X_2 \neq 0]\cdot \\
& \sum_{l=1}^{L}\{P[C = l|\underline{X} = \underline{x}, X_1 \neq 0, X_2 \neq 0]\cdot \\
& P[E = 1|\underline{X} = \underline{x}, C = l, X_1 \neq 0, X_2 \neq 0]\} \\
=& P[X_1 \neq 0, X_2 \neq 0] \sum_{l=1}^{L} P[C = l|\underline{X} = \underline{x}, X_1 \neq 0, X_2 \neq 0]e^{-r_l^2(\underline{x})/2} \quad (2.25)
\end{aligned}$$

2. One feature is categorical: One of the features is 0. Suppose that only x_1 is non-zero

in the feature pair.

$$\begin{aligned}\text{P-value} =& P[E=1, X_1 \neq 0, X_2 = 0 | \underline{X} = \underline{x}] \\ =& P[X_1 \neq 0, X_2 = 0] \sum_{l=1}^{L} \{P[C = l | X_1 = x_1, X_1 \neq 0, X_2 = 0] \cdot \\ & P[E = 1 | X_1 = x_1, C = l, X_1 \neq 0, X_2 = 0]\} \\ =& P[X_1 \neq 0, X_2 = 0] \cdot \\ & \sum_{l=1}^{L} P[C = l | X_1 = x_1, X_1 \neq 0, X_2 = 0] \left(1 - \left|\text{erf}\left(\frac{x_1 - \mu}{\sqrt{2\sigma^2}}\right)\right|\right) \end{aligned} \tag{2.26}$$

3. Both features are categorical: Both of the features are 0.

$$P[E = 1 | \underline{x}] = P[X_1 = 0, X_2 = 0] \tag{2.27}$$

2.5.4 Alternating feature representation - with categorical features 0 and ACK (together)

The previous approach treats only 0's as categoricals. But, following the discussion in Section 2.5.1, ACK packets can also be treated as categoricals. This changes the definition of being a categorical feature, compared to Section 2.5.3.

The p-value calculation for each case is provided below:

1. No feature is categorical: None of the features in the feature pair is 0 or ACK.

$$\begin{aligned}\text{P-value} =& P[E=1, X_1 \neq 0, A_1 = 0, X_2 \neq 0, A_2 = 0 | \underline{X} = \underline{x}] \\ =& P[X_1 \neq 0, A_1 = 0, X_2 \neq 0, A_2 = 0] \cdot \\ & \sum_{l=1}^{L} \{P[C = l | \underline{X} = \underline{x}, X_1 \neq 0, A_1 = 0, X_2 \neq 0, A_2 = 0] \cdot \\ & P[E = 1 | \underline{X} = \underline{x}, C = l, X_1 \neq 0, A_1 = 0, X_2 \neq 0, A_2 = 0]\} \\ =& P[X_1 \neq 0, A_1 = 0, X_2 \neq 0, A_2 = 0] \cdot \\ & \sum_{l=1}^{L} P[C = l | \underline{X} = \underline{x}, X_1 \neq 0, A_1 = 0, X_2 \neq 0, A_2 = 0] e^{-r_l^2(\underline{x})/2} \end{aligned} \tag{2.28}$$

2. One feature is categorical: One of the features is 0 or ACK. For GMM modeling purposes, the value of the categorical doesn't matter. The model is built upon the

samples that have 0 or ACK for the categorical feature. The particular value of the categorical feature affects only the multiplicating probability.

$$\begin{aligned}
\text{P-value} =& P[E=1,(X_1\neq 0,A_1=0),(X_2=0\cup A_2=1)|\underline{X}=\underline{x}] \\
=& P[(X_1\neq 0,A_1=0),(X_2=0\cup A_2=1)]\cdot \\
& \sum_{l=1}^{L}\{P[C=l|X_1=x_1,(X_1\neq 0,A_1=0),(X_2=0\cup A_2=1)]\cdot \\
& P[E=1|X_1=x_1,C=l,(X_1\neq 0,A_1=0),(X_2=0\cup A_2=1)]\} \\
=& P[(X_1\neq 0,A_1=0),(X_2=0\cup A_2=1)]\cdot \\
& \sum_{l=1}^{L}P[C=l|X_1=x_1,(X_1\neq 0,A_1=0),(X_2=0\cup A_2=1)]\cdot \\
& \left(1-\left|\text{erf}\left(\frac{x_1-\mu}{\sqrt{2\sigma^2}}\right)\right|\right)
\end{aligned} \tag{2.29}$$

where $x_2\in\{0,40,52\}$, if ACK packets can take values 40 or 52. If the 1st feature was a categorical instead of the 2nd, then x_1 and x_2 would be swapped.

3. Both features are categorical: Both of the features are 0 or ACK.

$$\text{P-value} = P[(X_1=0\cup A_1=0),(X_2\neq 0\cup A_2=1)] \tag{2.30}$$

Here, we are assigning a single event $(X_1=0\cup A_1=0)$ to being categorical, although there are 2 different types of categoricals, which are 0 and ACK. Considering 0 or ACK packets as the same type categoricals bears the danger of mixing the statistical properties of the ACK packets and non-existing locations (0 value). This leads us to the next feature representation in Section 2.5.5.

2.5.5 Alternating feature representation - with categorical features 0 and ACK (separately)

Here, we are using a similar approach to Section 2.5.4, but we treat non-existing locations (0's) and ACKs as different kinds of categoricals. This affects which samples to use in the model building phase.

The p-value calculation for each case is provided below:

1. No feature is categorical: None of the features in the feature pair is 0 or ACK. P-value is calculated with (2.28).

2. One feature is categorical: Assume that the 1st feature is continuous and the 2nd feature is 0 or an ACK. (If the 1st feature was a categorical instead of the 2nd, then x_1 and x_2 would be swapped.) Assume that the 2nd feature is 0.

$$\begin{aligned}
\text{P-value} =& P[E=1,(X_1\neq 0,A_1=0),(X_2=0,A_2=0)|\underline{X}=\underline{x}] \\
=& P[(X_1\neq 0,A_1=0),(X_2=0,A_2=0)]\cdot \\
& \sum_{l=1}^{L}\{P[C=l|X_1=x_1,(X_1\neq 0,A_1=0),(X_2=0,A_2=0)]\cdot \\
& P[E=1|X_1=x_1,C=l,(X_1\neq 0,A_1=0),(X_2=0,A_2=0)]\} \\
=& P[(X_1\neq 0,A_1=0),(X_2=0,A_2=0)]\cdot \\
& \sum_{l=1}^{L}\{P[C=l|X_1=x_1,(X_1\neq 0,A_1=0),(X_2=0,A_2=0)]\cdot \\
& \left(1-\left|\text{erf}\left(\frac{x_1-\mu}{\sqrt{2\sigma^2}}\right)\right|\right)\}
\end{aligned} \tag{2.31}$$

3. Both features are categorical: Both of the features are 0 or ACK. Since there is no GMM model to talk about in this case, only the probabilities are in effect. P-value calculation is as in (2.30).

2.5.6 Alternating feature representation - with categorical features 0 and ACK (separately) (normalized p-values)

This approach is similar to Section 2.5.5, but with one difference. The p-values for each test are normalized so that they are distributed between the minimum p-value that is achieved by that test and 1. The rest of the calculations are the same as in Section 2.5.5. The need for doing this arouse from the fact that when we use categorical features, probability of belonging to a category (or not) is in play. This prevents most of the p-values from reaching value 1. The importance of high p-values is they don't tend to get involved in the early cluster detections, which is a beneficial property for anomaly detection with our p-value clustering approach.

2.5.7 Alternating feature representation - with categorical features 0 and ACK (separately)(without probabilities)

Here, approach described in Section 2.5.5 is used, but the probabilities corresponding to each conditioning context are omitted in the p-value calculations.

Hence, the p-value calculation for each case becomes as below:

1. No feature is categorical: None of the features in the feature pair is 0 or ACK.

$$\begin{aligned}\text{P-value} =& P[E=1|\underline{X}=\underline{x}, X_1\neq 0, A_1=0, X_2\neq 0, A_2=0]\\ =& \sum_{l=1}^{L}\{P[C=l|\underline{X}=\underline{x}, X_1\neq 0, A_1=0, X_2\neq 0, A_2=0]\cdot\\ & P[E=1|\underline{X}=\underline{x}, C=l, X_1\neq 0, A_1=0, X_2\neq 0, A_2=0]\}\\ =& \sum_{l=1}^{L} P[C=l|\underline{X}=\underline{x}, X_1\neq 0, A_1=0, X_2\neq 0, A_2=0]e^{-r_l^2(\underline{x})/2} \end{aligned} \tag{2.32}$$

2. One feature is categorical: One of the features is 0 or ACK. The GMM model is built, based on the value of the categorical feature, as explained in Section 2.5.5.

$$\begin{aligned}\text{P-value} =& P[E=1|\underline{X}=\underline{x}, X_1\neq 0, A_1=0, X_2=0, A_2=0]\\ =& \sum_{l=1}^{L}\{P[C=l|X_1=x_1, (X_1\neq 0, A_1=0), (X_2=0, A_2=0)]\cdot\\ & P[E=1|X_1=x_1, C=l, (X_1\neq 0, A_1=0), (X_2=0, A_2=0)]\}\\ =& \sum_{l=1}^{L}\{P[C=l|X_1=x_1, (X_1\neq 0, A_1=0), (X_2=0, A_2=0)]\cdot\\ & \left(1-\left|\text{erf}\left(\frac{x_1-\mu}{\sqrt{2\sigma^2}}\right)\right|\right)\} \end{aligned} \tag{2.33}$$

   where $x_2 = 0$. Like mentioned before, if the 1st feature was a categorical instead of the 2nd, then x_1 and x_2 would be swapped. Above, it is assumed that only 2nd feature categorical.

3. Both features are categorical: Both of the features are 0 or ACK. P-value is 1, since only tool to quantify the p-values were probabilities of the features getting the particular categorical values. Removal of these probabilities left this case's p-value calculation with 1.

$$\text{P-value} = 1 \tag{2.34}$$

2.6 Supervised Classification Results

Below, supervised average test set decision tree classification performances for our proposed 20-dimensional alternating feature space (Section 2.5.2) are provided. The confusion matrix shown in Table 2.4 can be compared to the one shown in Table 2.3, which shows the result based on the features proposed in [64] and those used in [22], which do not preserve packet sequence information. These results are obtained by using C4.5 decision tree in Weka ([46]) (in Weka, this decision tree is called J48).

Note that the features used in [22] do not lead to accurate classification of the Zeus flows. This comparison suggests the importance of exploiting statistical dependencies between packet sizes in the length-20 sequence, for discriminating Zeus from Web. If such exploitation is essential for supervised discrimination, it should also be pivotal for (the more challenging problem of) anomaly detection of Zeus flows, amongst a batch of Web flows.

9-D Feature Set Used in [22]	Detected as Web	Detected as Zeus
Actual Web	2895	30
Actual Zeus	17	22

Table 2.3. Supervised decision tree confusion matrix for the features (except for timing based features) used in [22] (9-D Feature set) for File 1

	Detected as Web	Detected as Zeus
Actual Web	2896	29
Actual Zeus	9	30

Table 2.4. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 1

	Detected as Web	Detected as Zeus
Actual Web	5382	31
Actual Zeus	8	31

Table 2.5. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 2

	Detected as Web	Detected as Zeus
Actual Web	1607	27
Actual Zeus	13	26

Table 2.6. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 3

	Detected as Web	Detected as Zeus
Actual Web	4513	30
Actual Zeus	12	27

Table 2.7. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 4

	Detected as Web	Detected as Zeus
Actual Web	5427	49
Actual Zeus	15	24

Table 2.8. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 5

	Detected as Web	Detected as Zeus
Actual Web	4336	40
Actual Zeus	12	27

Table 2.9. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 6

	Detected as Web	Detected as Zeus
Actual Web	1689	27
Actual Zeus	11	28

Table 2.10. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 7

	Detected as Web	Detected as Zeus
Actual Web	18332	67
Actual Zeus	8	31

Table 2.11. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 8

	Detected as Web	Detected as Zeus
Actual Web	11698	101
Actual Zeus	11	28

Table 2.12. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for Combined File

	Detected as Web	Detected as P2P
Actual Web	5399	14
Actual P2P	10	261

Table 2.13. Supervised decision tree confusion matrix for 20-dimensional alternating feature space for File 2 (P2P)

Chapter 3

Anomaly Detection – Sample-wise Detection Approach

In this chapter, our sample-wise sequential anomaly detection approach is explained. Experimental results are provided at the end of the chapter.

3.1 Sample-wise Anomaly Detection with Growing Number of Tests

Consider sequential detection applied to the batch $\mathcal{X} = \{\underline{x}_i, i = 1, \ldots, N, \underline{x}_i \in \mathcal{R}^D\}$ and suppose that k detections have already been made, with $\mathcal{T}^{(k)} \subset \mathcal{T}$ the set of tests used in making the first k detections, $\mathcal{T}$ the set of all possible tests that may be used. For now, we will suppose that, before any detections are made, we start with a default initial test, *i.e.* $\mathcal{T}^{(0)} = \{t_0\}$. (In the sequel, we specify how the first test is chosen). Let $\mathcal{S}^{(k)} \subset \mathcal{S} \equiv \{1, 2, \ldots, N\}$ denote the indices of the first k detected samples, with $\mathcal{S}^{(0)} = \emptyset$.

In making the kth detection, we thus have two choices:

1) Use a test from the existing set of tests $\mathcal{T}^{(k-1)}$,

2) Use a new test.

Let $v_k \in \{0, 1\}$ be defined as follows:

$$v_k = \begin{cases} 1, & \text{if a } \textit{new} \text{ test is used in making the } k\text{th detection} \\ 0, & \text{if an existing test is used in the } k\text{th detection.} \end{cases}$$

If $v_k = 1$, then we set $\mathcal{T}^{(k)} = \mathcal{T}^{(k-1)} \cup \{t_k\}$, where t_k denotes the new test (one not used in the first $k-1$ detections). Now, suppose that p_1 is the smallest p-value measured for any of the samples in the set $\mathcal{S} - \mathcal{S}^{(k-1)}$ using the existing tests and let p_2 be the smallest p-value measured using a new test. Under choice 1), the probability of observing a p-value more extreme than p_1, under the null hypothesis (assuming independent tests) is $1 - (1-p_1)^{|\mathcal{T}^{(k-1)}|(N-k+1)}$. Under choice 2, this probability is $1 - (1-p_2)^{(|\mathcal{T}-\mathcal{T}^{(k-1)}|)(N-k+1)}$. To maximize statistical significance of the kth detection, we should make the choice that gives the smaller of the two probabilities. Also, we note that this probability is our estimate of the probability that the kth detection is a false alarm, with one minus this probability our estimate of the probability that this is a true detection. Thus, our choice maximizes the "increment" that the kth detection gives to (effectively, our estimate of) the true detection rate (P_{D}) and simultaneously minimizes the increment given to our estimate of the false alarm rate (P_{FA}).

More generally, we can write an objective function that measures the *aggregate* statistical significance of the first L detections. First, we define

$$p^*(\mathcal{T}', \mathcal{S}') \equiv \min_{t \in \mathcal{T}', s \in \mathcal{S}'} p(t, s),$$

where $p(t, s)$ is the p-value for test t on sample s. Then, we have:

$$\begin{aligned} S_A(L) = &\sum_{k=1}^{L} v_k (1 - (1 - p^*(\mathcal{T} - \mathcal{T}^{(k-1)}, \mathcal{S} - \mathcal{S}^{(k-1)}))^{(|\mathcal{T}-\mathcal{T}^{(k-1)}|)(N-k+1)}) \\ &+ (1 - v_k)(1 - (1 - p^*(\mathcal{T}^{(k-1)}, \mathcal{S} - \mathcal{S}^{(k-1)}))^{|\mathcal{T}^{(k-1)}|(N-k+1)}). \end{aligned} \tag{3.1}$$

Note that, based on our above discussion, (3.1) can also be interpreted as an (aggregated) estimate of the false alarm rate, associated with the first L detections ($P_{\mathrm{FA}}(L)$). In minimizing this quantity, we are also equivalently maximizing an (aggregated) estimate of the true detection rate ($P_{\mathrm{D}}(L)$). In other words, minimizing (3.1) is consistent with maximizing "front-loaded" partial area (for first L detections) under an (estimated) ROC curve (AUC). We aim to maximize the partial AUC associated with the first L detections (as opposed to the total AUC) because a human operator (or some automated response system) may only be able to handle (corroborate and act on) L detections for a batch of size N, and detections should clearly be prioritized by their significance level. We propose two greedy strategies for maximizing (3.1). The first simply seeks, at the kth detection step, to maximize significance of this single detection without consideration of the effect on significance of future detections. The second (lookahead) strategy should

obtain better decision sequences (in the sense of $S_A(L)$) than those found by the first method, albeit while requiring greater computational complexity.

3.1.1 Strategy 1: No Lookahead

1) *Randomly* select the first test, t_0, from the full set $\mathcal{T}$. Set $\mathcal{T}^{(0)} = \{t_0\}$.

2) For $k = 1, \ldots, L$:
   Set $v_k \leftarrow 0$. Reset v_k to 1 if

$$1 - (1 - p^*(\mathcal{T} - \mathcal{T}^{(k-1)}, \mathcal{S} - \mathcal{S}^{(k-1)}))^{(|\mathcal{T} - \mathcal{T}^{(k-1)}|)\tau}$$

$$< 1 - (1 - p^*(\mathcal{T}^{(k-1)}, \mathcal{S} - \mathcal{S}^{(k-1)}))^{|\mathcal{T}^{(k-1)}|\tau}.$$

   where $\tau \equiv N - k + 1$. Denote the pair achieving maximum significance by (t_k, s_k).
   Set $\mathcal{S}^{(k)} = \int^{(k-1)} \cup \{s_k\}$.
   If $(v_k = 1)$ then $\mathcal{T}^{(k)} = \mathcal{T}^{(k-1)} \cup \{t_k\}$
   else $\mathcal{T}^{(k)} = \mathcal{T}^{(k-1)}$.
   Endfor

3) Output the detection sets $\mathcal{S}^{(L)}$ and $\mathcal{T}^{(L)}$.

We make the following observations about this approach:

i) Random selection of the first test is done so that, initially, we need only correct for a single test, in assessing statistical significance – if, instead, we were to initially evaluate all tests on all samples, and add the test achieving the smallest p-value on some sample, we need to correct for *all* tests at the very outset. In such a case, detections will simply be made in order of increasing p-values. It is only by starting from a randomly chosen initial test that parsimony in the use of tests and, thus, in multiple testing correction, can be achieved, with the detection sequence now chosen to minimize (3.1). Such parsimony is hoped to be beneficial when anomalies have a common statistical character, e.g. when an unknown anomalous class is present in the batch.

ii) Note that our procedure has *built-in* analytical significance assessment for each detection made – thus, one can stop making detections when the assessed significance (at some step) falls below a preset threshold (if the operator's capacity (L) has not yet been reached).

3.1.2 Strategy 2: Lookahead

At the first step, a single test will be used. At step 2, there is the choice of sticking to this test or adding a second test. At step 3, there are 4 possible choices for the sequence of number of tests in use: $\{(1,1,1),(1,1,2),(1,2,2),(1,2,3)\}$. More generally, for $U < L$ detections, there are 2^U such sequences, and we have not even considered the possible test configurations (the particular set of tests comprising a given number of detections sequence) that need to be evaluated for each such "number of tests" sequence. Thus, the solution space grows at least exponentially in U. Global optimization of (3.1) for large L is apparently infeasible. However, it is possible to improve on Strategy 1 at some computational expense. The key observation is that, whereas one particular test, if chosen at step k, may maximize statistical significance of the kth detection, another test, if instead chosen at step k, may help achieve greater *aggregate* statistical significance if one looks ahead to additional detection steps $k+1, k+2$, and so on. In other words, if there is test "clustering", wherein the same test (once added) will be repeatedly used for multiple consecutive (or closely spaced) detections, such a test may be more valuable (in the sense of (3.1)) than a test which, while maximizing statistical significance for the kth detection, will not be used subsequently. One strategy exploiting such "test clustering" is as follows.

Consider maximizing aggregate statistical significance of the *first three* detections made. As noted above, there are four possible choices for the number of tests sequence. For the first choice, $(1,1,1)$, with a single test used, the best test can be found with complexity $O(|\mathcal{T}|)$. For the sequences $(1,1,2)$ and $(1,2,2)$, finding the best pair of tests in general requires $O(|\mathcal{T}|^2)$ complexity (but this is the worst-case complexity – it can be reduced to $O(|\mathcal{T}|)$ if the two most significant p-values, using different tests, are not for the same samples). Likewise, for the sequence $(1,2,3)$, finding the best test triple will require, worst case, $O(|\mathcal{T}|^3)$ complexity, but this can be reduced even to $O(|\mathcal{T}|)$ if the three most significant detections, using different tests, are all for different samples. Having found the best test sequences for each of these four cases, we can select the case (with associated detections and tests) that achieves maximum aggregate statistical significance for the first three detections made. We can then make permanent *either* just the first detection or all three detections in the triple. This procedure can then be repeated. In the former case, this means we next consider detections 2, 3, and 4, while in the latter case we next consider detections 4, 5, and 6. If only the first detection in a triple is made permanent, this procedure "looks ahead" two samples, in making its detection decisions. If all three detections are made permanent, this procedure alternatively makes

three detections *jointly*. Note that we can certainly extend this approach to look ahead more than two samples, albeit with increased complexity.

3.1.3 Determining when to stop: significance assessment of detections:

Our Algorithm 1, which does not perform lookahead, naturally yields statistical significance assessments for each detection. Our lookahead procedure, which more directly maximizes aggregate statistical significance, also directly yields these significance assessments, but they may not be as accurate as in the non-lookahead case. Accordingly, coupled to Strategy 2, as an alternative, one can use standard empirical assessment of statistical significance. Specifically, we suppose that, separate from the batch $\mathcal{X}$, there is a fairly large database of examples from the known data class. One can then randomly draw numerous batches (all of size N) from this database. For each such (null) batch, one can apply our sample-wise detection procedure, which will yield a sequence of significance values, based on $1-(1-p)^{\#\text{tests}}$, associated with the detection sequence. For the kth detection in the *actual* batch, with significance value δ, one can then obtain an empirical significance level, measured as the fraction of kth significance values in the null batches that are smaller than δ. Clearly, there is an associated accuracy/computation tradeoff – complexity grows linearly with the number of batches (K) and the smallest significance level attainable for K batches is $\frac{1}{K}$. Even without *highly* accurate significance assessment, via their determination of a detection *sequence*, our detection approaches give an order of prioritization of samples for consideration (as {anomalous, suspicious, interesting}) by a human operator. This may be all that is necessary given that an operator has finite capacity for investigating anomalies.

3.2 Experimental Results

Figure 3.1 shows the number of tests used to make detections (averaged over all ten test folds), as a function of number of detections made, for our approaches and for the all-pairs approach. It can be observed, as expected, that our methods are more spartan in their use of tests than all-pairs.

In Figure 3.2, ROC curve results, which are also averaged over all ten test folds, are shown for a) use of a single joint test, based on the full 20-dimensional feature vector; b) use of all pairwise tests; c) use of our (no lookahead) Strategy 1, with growing number of tests; d) use of our lookahead strategy with modest lookahead order 2; e) detection based on aggregation (summing) the log p-values of all the tests. We note the poor performance

of the single joint test approach, and only very small differences in the ROC performance between our proposed detection strategies and the all-pairwise approach. Finally, the fact that the best performance is achieved by detecting flows based on aggregation of *all* log p-values suggests that, at least for this domain (detecting Zeus amongst Web flows), parsimonious use of tests may not be the optimal strategy, i.e., apparently useful information is gleaned by exploiting all twenty features (and all feature pairs). Likewise, aggregation of p-values is another way of utilizing all of the features, which performs the best among the methods for this dataset. By the same token, the poor performance of the single joint test suggests that how these features are jointly exploited is important – significant inaccuracy is apparently introduced in the GMM modeling/estimation for the joint (20-dimensional) feature space.

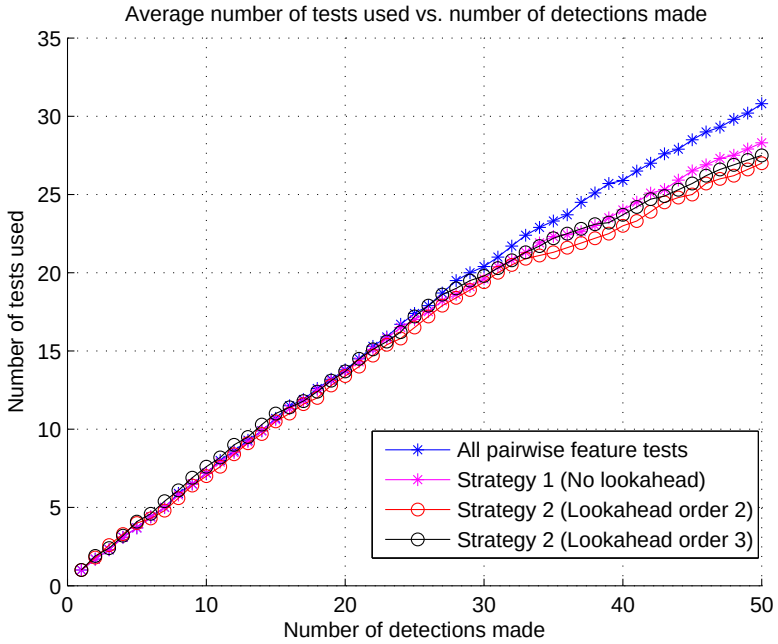

Figure 3.1. Number of tests used versus number of made detections, for several methods, in detecting Zeus bots among Web flows. (File 1)

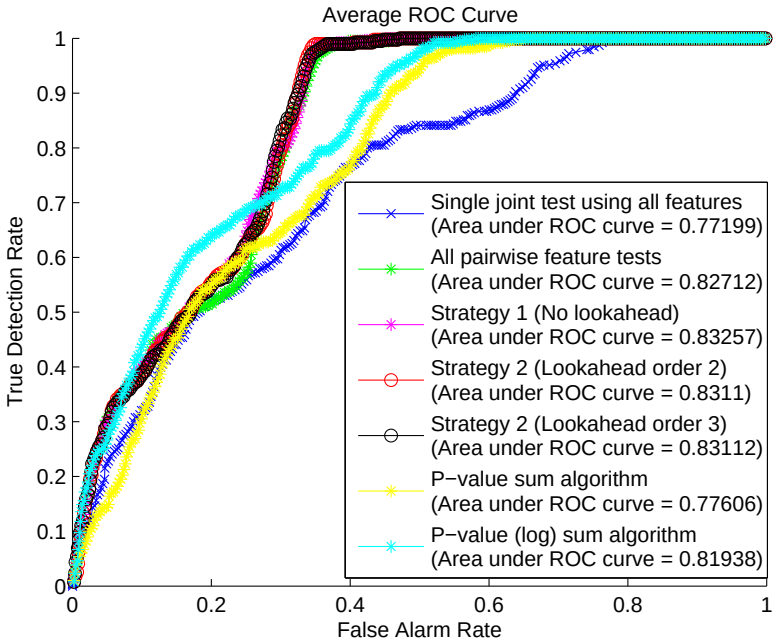

Figure 3.2. ROC curves for several anomaly detection methods, in detecting Zeus bots among Web flows. (File 1)

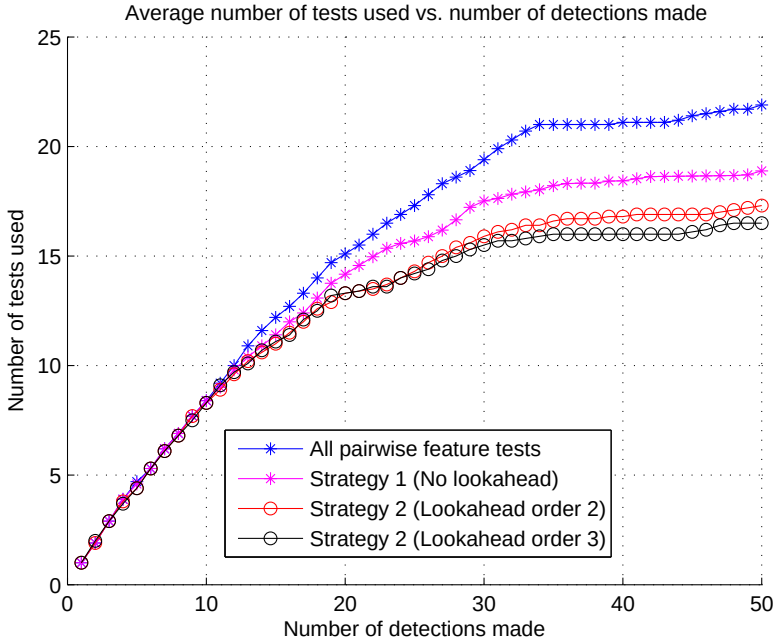

Figure 3.3. Number of tests used versus number of made detections, for several methods, in detecting Zeus bots among Web flows. (File 5)

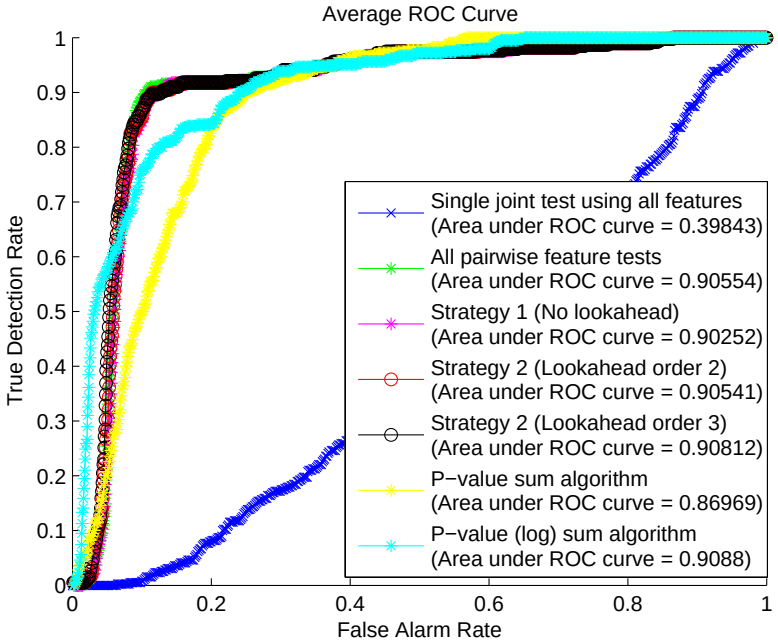

Figure 3.4. ROC curves for several anomaly detection methods, in detecting Zeus bots among Web flows. (File 5)

Chapter 4

Anomaly Detection – Cluster-wise Detection Approach

In this chapter, our cluster-wise anomaly detection approach is explained. Experimental results are provided at the end of the chapter.

4.1 Clustering Criterion and Algorithm

Our detection algorithm aims to find the most significant outlier clusters by assessing approximate joint p-values for candidate clusters. The p-value for a cluster of samples, defined over a subset of all M pairwise feature tests, is calculated by combining the individual p-values for all tests in the test subset, for all samples in the cluster. Let $p(t, s)$ be the p-value for test t on sample s. Then, by assuming tests are statistically independent[1], the p-value of a cluster factors as a product over the p-values of the individual tests, and can be calculated as follows[2]:

$$\begin{aligned}\text{Score} =& \mathcal{L}[\text{cluster with } N_c \text{ samples, } M_c \text{ tests}|\text{null hypothesis}] \\ =& N_c! \binom{N}{N_c} \binom{M}{M_c} \prod_{m=1}^{M_c} \prod_{n=1}^{N_c} p(i_m, j_n). \end{aligned} \tag{4.1}$$

[1] This is not really valid, since two pairwise feature tests may involve a common feature; moreover, for two tests involving disjoint feature pairs, the features across the tests may anyway possess statistical dependencies. However, this independence assumption is made for analytical tractability of our joint p-value assessment.

[2] The logarithm of (4.1) can be taken to avoid problems of numerical precision/underflow.

Here, N is the number of samples in the full data batch, N_c is the number of samples in the cluster, M is the total number of available tests, M_c the number of tests evaluated for the cluster, and with $p(i_m, j_n)$ the individual p-value corresponding to test i_m on sample j_n. The multiplicative factor $N_c!\binom{N}{N_c}\binom{M}{M_c}$ compensates the joint significance score to account for the number of candidate clusters of size (N_c, M_c), in a batch with N samples and M tests. This compensation aims to allow fair comparison of scores for any two candidate clusters that may possess different numbers of samples or different numbers of tests. Note that permutations are counted for the samples – *e.g.*, if $M_c = 1$, there are N_c choices for the smallest p-value, $N_c - 1$ for the second smallest p-value, and so on.

Note that, since p-values are less than one, the p-value product in (4.1) strictly decreases with more sample inclusions and use of more tests. However, since the assumption of test independence becomes grossly invalid as the size of the test subset M_c is increased, we must prevent "rewarding" the use of many tests in (4.1). We note that the compensation (penalty) term increases as more samples are included. Moreover, for $M_c < M/2$, the penalty term on the number of tests also increases for increasing number of tests. However, this penalty term in fact *decreases* as M_c increases beyond $M/2$. Thus, the $\binom{M}{M_c}$ penalty only dissuades large test subsets for $M_c < M/2$. Accordingly, we place an upper bound on M_c (which, in practice, we set well below the value $M/2$).

By forming clusters (a sample subset joint with a test subset) to minimize (4.1), we are identifying the jointly most anomalous subset of samples, and the subset of tests that elicits these anomalies. Accordingly, the score in (4.1) approximately assesses statistical significance like a traditional p-value (*e.g.*, (2.20)), but for a cluster of samples, rather than for a single sample.

If (4.1) is naively calculated for every possible candidate cluster, the computational complexity will be $O(2^{NM})$, which is huge even for small batches. This emphasizes the need for an efficient way to find the most significant clusters, minimizing (4.1). For this purpose, for a given candidate subset of M_c tests, we sort all samples in increasing order of their p-value products ($\prod_{m=1}^{M_c} p(i_m, j_n)$). The incremental contribution to the score obtained by including a new sample to a cluster with M_c tests and (currently) N_c samples is:

$$\frac{\text{Next score}}{\text{Current score}} = \frac{(N_c+1)!\binom{N}{N_c+1}\binom{M}{M_c}\prod_{m=1}^{M_c}\prod_{n=1}^{N_c+1} p_{i_m,j_n}}{N_c!\binom{N}{N_c}\binom{M}{M_c}\prod_{m=1}^{M_c}\prod_{n=1}^{N_c} p_{i_m,j_n}}$$

$$=(N - N_c) \prod_{m=1}^{M_c} p_{i_m, j_{N_c+1}} \tag{4.2}$$

where j_{N_c+1} is the index of the new sample. Thus, working in sorted order, we include samples in a cluster candidate so long as (4.2) is smaller than 1 with each new sample inclusion:

$$\begin{aligned} (N - N_c) \prod_{m=1}^{M_c} p_{i_m, j_{N_c+1}} &< 1 \\ \prod_{m=1}^{M_c} p_{i_m, j_{N_c+1}} &< \frac{1}{N - N_c} \end{aligned} \tag{4.3}$$

Another trick to speed up our algorithm is caching (storing) the p-value products for each sample at each test order as a pre-processing step. This increases the storage requirements, but so long as the test order M_c is kept low, this extra storage requirement is not huge. These strategies dramatically reduce the computation required to choose the best sample subset given a fixed set of tests, and to evaluate the associated joint significance score.

It is crucial to emphasize our p-value clustering algorithm forms the clusters starting from the lowest p-values, therefore from the most anomalous samples. As the algorithm continues to detect clusters of samples, N therefore $N - N_c$ will get smaller. So, $1/(N - N_c)$ term in (4.3) will get larger, allowing higher p-values to be used in left-hand side of (4.3).

Another point worth notable is that increasing M_c (which is the test combination order) allows higher p-values (therefore more samples) to the cluster to be detected. This is another reason to impose an upper bound on M_c. An example will help illustrate this relation between p-values and M_c. The effect of addition of a sample on the score of a cluster is given in (4.2). We continue adding samples as long as each new sample decreases the score of the cluster, which means (4.3) is satisfied. Suppose that there are 64 flows left in the test set. So, $N - N_c = 64$. Let's compare test combination orders 2 and 3 ($M_c = 2$ and 3). For order 2, geometric mean of the p-values of the current sample corresponding to the 2 tests must be no greater than 1/8. For order 3, this geometric mean becomes 1/4. So, using higher orders in the p-value clustering approach means allowing samples with higher p-values, as well as the ones with lower p-values, into the cluster. The lower the order is, the tighter the upper bound constraint on the p-value becomes.

A full description of our overall detection algorithm, built around detecting clusters that minimize (4.2), will be given in the next section.

4.2 Implementation Details of P-value Clustering

Bivariate GMM models for all $\binom{20}{2}$ feature pairs were fit to the Web training set flows via the Expectation-Maximization algorithm. The existence of categorical features (Section 2.5) led us to impose a lower bound on the diagonal components of the covariance matrices of the bivariate GMM models. We used 10 as a lower bound, since this prevents overlapping of any Gaussian components representing 40-sized and 52-sized ACK packets. Additionally, since the packet sizes take integer values, it is reasonable to choose a lower bound at least 1. We have observed that AUC performance is only modestly sensitive to this choice. To select the number of Gaussian components for each of the bivariate GMM models, the Bayesian Information Criterion (BIC) ([92]) was used. After fitting the GMM models, for each test batch data sample, for all pairwise feature tests, the p-values were calculated. Then, these p-values were used in the score function (4.1) to evaluate cluster candidates and find the most significant outlier cluster. All the samples in this cluster comprise the first set of detected samples. After their detection, these samples are removed from the test batch. Then, this process (detecting the samples of the most significant outlier cluster according to evaluation of (4.1) and removing them from the test batch) is repeated until the samples in the test batch are depleted. ROC plots were measured by counting the false alarms and true detections for each sequentially detected cluster and reflecting this in "jumps" in the ROC curve. The ten ROC plots obtained based on 10-fold cross-validation were used to calculate average ROC plots. The performance comparison metric for different methods can be selected as area under ROC (AUC). Another metric could be the true detection rate corresponding to a specified false alarm rate (such as 0.05, 0.1). A steeper curve implies earlier detection of most of the anomalous samples.

As the number of allowed tests in a candidate cluster is increased, the score function in (4.1) monotonically decreases. However, as noted earlier, due to the escalating inaccuracy of the test independence assumption for increasing test subset size M_c, this decrease in the score is not necessarily indicative of improvement in the ROC plot. In fact, as seen from the next section, increasing the maximum test order may result in a degradation in the AUC performance. Increasing the order also leads to larger cluster detections at each time. These effects led us to impose an upper bound on the allowed number of tests

to be used in a cluster, *i.e.*, the maximum test order. As we will observe from the ROC plots, this limitation helps the algorithm to ensure good AUC performance. Essentially, this forces focusing on the most discriminative tests for each cluster detection. Note also that these most significant tests are automatically updated for each sequential cluster detection, since the detection and removal of the most anomalous samples may lead to a different "most discriminative subset of tests" for the next cluster detection.

For a given maximum test order, the globally minimum score will be achieved by exhaustively evaluating all combinations of subsets of tests up to the maximum order. For example, for a maximum test order of 3, we need to consider all possible test subset combinations with one, two, and three tests, and find the cluster that has the minimum score ((4.1)) amongst all these combinations. As the maximum test subset order is increased, it becomes computationally infeasible to evaluate all test subset candidates. To evaluate for test subset orders up to 5, we limited the number of test subset candidates at the higher orders (4 and 5). Specifically, to find the order k test combinations, we considered the best W order $k-1$ test combinations and trial-added every remaining test to each of these W test subsets. Clearly, the computational requirements grow with W. In our experiments, we chose $W = 100$. It is worth noting that increasing W did not change the results much, which implies the chosen top W order $k-1$ test combinations give adequate search breadth.

In the approach proposed in Chapter 3 and in other AD methods with which we will compare, flow detections are made sequentially, one flow at a time. In other words, the statistical significance of each flow is assessed separately. But, in our method, we are evaluating the statistical significance of a cluster of flows, which results in anomalous cluster detections based on (4.1). Although it is not uncommon to observe singleton cluster detections (a cluster consisting of a single flow) based on the evaluation under our criterion, these singletons typically occur in the later detected clusters (after all or nearly all of the anomalous class flows are detected). The early detected clusters tend to consist of a group of flows that mainly consist of the anomalous class (*e.g.*, Zeus). Actually, it is observed that the first cluster may not necessarily include samples that belong to the anomalous class. However, shortly after such "outlier" clusters are detected, there is an early cluster that includes a majority of the Zeus samples.

4.3 Experiments

4.3.1 Methods of Comparison

These methods are used to compare and contrast the performance of p-value clustering.

- One-class SVM: It is a widely used kernel method for anomaly detection ([23], [109], and [47]). For one-class SVM experiments in this thesis, the LIBSVM software was used. There is the issue of selecting the appropriate value for the SVM hyperparameter, ν. In our anomaly detection setting, there is no validation set for selecting ν, so this can be a problem in practice. In order to assess the best-case (upper bound) performance for the one-class SVM, we selected the value for ν in the interval $(0, 1)$ that gives the best AUC performance on each data set tested. Hence, one-class SVM results illustrate performance upper bounds, because we optimistically chose ν. Figure 4.20 depicts the AUC performance sensitivity to the value of ν.

- Lookahead: This method is discussed in Chapter 3. To remind, this method makes individual sample detections, based on the results of a single test and uses lookahead to assess the effect of using a new test on the multiple testing-corrected significance (corrected p-value) for subsequent detections.

- P-value sum and p-value (log) sum: These are two other benchmark methods that are used for comparison. They sort the samples based on the sum of their p-values (log p-values) over all tests. Anomalous samples are expected to have small aggregated p-values. These are also used in Chapter 3.

- Decision Tree: C4.5 decision tree is used in some experiments to give an idea on the (average test set) supervised classification performance result for the corresponding file.

- Ensemble Methods: The methods that are making individual sample detections (Lookahead, p-value sum and p-value (log) sum) effectively based on the ranks that they assign to the samples in the batch. In fact, p-value clustering methods are also ranking the samples. The difference is that clustering methods treat the samples of each cluster equally, therefore they are practically assigning the same rank value to the samples of the same cluster. If the mean rank value of the samples is assigned to each sample in a cluster detected by the clustering methods (*e.g.*, if the first cluster consists of 5 samples, 3 will be assigned to each of these samples)

and these are summed with the ranks assigned by the other methods, then we can obtain an overall ranking of all the samples based on these resulting ranks. All or a subset of the methods can be used to obtain an ensemble result.

4.3.2 Results

In our experiments, several results are notable, regarding both comparisons between methods using the same feature representation and cross-comparisons between different feature representations. Both individual file results and summary of results corresponding to all files are provided for comprehensive and in-depth understanding. A summary of these results can be found in [57, 56].

4.3.2.1 Alternating feature representation results

We will present the results that are obtained by using alternating feature set that is described in Section 2.5.2. Figures 4.1-4.30 belong to this subsection. The discussion of the results of this subsection is provided below with different perspectives.

Effect of dataset on the performance: As seen in Figures 4.1, 4.3, and 4.4, performance is very much dataset-dependent. It is hard to provide intuitions solely based on these plots. There are a few factors in effect here.

1. *Port:* It can be seen from Figures 4.5 and 4.6 that the port that the flows are collected from is very important the performance. The reason is that port affects type and diversity of the traffic. It determines the characteristics of the web flows and therefore how the null is informed. For instance, datasets collected from port 15 are perform very well, possibly due to significant differences of the Web traffic collected from that port and Zeus traffic. And, also, these flows may have less diversity, making it sufficient to inform the null with less number of flows. In fact, the largest file among these 3 is experimented by reducing the training set size by 6 times and it is observed that the performance does not degrade with the reduced size.

2. *Dataset size:* This determines how well-informed the null is. And when the null is well-informed, then each feature better reflects the characteristics of the Web traffic. This will lead to better discrimination. It can be seen from Figures 4.5 and 4.6 that for most of the ports, the worst performing datasets are the smallest ones (shown in black), whereas the performance tends to improve when the dataset size (therefore training set size) increases (shown with redder dots).

3. *Time of day:* Taking into account file collection time of day explains some exceptions to the above two rules, *e.g.*, for port 25 datasets. There are 3 datasets collected from port 25 and the best performing dataset is the smallest of these. The reason might be that this dataset is collected early in the morning, which is different than the datasets collected in the afternoon. Early morning datasets tend to have less diversity compared to other datasets. So, time of day may also affect type and diversity of the traffic.

Overall performance of p-value dependent methods: Using this feature set to obtain p-values and using these for individual detections leads to successful results (p-value clustering, Lookahead, p-value sum, p-value (log) sum). The one-class SVM performs very poorly in the Zeus experiments, as can be seen in Figures 4.14, 4.16, and 4.18. It performs well in P2P experiment (Figure 4.19) like the other methods, but still one-class SVM is the worst of all. Since all of the methods are using the same feature representation, this indicates the statistical significance assessment power of p-values.

P-value clustering vs. Lookahead: P-value clustering outperforms Lookahead in almost all of the datasets, as seen in Figure 4.1. At this point, it is important to remember that Lookahead is a sample-wise detection approach.

P-value clustering vs. P-value sum and P-value (log) sum: Although, on the average, p-value clustering approaches lag p-value sum and p-value (log) sum in AUC performance as seen in Figure 4.2, clustering approaches tend to outperform p-value sum and p-value (log) sum for larger files. This might be due to the existence of well-informed features for large files (large training set), therefore making it possible for feature selection methods to obtain good discrimination. On the other hand, when the file size is small, this means features are not informed enough, so the collective usage of them (as in p-value sum and log sum) leads to better performance, instead of selectively using them. Since all of these use the same p-values, the difference underlines the importance of test (therefore feature) selection in anomaly detection, at least when alternating feature representation is used. It is worth mentioning that when we use TP rate in the first 40 detections for performance assessment, instead of AUC, the early detection performance of methods are evaluated. With this criterion, p-value clustering order 2 is the best of all methods on the average, which can be seen in Figure 4.8. This means that feature selection helps boosting the early detection performance, which can be crucial if the aim to detect as many anomalies as possible in a short time.

Effect of order increase in P-value clustering: For our p-value clustering method, increasing the maximum subset order affects the performance in different ways. There

is not a single type of effect of increasing (maximum test combination) order on the area under ROC performance. Some of the results depict that order increase from 2 to 3 degrades the performance, whereas the remaining results show that this change improves the performance. But, order increase from 3 to 5 leads to a degradation in most of the results. This is consistent with our earlier observation that the independent test assumption becomes poor as the maximum test subset order increases. Average effect of order increase in AUC performance can be seen in Figure 4.2.

Effect of variance lower bound: The existence of categorical features led us to impose a lower bound on the diagonal components (variances) of the covariance matrices of the bivariate GMM models. It is observed that AUC performance is only modestly sensitive to this choice. To show this slight dependence, results for different lower bound choices are provided for some of the files. Lower bound value that is used in each experiment is provided in the captions.

Effect of using larger dataset: As mentioned above, in Figures 4.5 and 4.6, it can observed that when the port number is kept constant, for larger datasets, the performance tends to get better. To further understand the effect of dataset size, the following experiment is performed on File 13. Firstly, File 13 is experimented as it is (Figure 4.29). Secondly, it is divided into 5 subsets, and experiments are done by using each of these subsets separately (Figure 4.30). So, in this way, the training set size in the latter case is 5 times smaller. Smaller training set size means poorly informed null. With smaller training set, only p-value sum improved, all other methods degraded. This points to the fact that performance of a selection od features with poorly informed null is not as good as the collective performance of them. Better informed null is suitable for using feature selection. Another experiment pointing to the importance of file size is the experiment done with the largest file, which is File 8 (Section 4.3 provides file sizes). It can be seen from Figure 4.25 that the performance is clearly better than all other files. The reason might be the positive effect of large file (therefore training set) size (in addition to the other effects, such as port and time of day). To further investigate, we combined all of the port 6 files in LBNL repository. There are 6 such files in total, whose names are provided in Section 4.3. 2 of these files (4 and 7) are used also used individually in the other experiments. Results of experiments that use the combined file are in Figures 4.26, 4.27, and 4.28. It can be seen by comparing these results with File 4 (Figure 4.21) and File 7 (Figure 4.24) that combined file p-value clustering results outperform both of the individual file results. In fact, as seen from Figure 4.5, p-value clustering order 5 performance of the combined file (shown with port 0 in the figure) is

better than the AUCs of all individual files (port 6 files) obtained with this method.

Supervised vs. Unsupervised detection: C4.5 decision tree results are shown as black dots in some of the figures. The complete decision tree results, including the ones that are not shown here, are provided in Section 2.6. It can be observed that datasets that have similar decision tree results may have quite different anomaly detection results. For example, Files 1, 2, and 8 have similar decision tree results, whereas very different anomaly detection performances. This is not unexpected, since supervised detection is much easier than unsupervised detection.

Zeus vs. P2P: Differences in the results suggest that anomalous behavior is very different in Zeus and P2P cases. In the P2P case, we see that anomalous behavior can be well-captured using either all or using only a small subset of the features. This suggests that all the features are largely discriminating between the Web and P2P classes. But, this is not completely valid for all Zeus datasets. Most of the Zeus experiments lag the P2P experiment result in performance, no matter what the compared method is. This means that discrimination between Zeus and Web classes is not as easy to accomplish as P2P case in anomaly detection setting.

Individual Methods vs. Ensemble: By using the detection order of methods as ranks assigned to the samples, ensemble approaches can be devised. These ranks from each method can be summed for each sample and a new ranking can be obtained. It is observed that ensemble methods never underperform all of the methods in use. Ensemble methods either achieve some performance in between the methods in use, or in some cases, they achieve better than all of the methods used. In the latter cases, it can be said that there is a consensus on the true positives but not on the false positives. On the average, ensemble approach improves the results. Performances of ensemble methods can be seen in Figures 4.1, 4.3, 4.4, 4.7, and 4.9.

Most of the performance comparisons that are provided above are based on area under ROC (AUC) performances of the methods. Other useful comparison metrics can be derived by paying attention to the early detection performance. For example, true positive rate in a certain number of first detections can be such a criterion, which is also used here (by using first 40 detections). Another criterion can be the true detection performance corresponding to a certain low false alarm rate, *e.g.*, 0.1, 0.2. There are some cases where the methods are significantly distinguishable from each other in early detection success, where they have pretty close success to each other when AUC is used to compare. This difference in early detection success between methods is noticable in Figures 4.21, 4.23, 4.26, 4.27, and 4.28. In all of these, p-value clustering is superior

to other methods with this criterion. More comprehensive results over all files for early detection performances are collected in Figures 4.7 and 4.8. A crucial issue here is that smaller datasets tend to have higher TP rate, which is expected since there are less number of Web flows competing with the Zeus flows to take place in the first 40 detections (remembering that 10-fold cross-validation is used, resulting in different number of Web flows in the test set for each file).

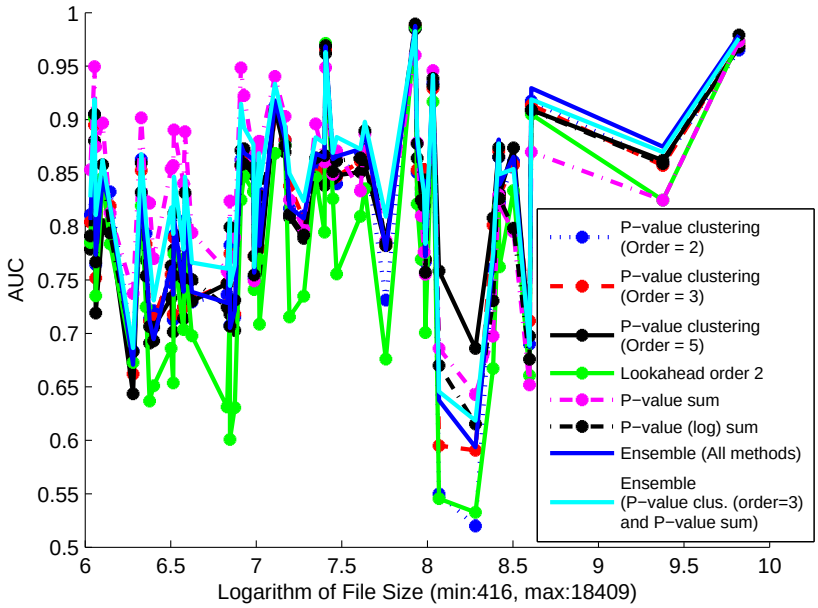

Figure 4.1. Area under ROC performances vs. file size for all files and all methods (alternating feature representation) (variance lower bound=1)

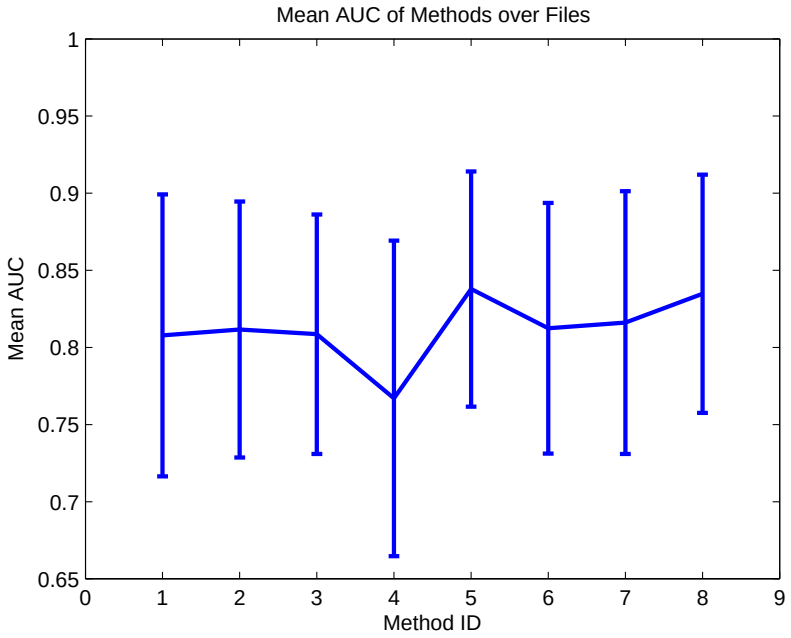

Figure 4.2. Mean area under ROC performances over all files for all methods (Method IDs: 1=p-value clus order 2, 2=p-value clus order 3, 3=p-value clus order 5, 4=lookahead, 5=p-value sum, 6=p-value log sum, 7=ensemble of all methods, 8=ensemble of p-value clus order 3 and p-value sum) (alternating feature representation) (variance lower bound=1)

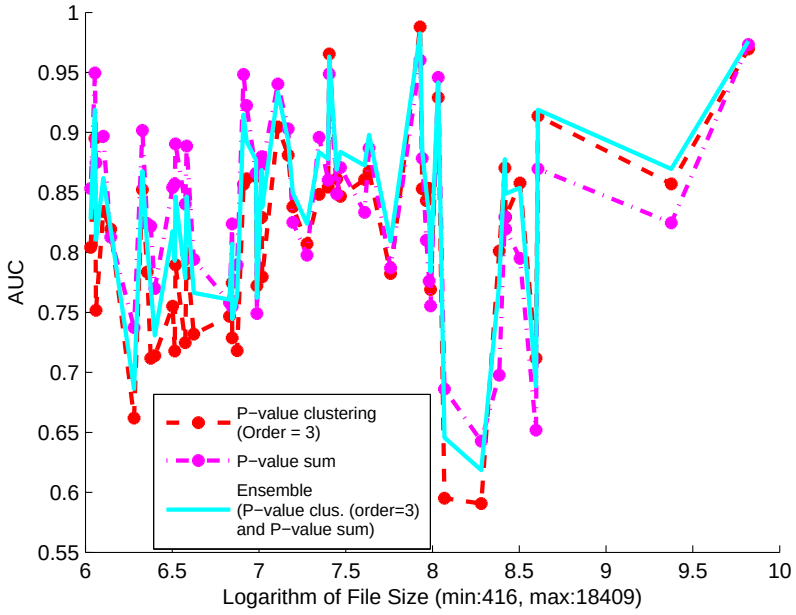

Figure 4.3. Area under ROC performances vs. file size for all files and 3 selected methods (alternating feature representation) (variance lower bound=1)

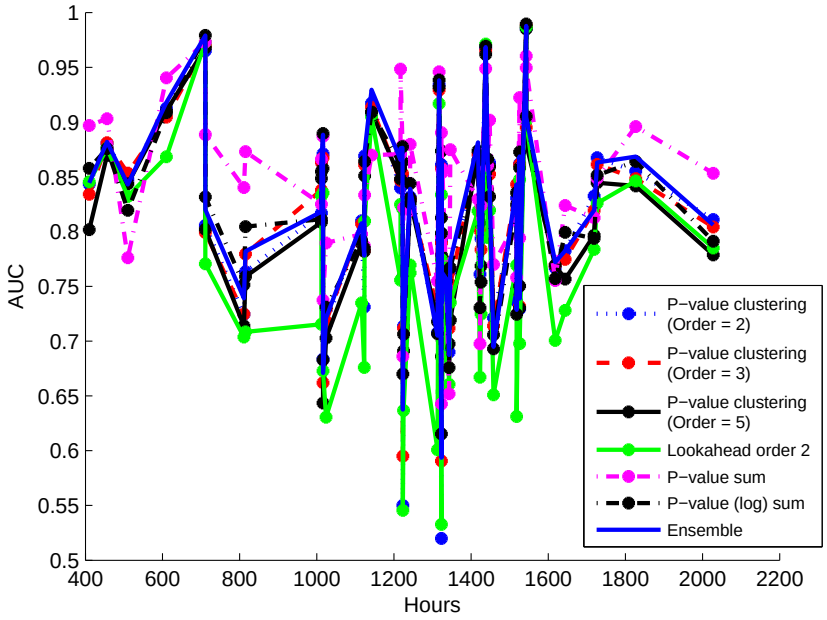

Figure 4.4. Area under ROC performances vs. hour of day for all files and all methods (hour of day range: 04:10-20:28) (alternating feature representation) (variance lower bound=1)

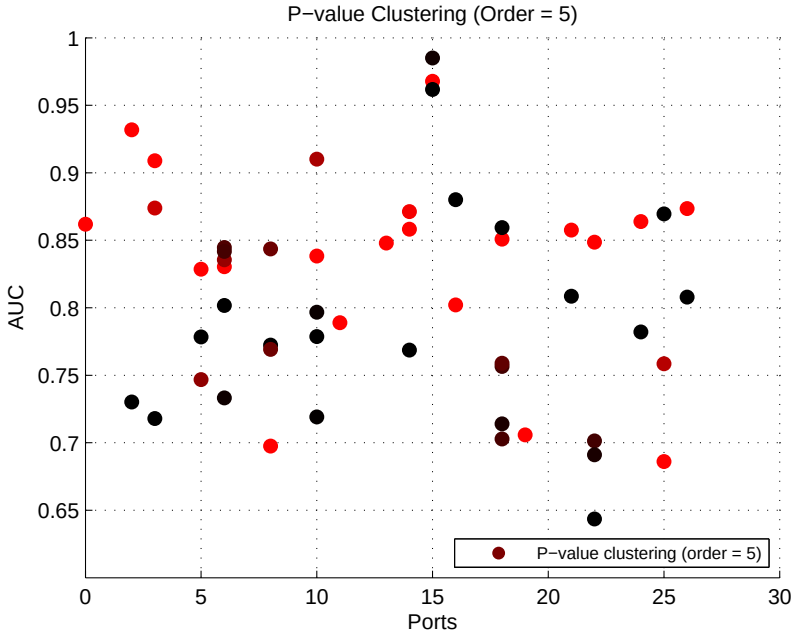

Figure 4.5. Port and dataset size dependence of area under ROC performance of p-value clustering (order 5) (sizes of the datasets corresponding to each port are color-coded: red shows the largest dataset, black shows the smallest dataset, other files are depicted with colors in between red and black) (alternating feature representation) (variance lower bound=1)

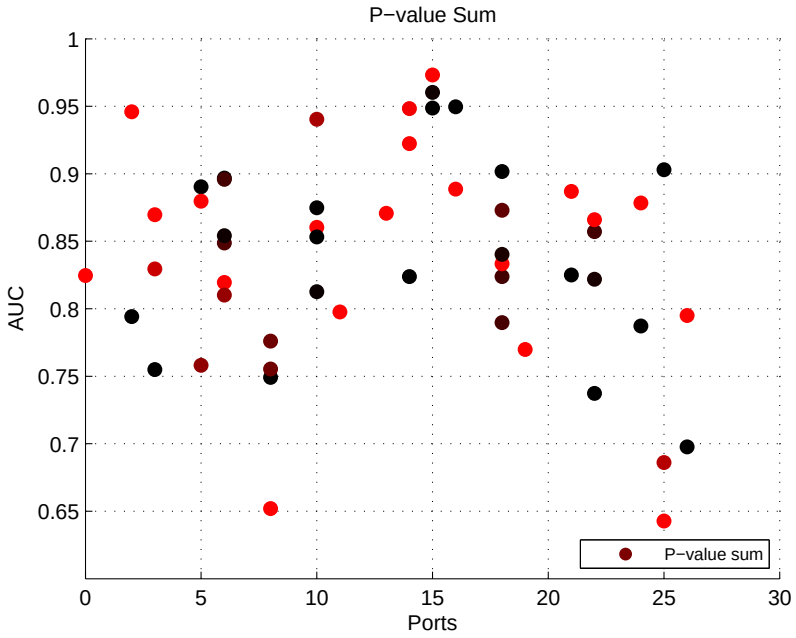

Figure 4.6. Port and dataset size dependence of area under ROC performance of p-value sum (dataset sizes corresponding to each port are color-coded: red shows the largest dataset, black shows the smallest dataset, other files are depicted with colors in between red and black) (alternating feature representation) (variance lower bound=1)

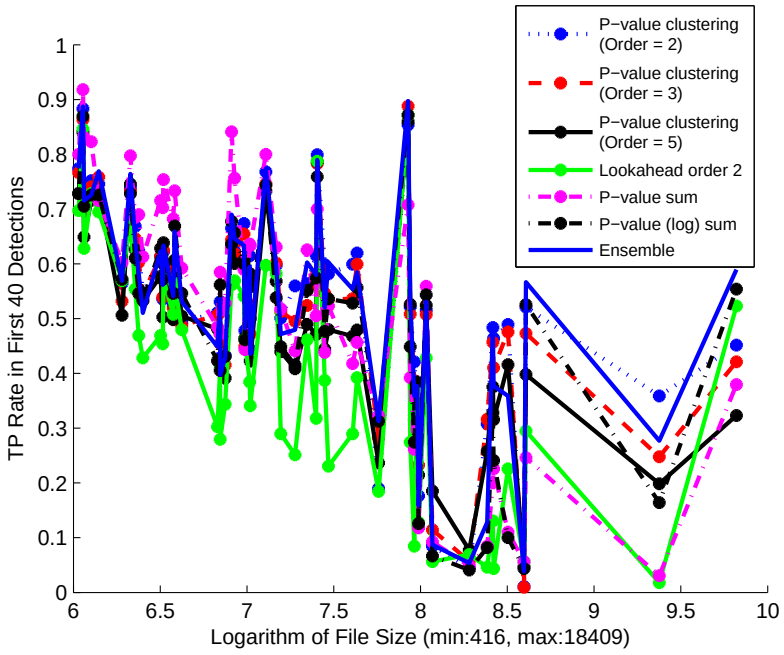

Figure 4.7. True positive rate in the first 40 detections for all files and all methods (alternating feature representation) (variance lower bound=1)

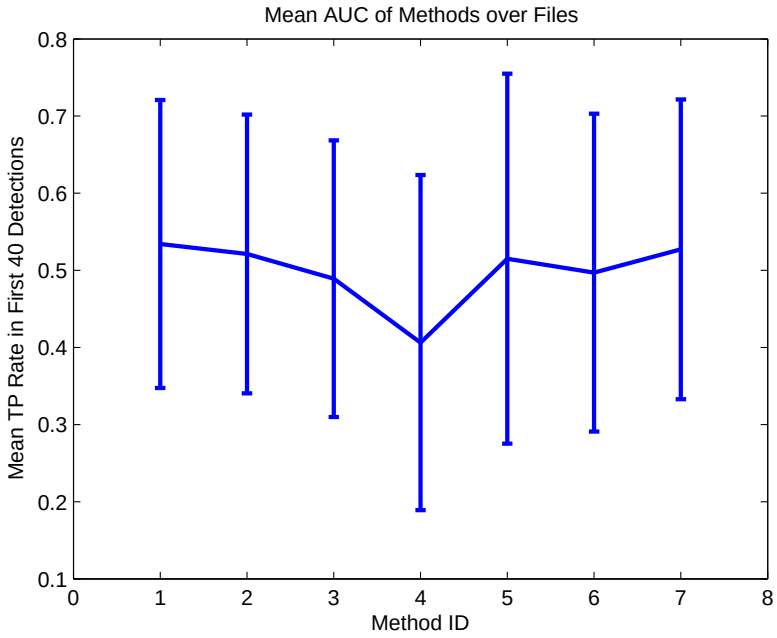

Figure 4.8. Mean true positive rate in the first 40 detections over all files for all methods (Method IDs: 1=p-value clus order 2, 2=p-value clus order 3, 3=p-value clus order 5, 4=lookahead, 5=p-value sum, 6=p-value log sum, 7=ensemble of all methods) (alternating feature representation) (variance lower bound=1)

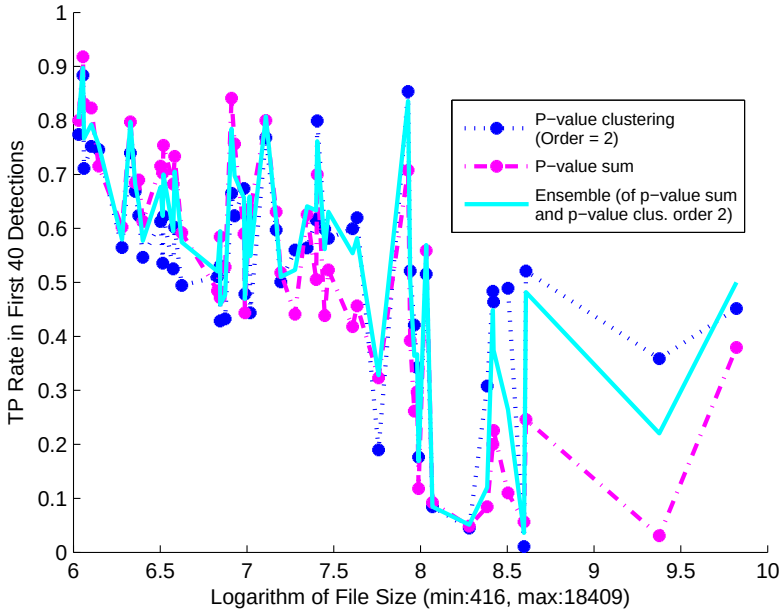

Figure 4.9. True positive rate in the first 40 detections for all files for 3 methods (alternating feature representation) (variance lower bound=1)

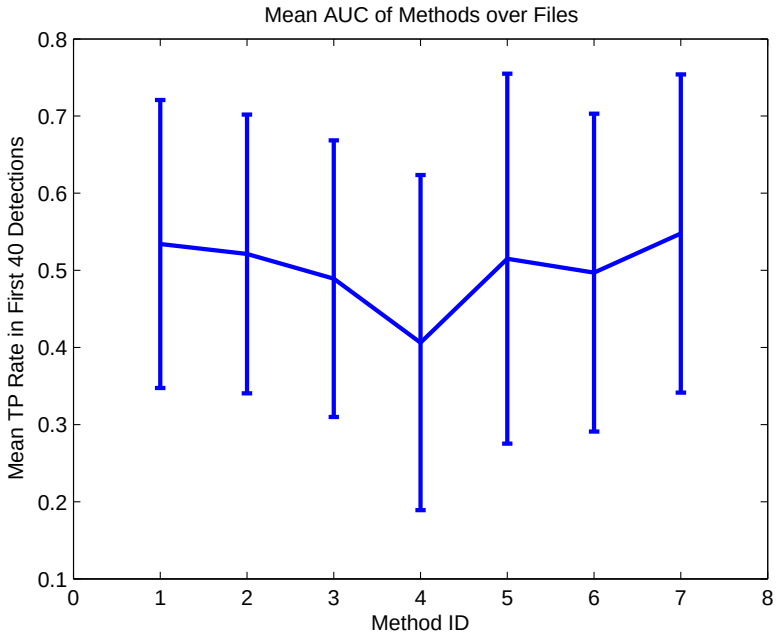

Figure 4.10. Mean true positive rate in the first 40 detections over all files for 3 methods (Method IDs: 1=p-value clus order 2, 2=p-value clus order 3, 3=p-value clus order 5, 4=lookahead, 5=p-value sum, 6=p-value log sum, 7=ensemble of p-value sum and p-value clus order 2) (alternating feature representation) (variance lower bound=1)

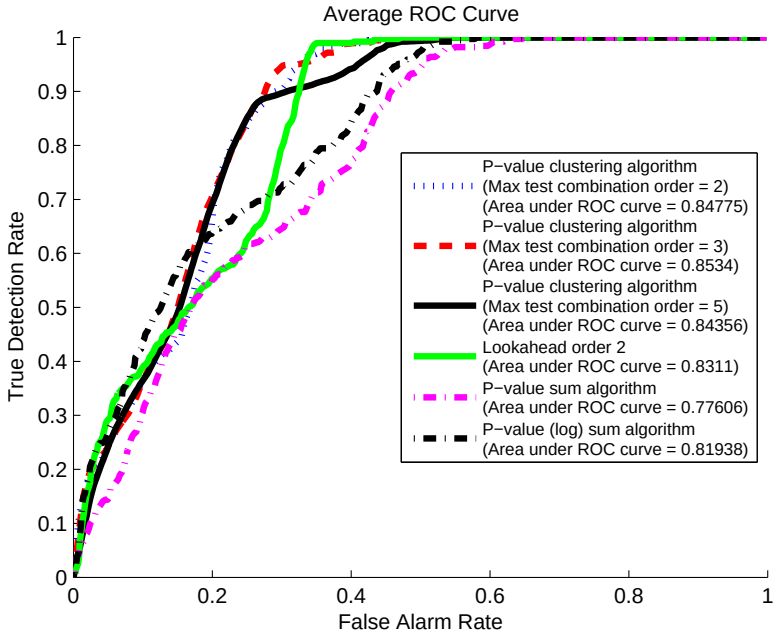

Figure 4.11. ROC curves (File 1 Web - Zeus) (alternating feature representation) (variance lower bound=1)

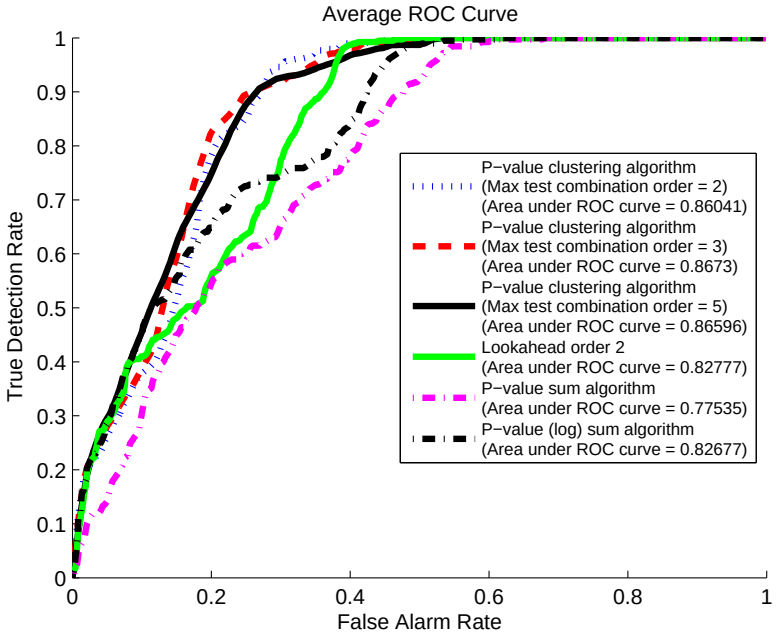

Figure 4.12. ROC curves (File 1 Web - Zeus) (alternating feature representation) (variance lower bound=3)

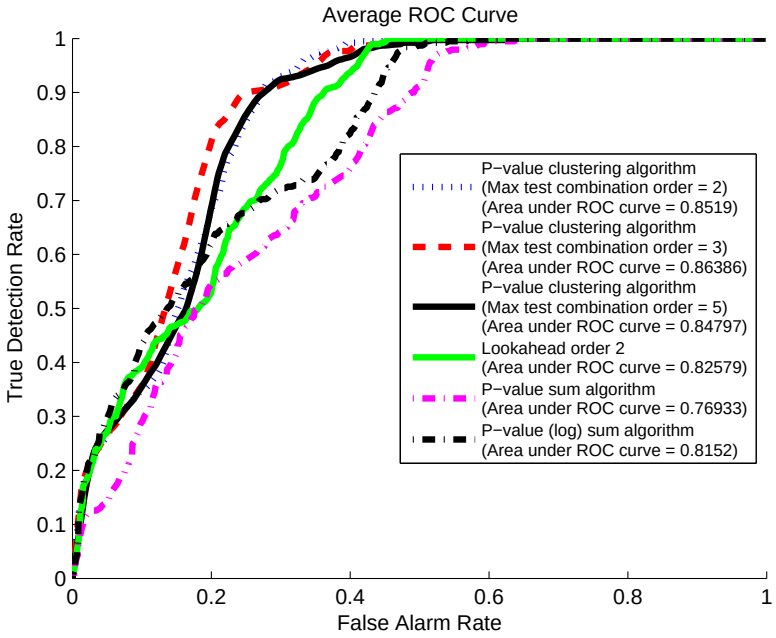

Figure 4.13. ROC curves (File 1 Web - Zeus) (alternating feature representation) (variance lower bound=5)

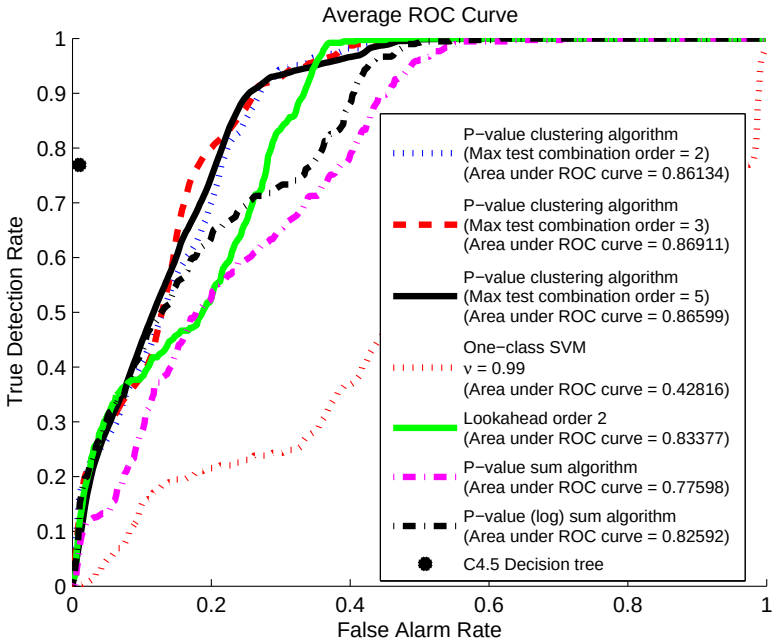

Figure 4.14. ROC curves (File 1 Web - Zeus) (alternating feature representation) (variance lower bound=10)

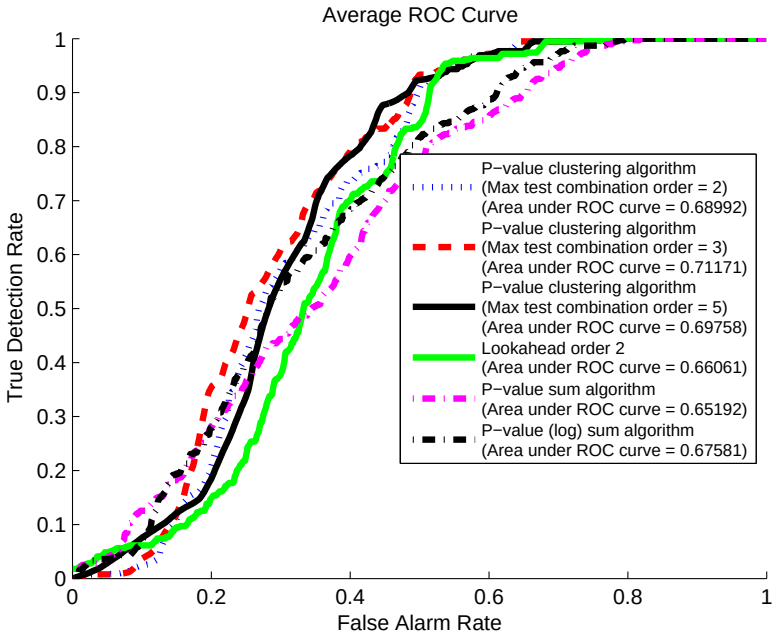

Figure 4.15. ROC curves (File 2 Web - Zeus) (alternating feature representation) (variance lower bound=1)

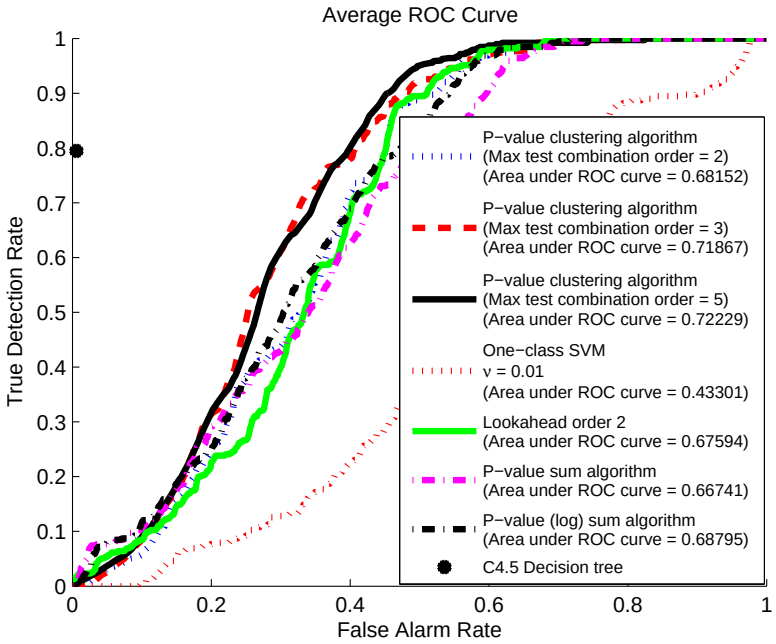

Figure 4.16. ROC curves (File 2 Web - Zeus) (alternating feature representation) (variance lower bound=10)

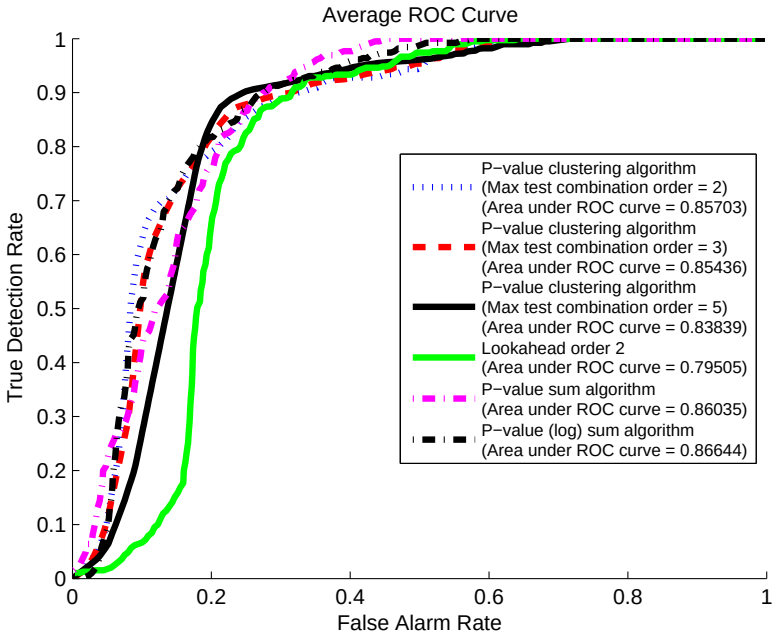

Figure 4.17. ROC curves (File 3 Web - Zeus) (alternating feature representation) (variance lower bound=1)

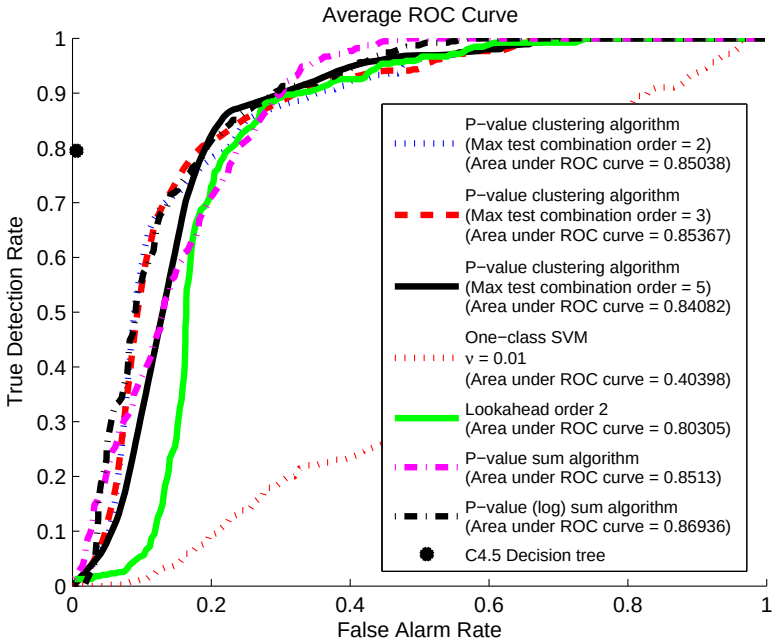

Figure 4.18. ROC curves (File 3 Web - Zeus) (alternating feature representation) (variance lower bound=10)

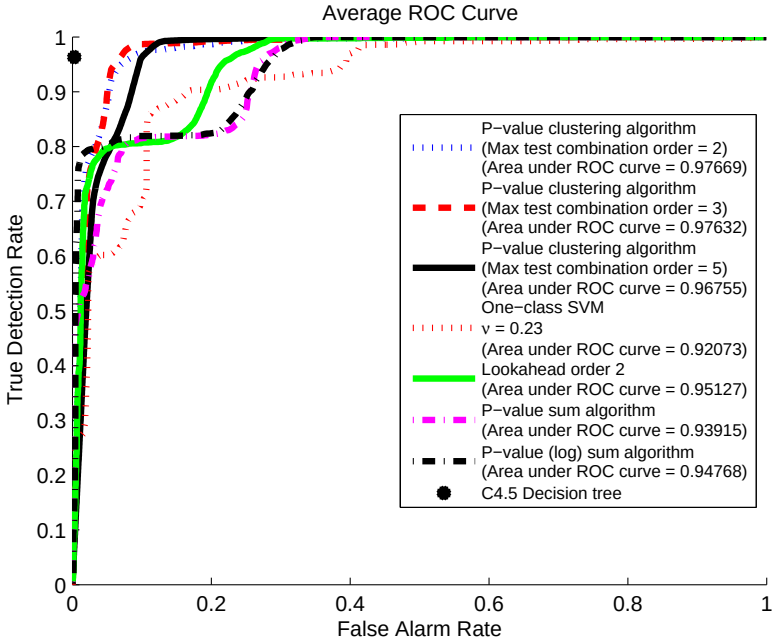

Figure 4.19. ROC curves (File 2 Web - File 2 P2P) (alternating feature representation) (variance lower bound=10)

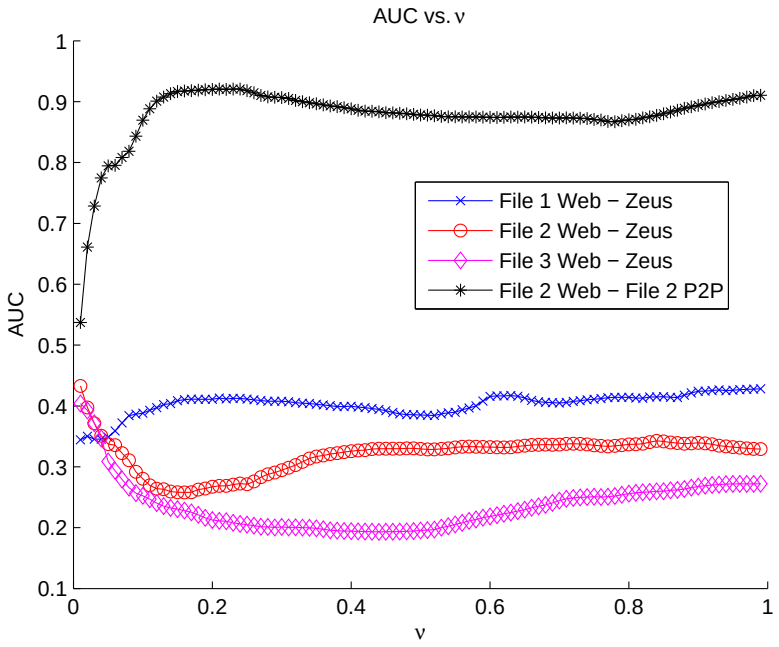

Figure 4.20. Sensitivity of AUC performance on ν parameter for the one-class SVM

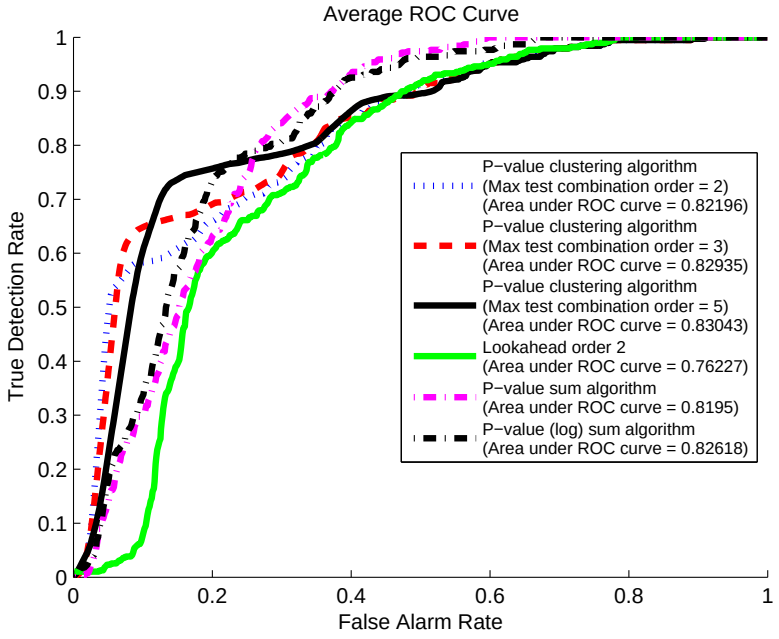

Figure 4.21. ROC curves (File 4 Web - Zeus) (alternating feature representation) (variance lower bound=1)

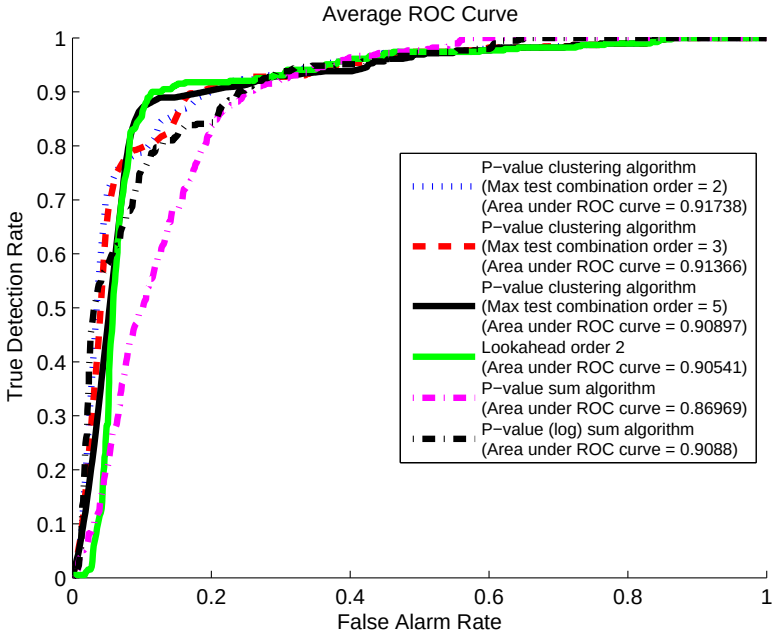

Figure 4.22. ROC curves (File 5 Web - Zeus) (alternating feature representation) (variance lower bound=1)

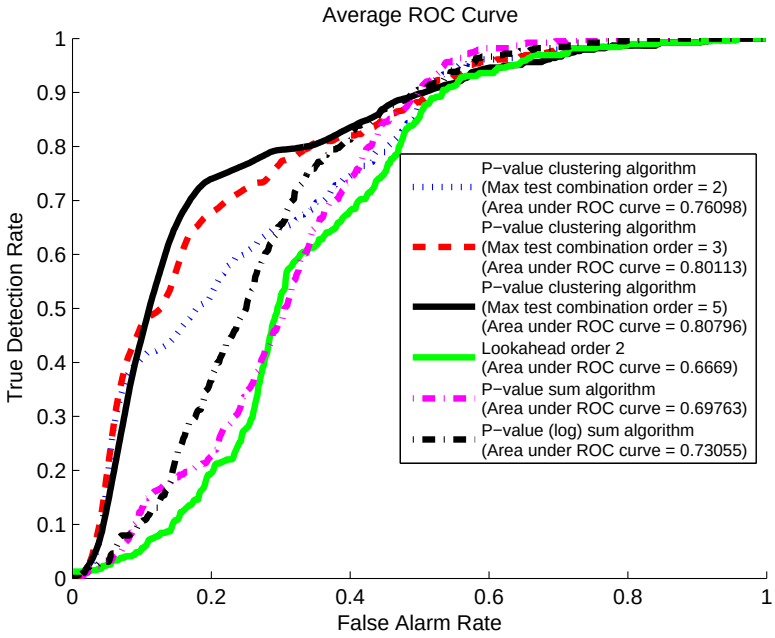

Figure 4.23. ROC curves (File 6 Web - Zeus) (alternating feature representation) (variance lower bound=1)

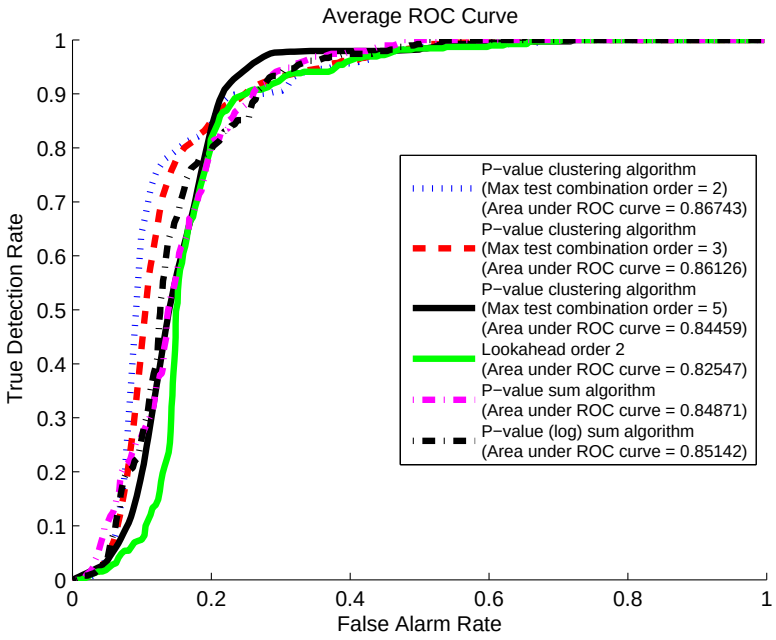

Figure 4.24. ROC curves (File 7 Web - Zeus) (alternating feature representation) (variance lower bound=1)

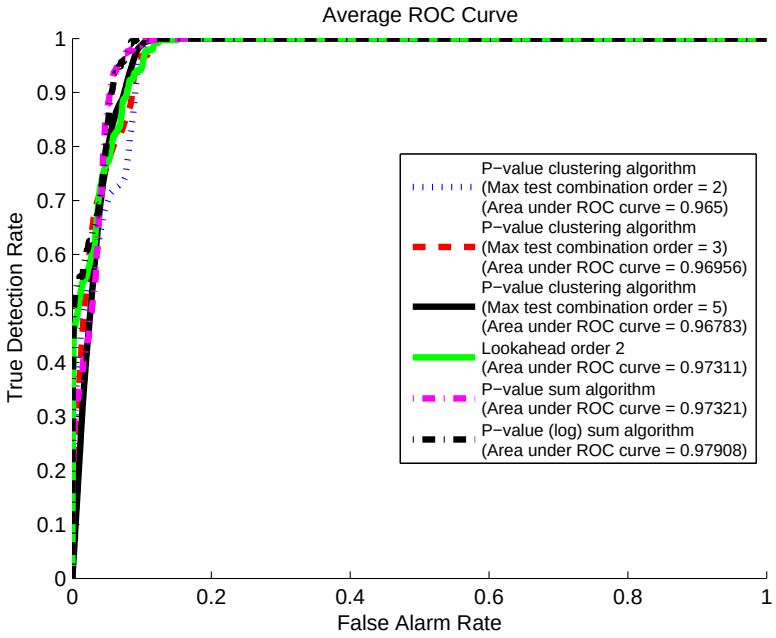

Figure 4.25. ROC curves (File 8 Web - Zeus) (alternating feature representation) (variance lower bound=1)

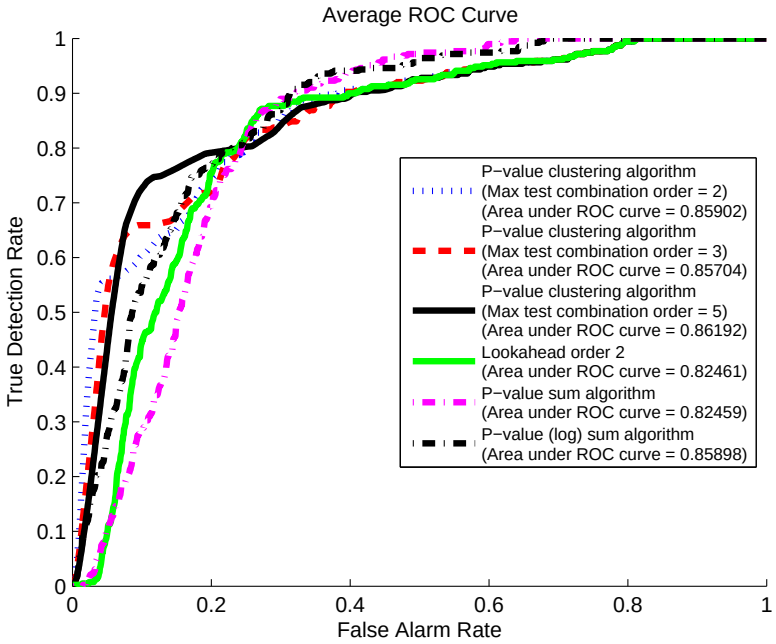

Figure 4.26. ROC curves (Combined File Web - Zeus) (alternating feature representation) (variance lower bound=1)

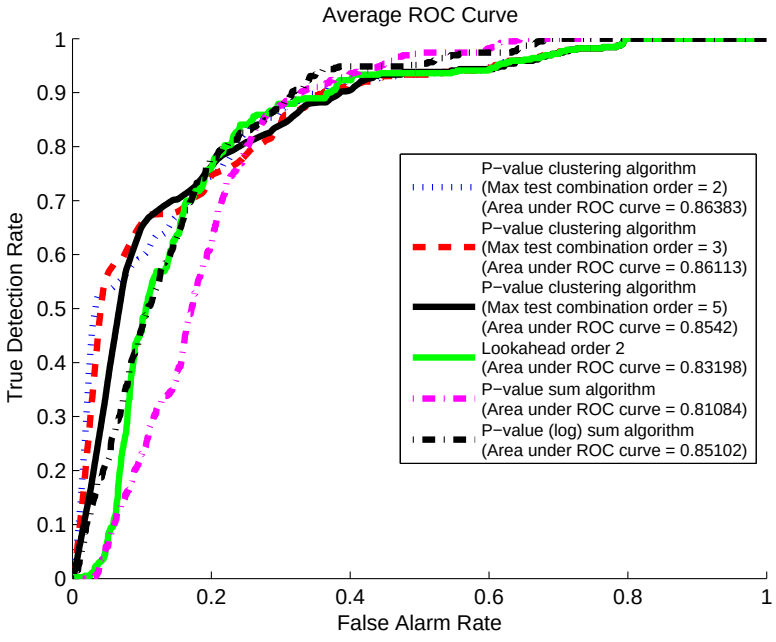

Figure 4.27. ROC curves (Combined File Web - Zeus) (alternating feature representation) (variance lower bound=5)

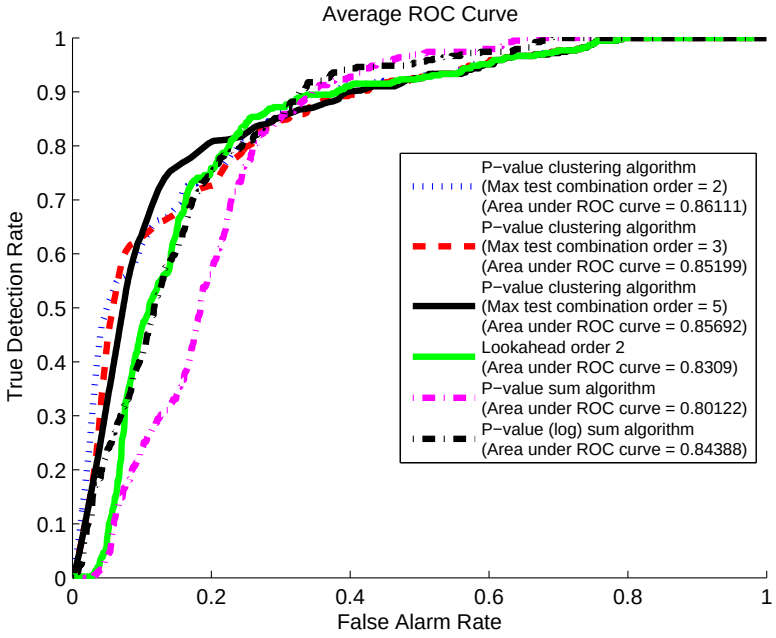

Figure 4.28. ROC curves (Combined File Web - Zeus) (alternating feature representation) (variance lower bound=10)

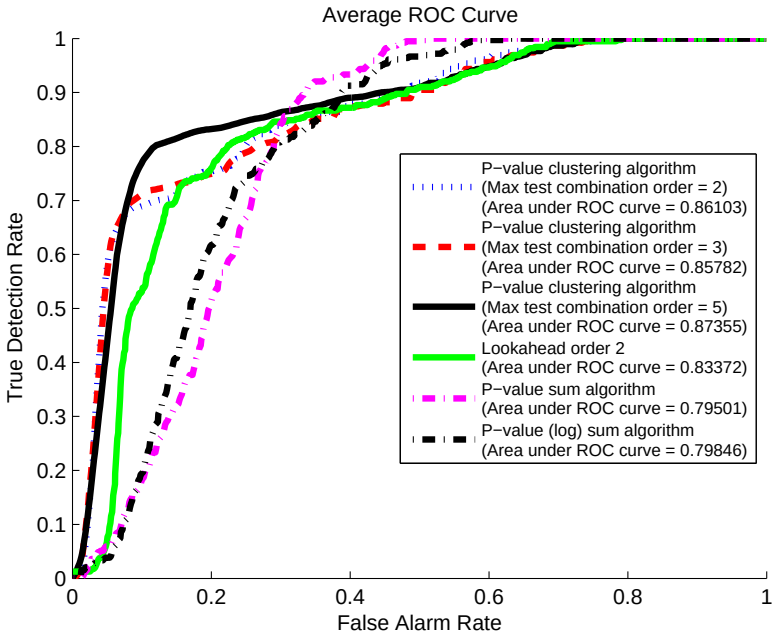

Figure 4.29. ROC curves (File 13 Web - Zeus) (alternating feature representation) (variance lower bound=1)

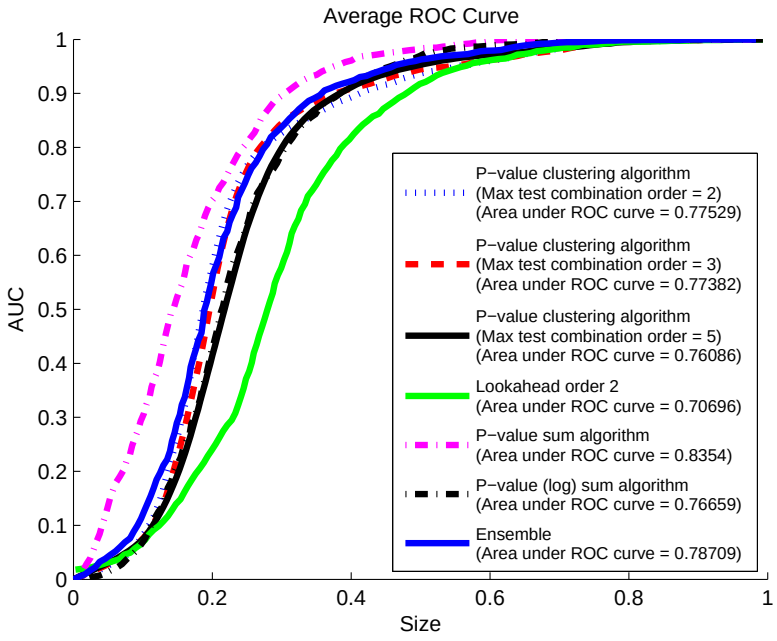

Figure 4.30. ROC curves (File 13 Web - Zeus) (File is divided into 5 subsets and the results are averaged) (alternating feature representation) (variance lower bound=1)

4.3.2.2 Alternating feature representation results - ACK packets modified

Results here show us the effects of having the same and different sizes for the ACK packets in Web and Zeus flows. It should be kept in mind that Zeus uses 40 as the ACK packet size in almost all of its flows. Choice of ACK packet size in Web flows depends on the file. Although 40 and 52 are seen in all datasets, the proportion of these values changes from file to file. We analyze 2 different scenarios here. In these transformations, the Zeus flows are not touched. The changes are made in only Web flows. These 2 different scenarios are mentioned in the captions of the figures as "same ACK packet size usage" (for 52 $\rightarrow$ 40 ACK packet size change in Web) and "different ACK packet size usage" (for 40 $\rightarrow$ 52 ACK packet size change in Web).

1) *Same-sized ACK packet usage:* The fundamental reason behind the difference in the size of the ACK packets is the usage of selective ACK (SACK). The scenario experimented here investigates the effects of not using SACK. This case is possible when SACK usage is not agreed by the client and server sides of the TCP communication during the 3-way handshake. For experimental purposes, we changed the ACK sizes of Web flows in a dataset. When we change all of the ACK packets sizes to 40 in Web flows, it means both Web and Zeus flows are forced to use the same packet size value for

the ACK packets. This removes the discriminative power of ACK packet values. The results related to this scenario are in Figures 4.31 - 4.36. The discussion of these results are provided below:

- Same-sized ACKs vs. Unmodified ACKs: If we compare these results with the corresponding ones (same file, same variance lower bound) in Section 4.3.2.1, we can see the effects on individual file results. It is intuitive to think that when Web and Zeus flows use the same ACK packet size, the performance will become worse. This is valid for Files 1 (Figure 4.31 vs. Figure 4.11), 2, and 3 (for orders 2 and 3). However, this is not the case for all of the results. For Files 3, 5, 6 (for order 5), and 7, the performance improved when the same ACK size is used. These imply that discriminative characteristics of our alternating feature set and p-value assessment algorithms do not exclusively rely on the different ACK packet size.

- P-value clustering vs. Other methods: As in the results without modification of ACK packets (Section 4.3.2.1), p-value clustering methods are superior to other approaches in some of the results, but not so for the other files.

- Effect of order increase in P-value clustering: For Files 1, 3, and 5, increasing p-value clustering maximum test combination order degrades the performance. Whereas, for Files 2 and 6, this increase makes a positive effect. For File 7, the performance peaks at order 3. So, there are more files for which order increase worsens the AUC performance.

2) *Different-sized ACK packet usage:* Unlike the previous scenario, changing all 40 packets to 52 has the effect of imposing a different packet size value for the ACK packets in Web and Zeus flows. This scenario is also possible to encounter, *e.g.*, when Web class uses SACK extensively and Zeus does not use it. The related results are in Figures 4.37 - 4.40.

- Different-sized ACKs vs. Unmodified ACKs: Since ACK packets are used frequently in TCP, having different packet sizes for the ACK packets will boost the performance. This can be observed in all of the results of this section.

- P-value clustering vs. Other methods: In contrary to the results with unmodified ACKs (Section 4.3.2.1), p-value sum and p-value (log) sum methods are slightly better than p-value clustering algorithms. In other words, the methods that exploit the diversity of using all of the tests perform slightly better than methods that

apply test selection. This might be a result of ACK packets occuring at multiple locations in the 20-dimensional (alternating) feature set with different packet sizes (40 and 52) for Web and Zeus, thus enhancing the discriminative power of all features (therefore tests). Under this scenario, it appears that all of the features are highly discriminative, both individually and collectively.

- Effect of order increase in P-value clustering: When the order is increased, especially from 3 to 5, there is a minor tendency to have a worse performance. But, still, AUC is very high for all orders, so that this may be considered negligible.

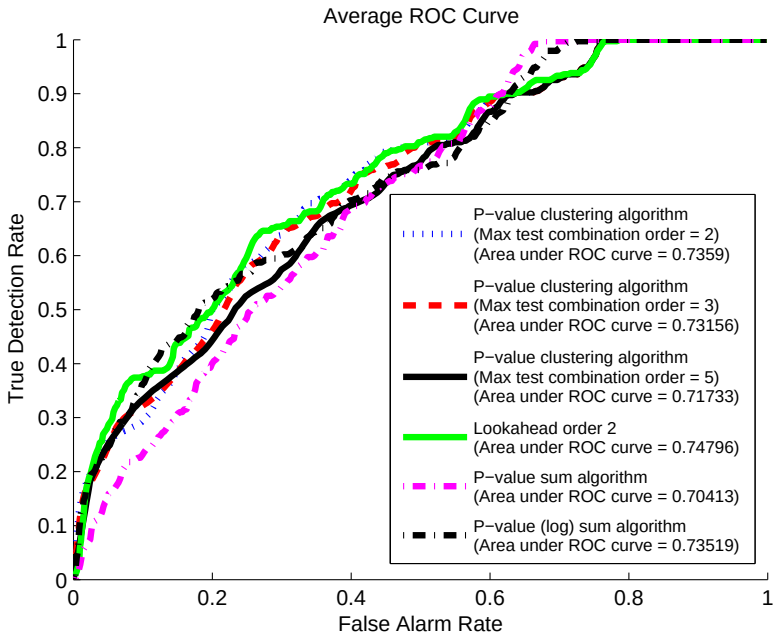

Figure 4.31. ROC curves (File 1 Web - Zeus) (alternating feature representation) (same-sized ACK packet usage) (variance lower bound=1)

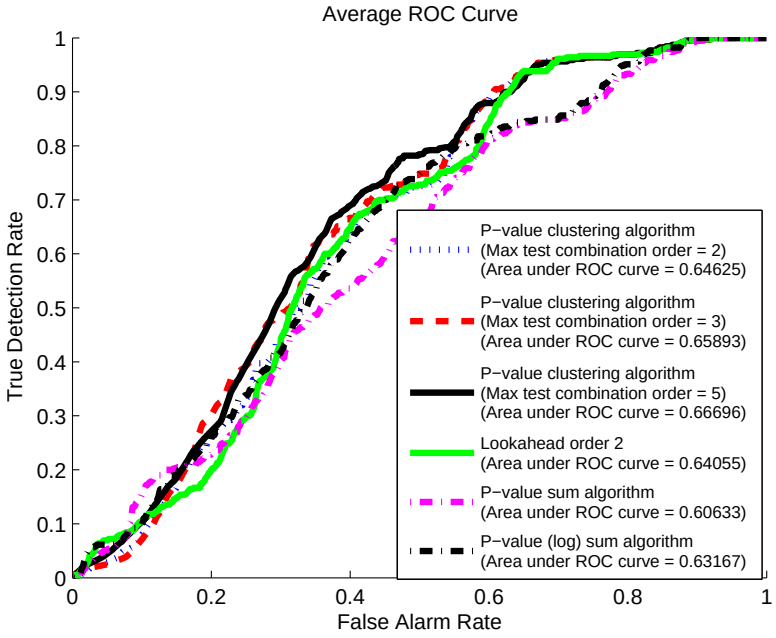

Figure 4.32. ROC curves (File 2 Web - Zeus) (alternating feature representation) (same-sized ACK packet usage) (variance lower bound=1)

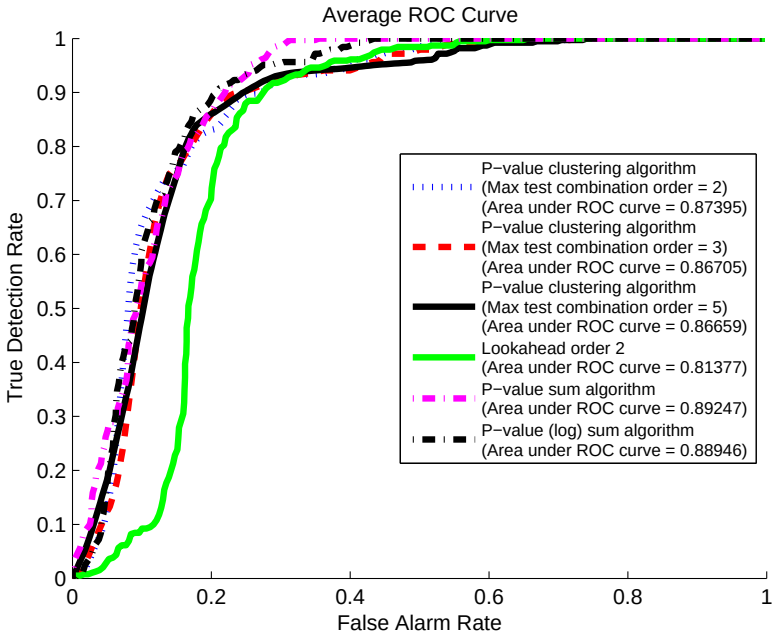

Figure 4.33. ROC curves (File 3 Web - Zeus) (alternating feature representation) (same-sized ACK packet usage) (variance lower bound=1)

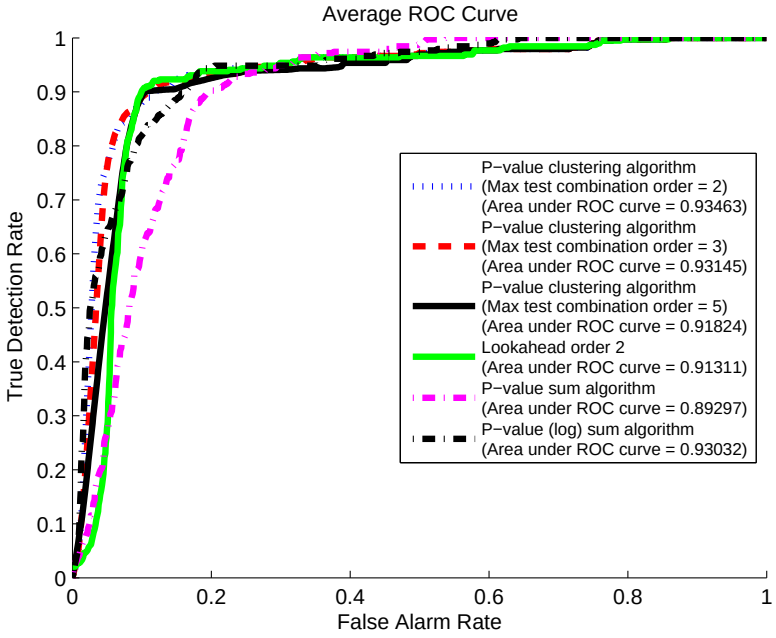

Figure 4.34. ROC curves (File 5 Web - Zeus) (alternating feature representation) (same-sized ACK packet usage) (variance lower bound=1)

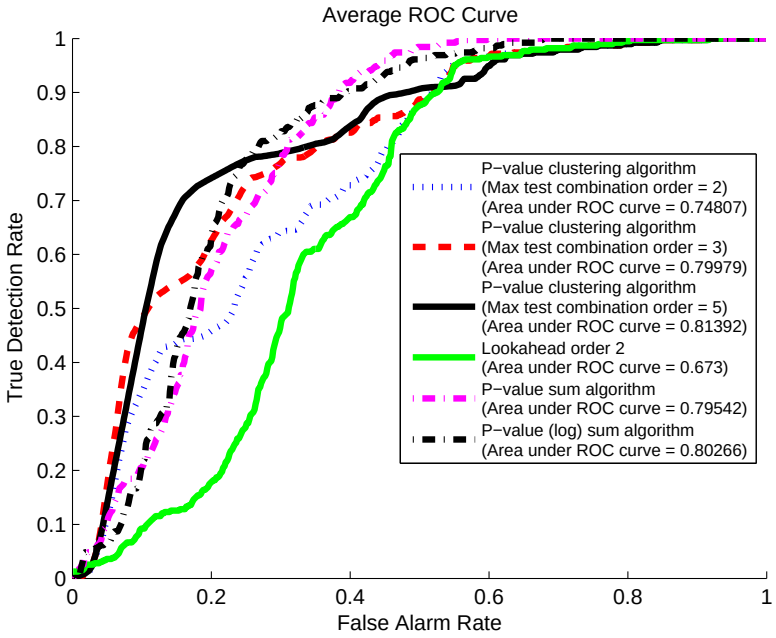

Figure 4.35. ROC curves (File 6 Web - Zeus) (alternating feature representation) (same-sized ACK packet usage) (variance lower bound=1)

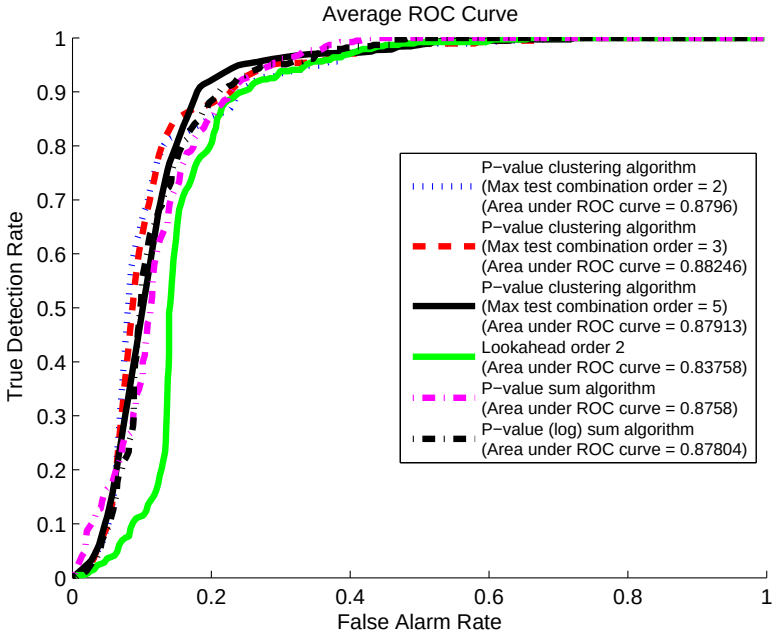

Figure 4.36. ROC curves (File 7 Web - Zeus) (alternating feature representation) (same-sized ACK packet usage) (variance lower bound=1)

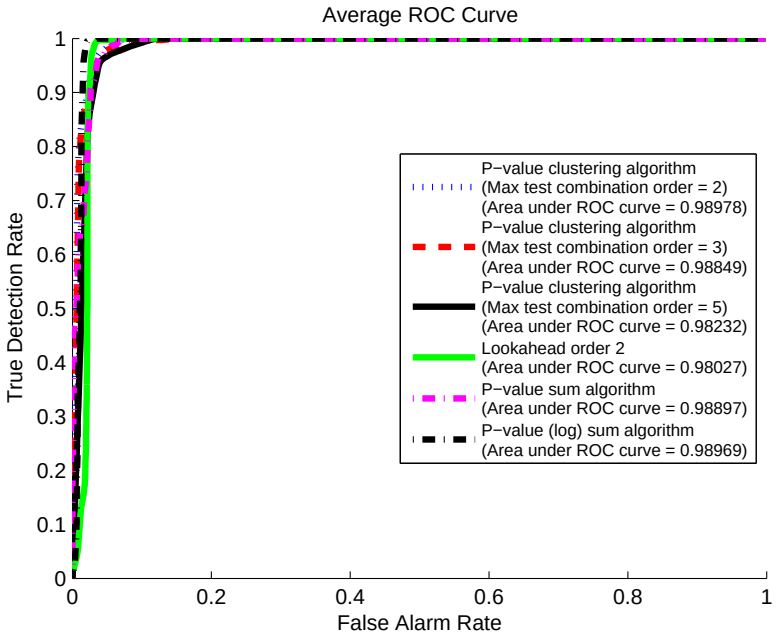

Figure 4.37. ROC curves (File 1 Web - Zeus) (alternating feature representation) (different-sized ACK packet usage) (variance lower bound=1)

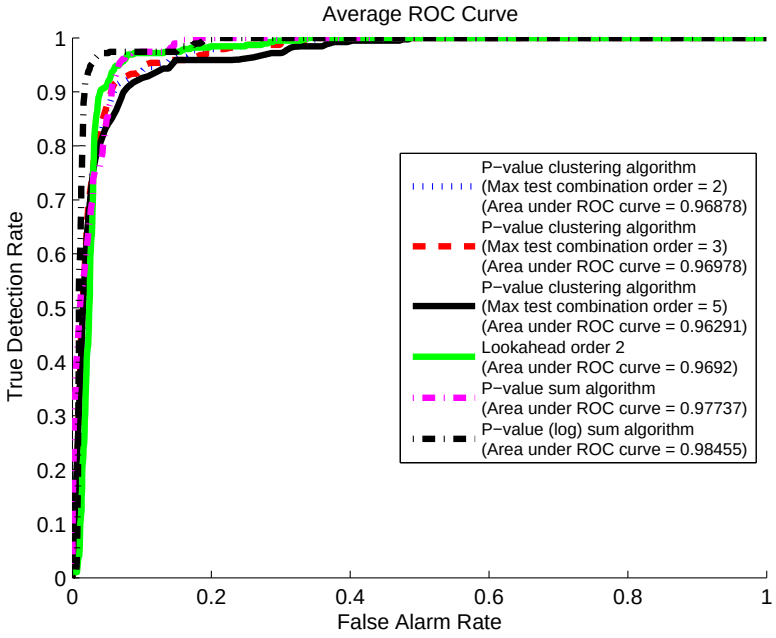

Figure 4.38. ROC curves (File 2 Web - Zeus) (alternating feature representation) (different-sized ACK packet usage) (variance lower bound=1)

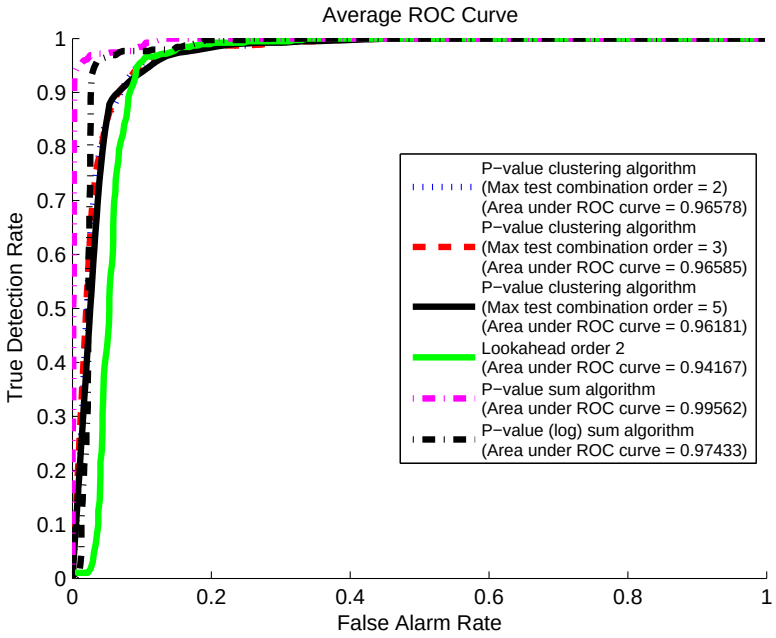

Figure 4.39. ROC curves (File 3 Web - Zeus) (alternating feature representation) (different-sized ACK packet usage) (variance lower bound=1)

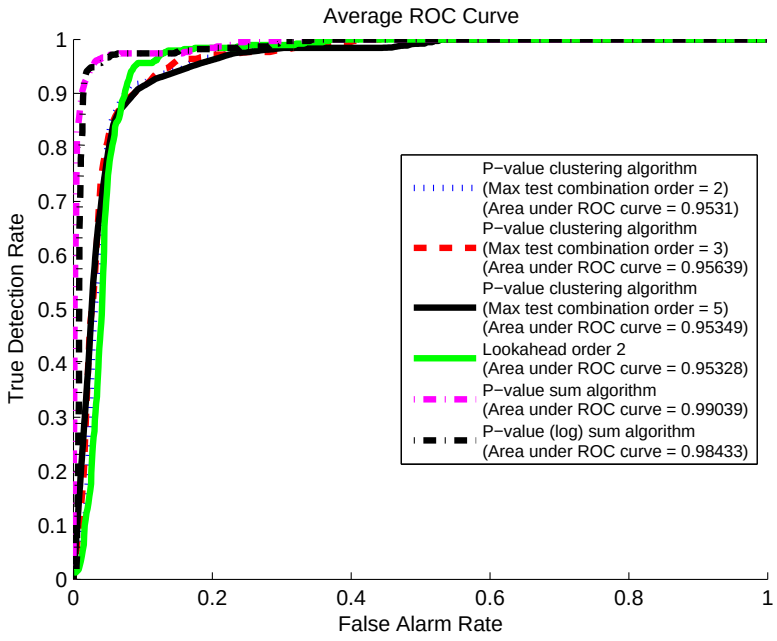

Figure 4.40. ROC curves (File 6 Web - Zeus) (alternating feature representation) (different-sized ACK packet usage) (variance lower bound=1)

4.3.2.3 Lossless feature representation results

Here, the results that are obtained by using lossless feature set that is described in Section 2.5.1 are presented. Figures 4.41-4.43 belong to this subsection.

P-value clustering vs. Other methods: P-value clustering results lag others in AUC performance. P-value sum and/or (log)sum methods perform the best in this feature representation.

Effect of order increase in P-value clustering: Order increase has a degrading effect on the p-value clustering performance.

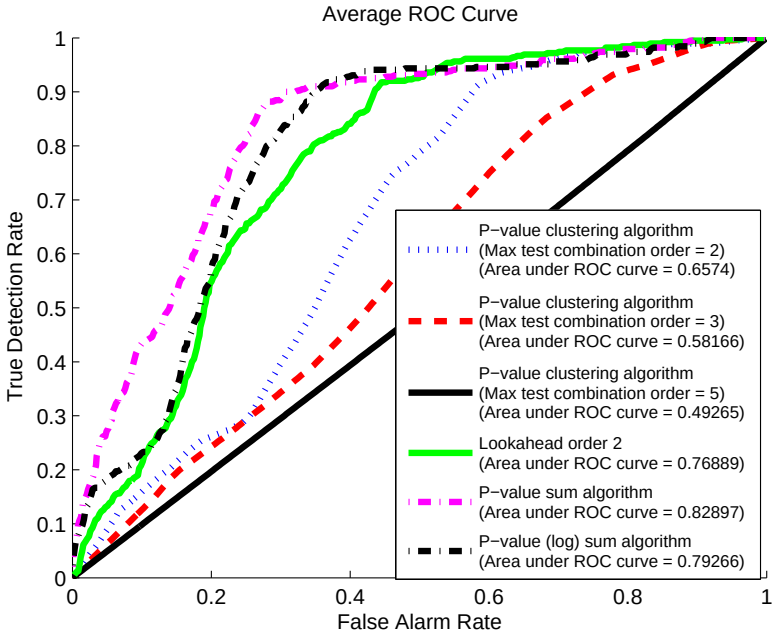

Figure 4.41. ROC curves (File 1 Web - Zeus) (Lossless feature representation) (variance lower bound=1)

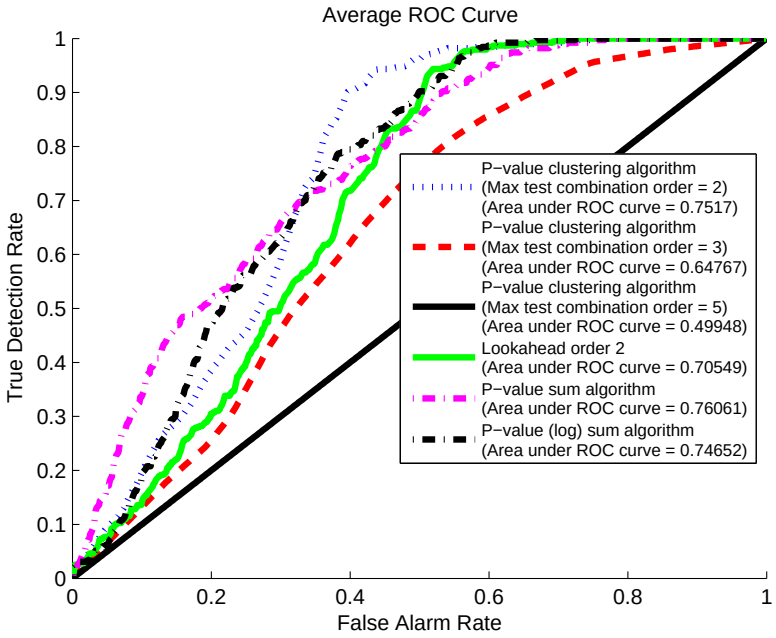

Figure 4.42. ROC curves (File 2 Web - Zeus) (Lossless feature representation) (variance lower bound=1)

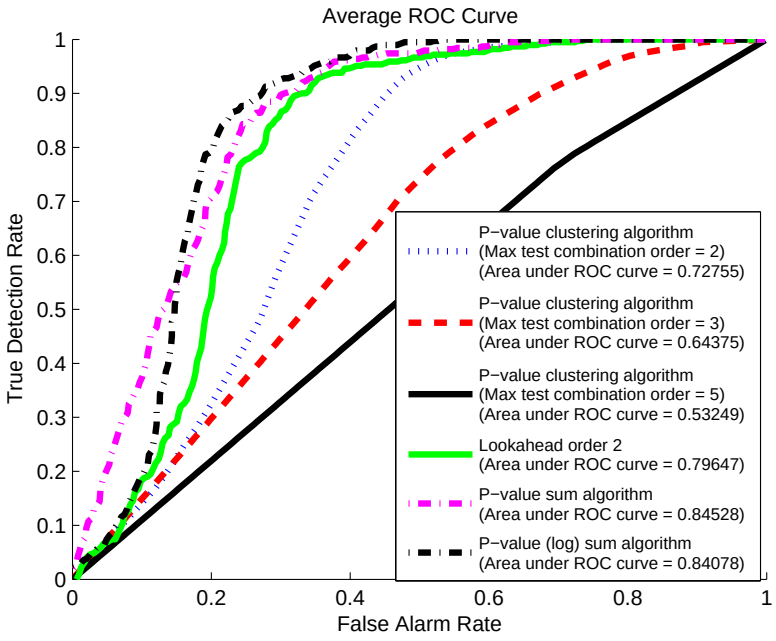

Figure 4.43. ROC curves (File 3 Web - Zeus) (Lossless feature representation) (variance lower bound=1)

4.3.2.4 Alternating feature representation results - with categorical feature 0

The results presented here are obtained by using the feature set that is described in Section 2.5.3. To remind, this feature set uses the alternating feature representation approach, but treats 0's as categorical features and uses them as conditioning context. The related results are in Figures 4.44 - 4.48.

P-value clustering vs. Other methods: P-value clustering with order 2 is better than p-value sum and p-value (log) sum algorithms in almost all results. Either Lookahead or p-value clustering with order 2 is the best performing method among all.

Effect of order increase in P-value clustering: In all of the results, this increase strictly degrades the AUC performance.

Considering the sharp deterioration in p-value clustering due to increasing test combination order together with the good performance of Lookahead against p-value sum (and log sum) underlines the importance of test selection in this feature representation and conditioning context. The bad ROC performance of p-value clustering with order 5 is because the detected clusters are becoming so large that even the existence of many anomalous (Zeus) flows in the early clusters is far from bringing a discrimination success

since high number of normal (Web) flows are present in all of the clusters, including the early and important clusters. Obviously, having large-sizes clusters means there are small number of them (for the same test set size). This can be observed from having smoother ROC plots as the order is increased, since each cluster detection means proceeding in the ROC line proportional to the size of the cluster.

In this feature representation and conditioning context, the high p-values are not seen as frequent as the alternating feature representation without conditioning. The reason of this is introducing the probabilities of each subset (resulting from conditioning) to the calculation of p-values (Section 2.5.3). This makes the p-values smaller, which in turn leads to larger clusters. The clusters get even larger when the order is increased. The reasons of both phenomena are explained in Section 4.1.

The next subsection (Section 4.3.2.5) gives results regarding to the modeling where ACK packets are also taken into account as categorical features, in addition to 0's.

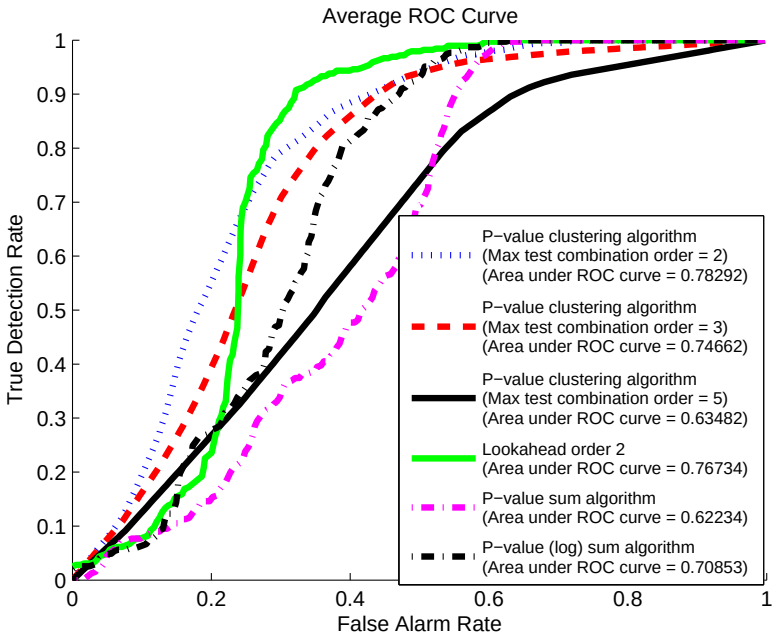

Figure 4.44. ROC curves (File 3 Web - Zeus) (alternating feature representation) (categorical feature 0) (variance lower bound=1)

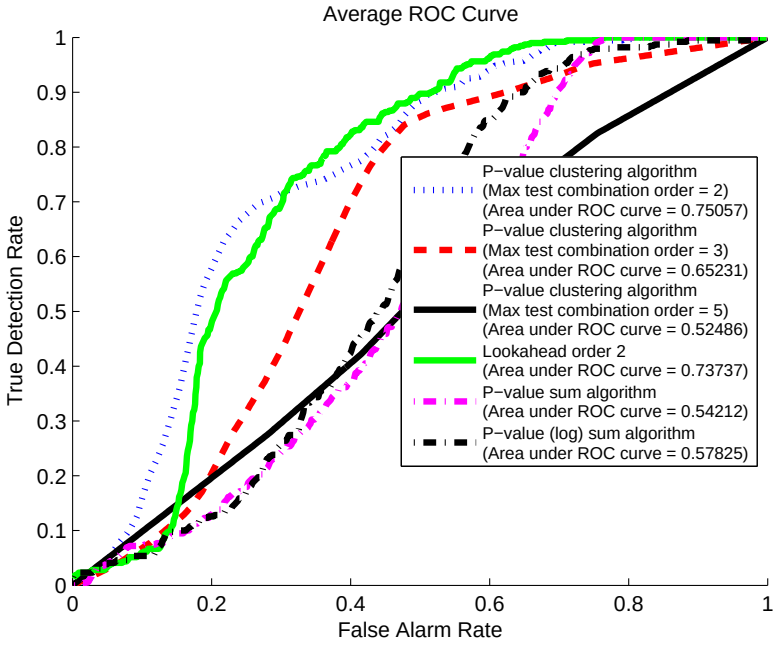

Figure 4.45. ROC curves (File 4 Web - Zeus) (alternating feature representation) (categorical feature 0) (variance lower bound=1)

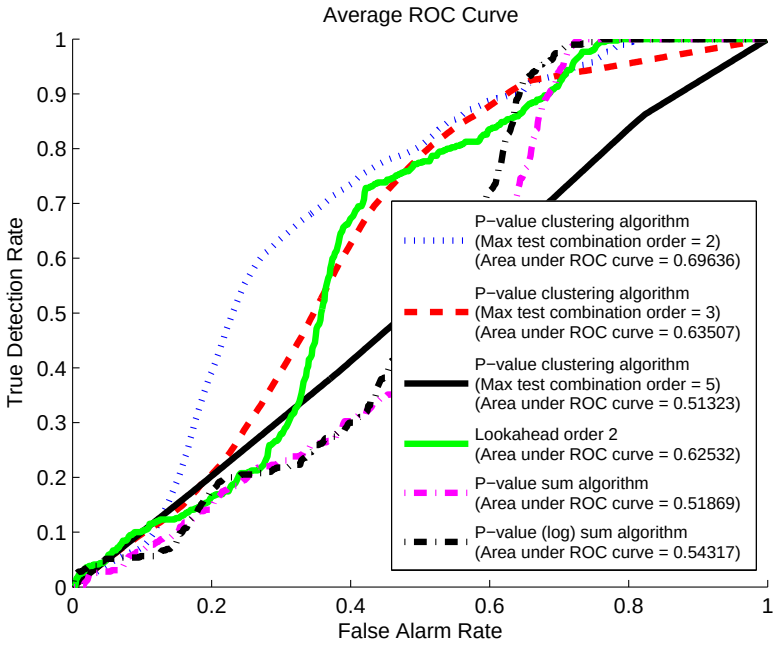

Figure 4.46. ROC curves (File 6 Web - Zeus) (alternating feature representation) (categorical feature 0) (variance lower bound=1)

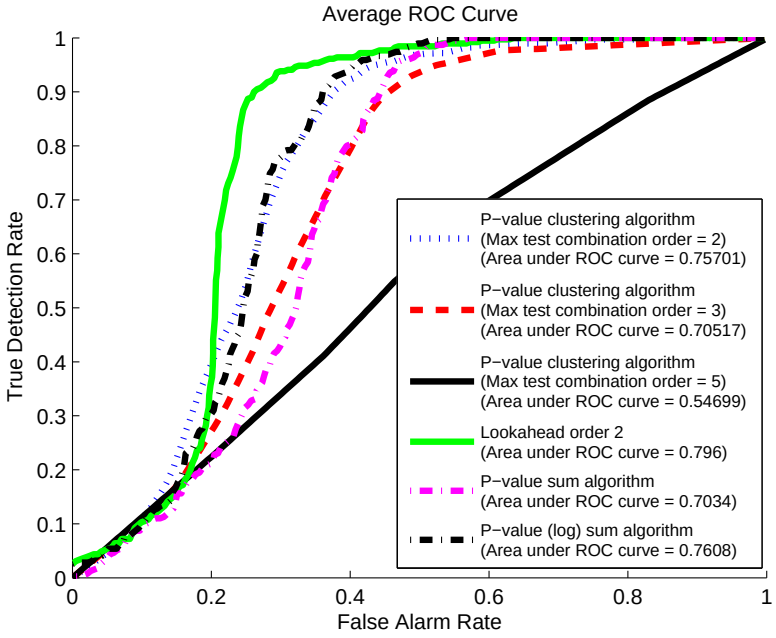

Figure 4.47. ROC curves (File 7 Web - Zeus) (alternating feature representation) (categorical feature 0) (variance lower bound=1)

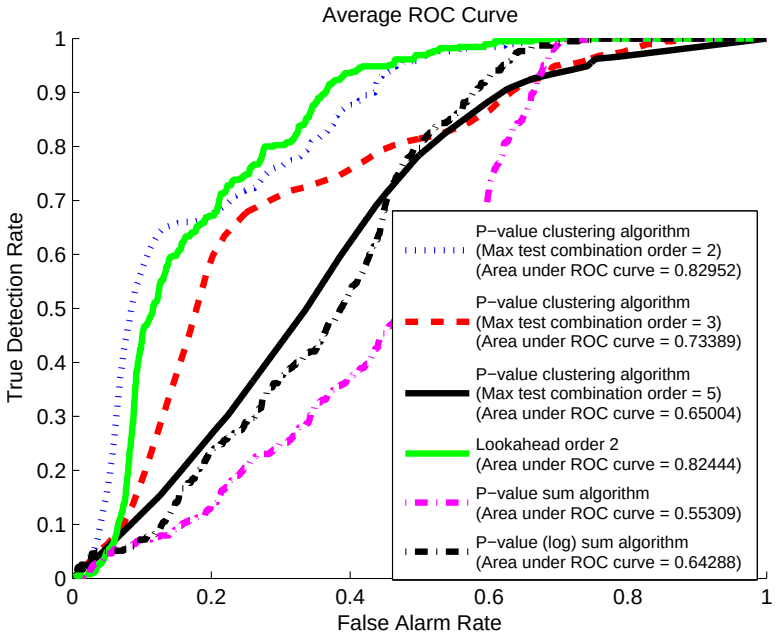

Figure 4.48. ROC curves (Combined File Web - Zeus) (alternating feature representation) (categorical feature 0) (variance lower bound=1)

4.3.2.5 Alternating feature representation results - with categorical features 0 and ACK (together)

These results are obtained by using the feature representation defined in Section 2.5.4. This feature set uses the alternating feature representation approach, but treats 0's *and* ACK packets as categorical features and uses them as conditioning context. The related results are in Figures 4.49 - 4.54.

Comparison of methods: As in the previous set of results (Section 4.3.2.4), p-value clustering with order 2 is superior to p-value sum and p-value (log) sum algorithms in almost all results. Lookahead is the best performing method in almost all of the files. The only exception to both of these is the File 8 result, shown in Figure 4.52.

Effect of order increase in P-value clustering: In all of the results, this increase strictly degrades the AUC performance, as in Section 4.3.2.4 results.

Categorical features 0 and ACK vs. Only categorical feature 0: Comparing the results presented in this subsection with the results in Section 4.3.2.4 (where only categorical features are 0's) reveals that p-value clustering performance is adversely affected by including ACK packets and 0's into the single category.

This latter observation leads us to treat 0's and ACK's as different types of categorical features. The effect of this is investigated in Section 4.3.2.6.

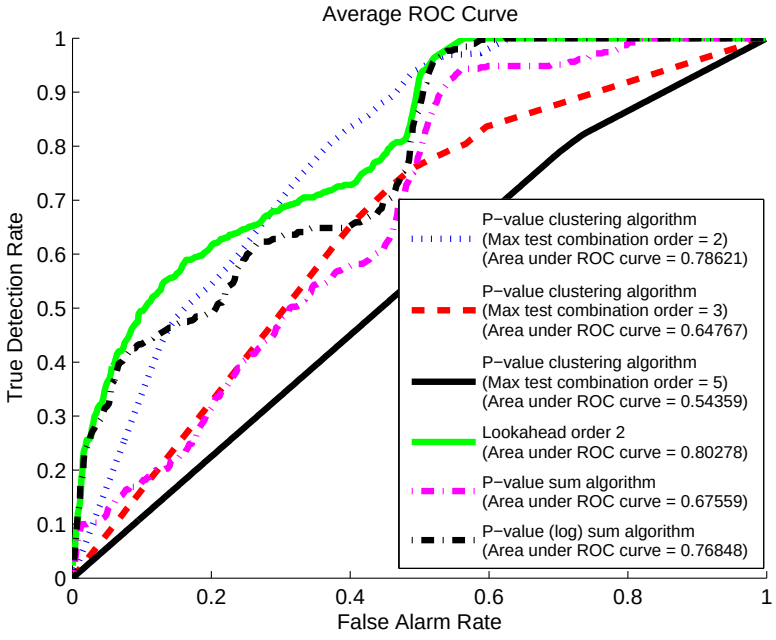

Figure 4.49. ROC curves (File 1 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the same category) (variance lower bound=1)

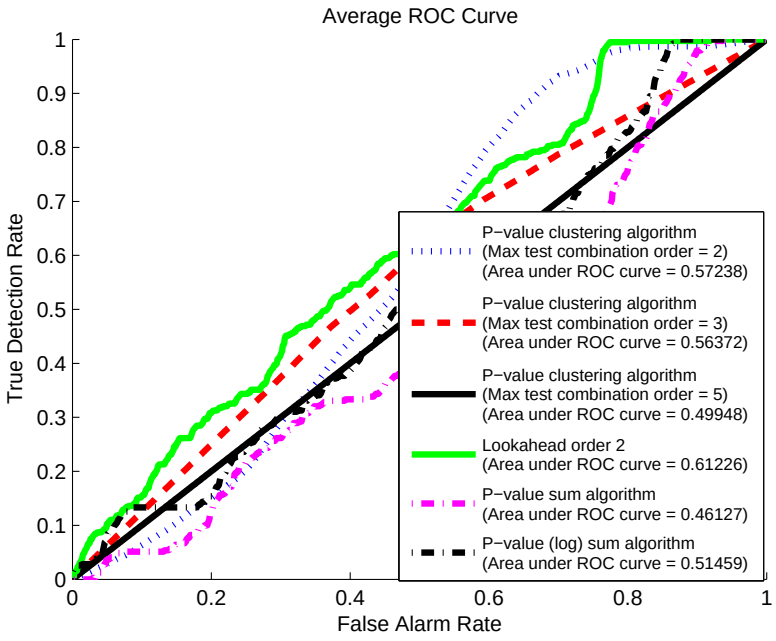

Figure 4.50. ROC curves (File 2 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the same category) (variance lower bound=1)

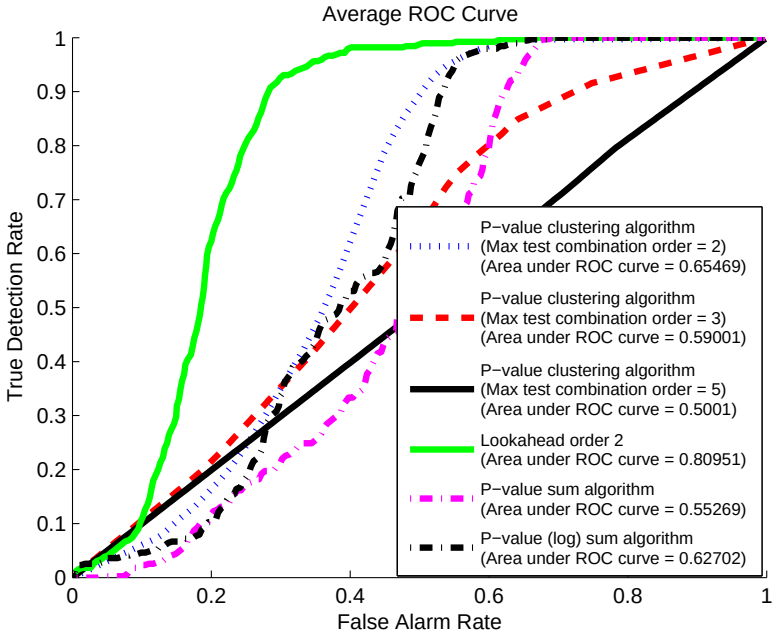

Figure 4.51. ROC curves (File 3 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the same category) (variance lower bound=1)

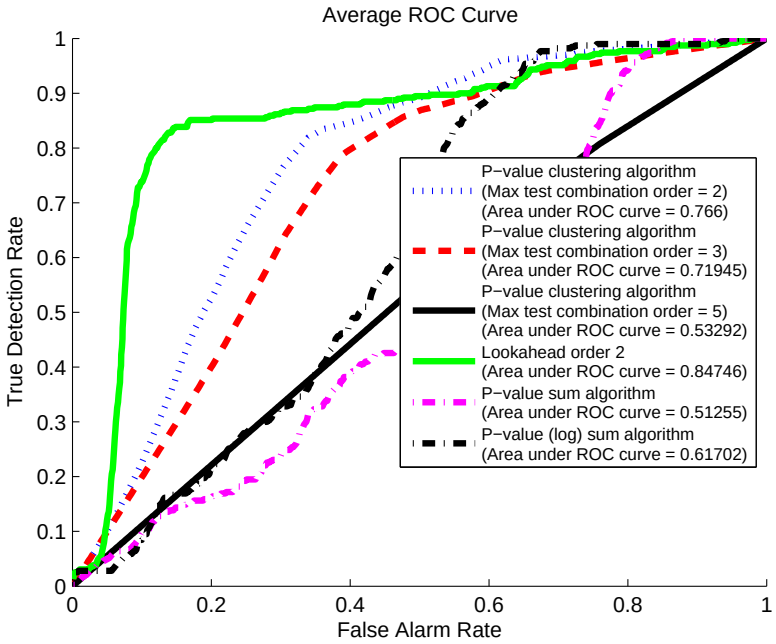

Figure 4.52. ROC curves (File 5 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the same category) (variance lower bound=1)

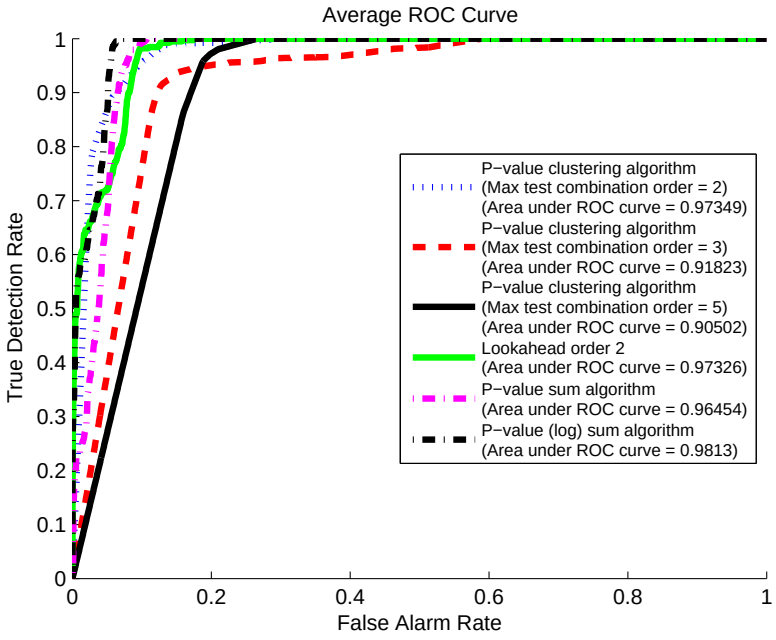

Figure 4.53. ROC curves (File 8 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the same category) (variance lower bound=1)

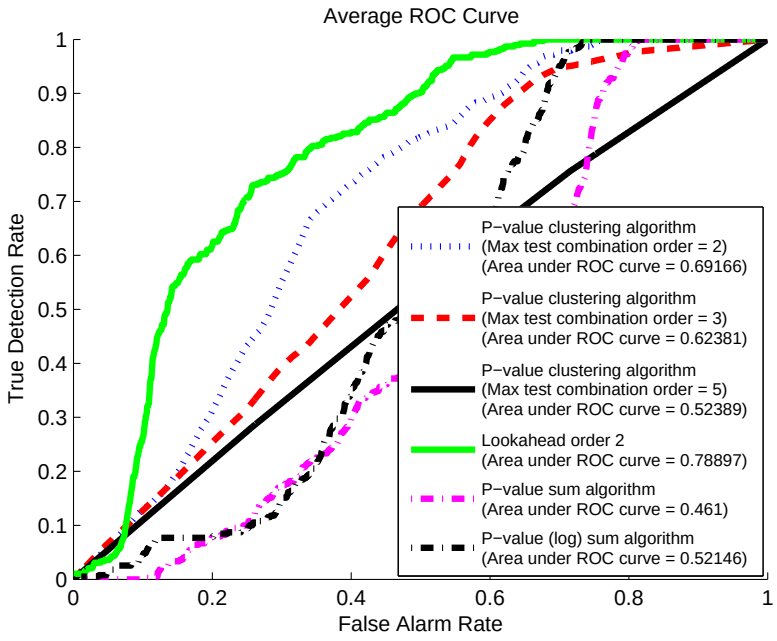

Figure 4.54. ROC curves (Combined File Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the same category) (variance lower bound=1)

4.3.2.6 Alternating feature representation results - with categorical features 0 and ACK (separately)

These results are obtained by using the feature representation defined in Section 2.5.5. The results in Figures 4.55 - 4.59 belong to this approach.

Comparison of methods: As in the previous set of results (Section 4.3.2.5), p-value clustering with order 2 performs well, compared to others. But, Lookahead is the best performing method in almost all of the files, the only exception being File 6 (Figure 4.57) where p-value clustering with order 2 outperforms all methods.

Effect of order increase in P-value clustering: Like in the previous approaches that employ categorical features, order increase strictly worsens the AUC performance.

Comparison with previous categorical approaches: Here, p-value clustering is worse than the case where only 0's are categoricals (Section 4.3.2.4). In the current approach, although the intention of separating the categories of 0's and ACKs was to obtain improvement due to avoiding any mixing of these 2 types of categorical features compared to Section 4.3.2.5, there is no uniform improvement for all methods and files. The continuation of relatively poor performance of p-value clustering methods might be due to the tendency of the algorithm to form large clusters.

Hence, these observations further lead us to make changes to the current approach, carrying us to the next subsection (Section 4.3.2.7).

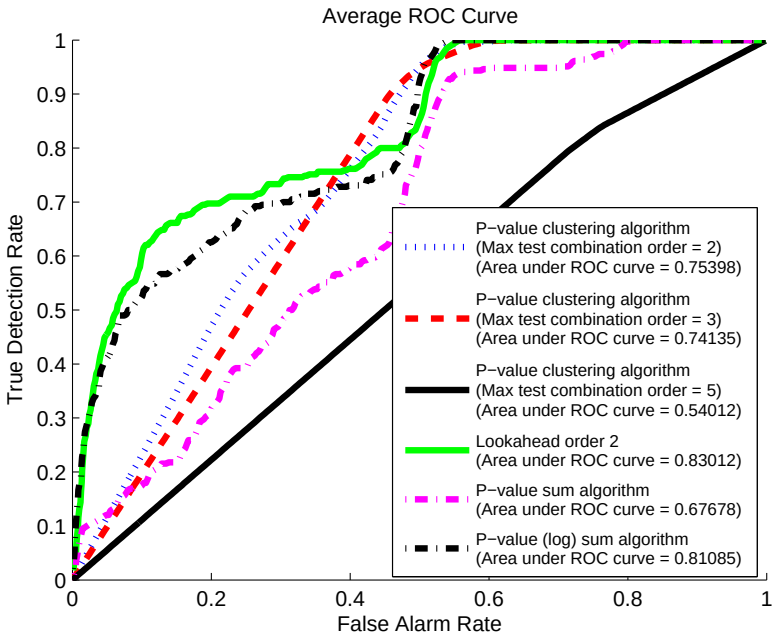

Figure 4.55. ROC curves (File 1 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (variance lower bound=1)

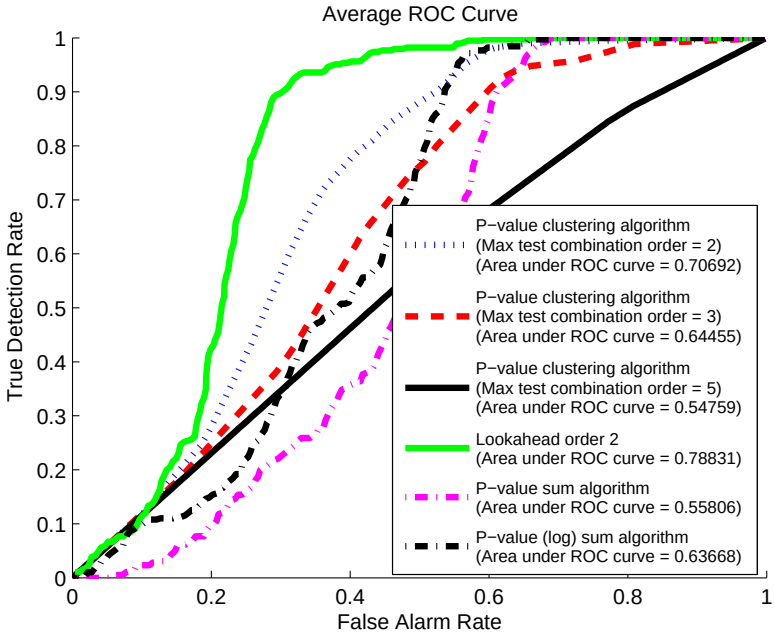

Figure 4.56. ROC curves (File 3 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (variance lower bound=1)

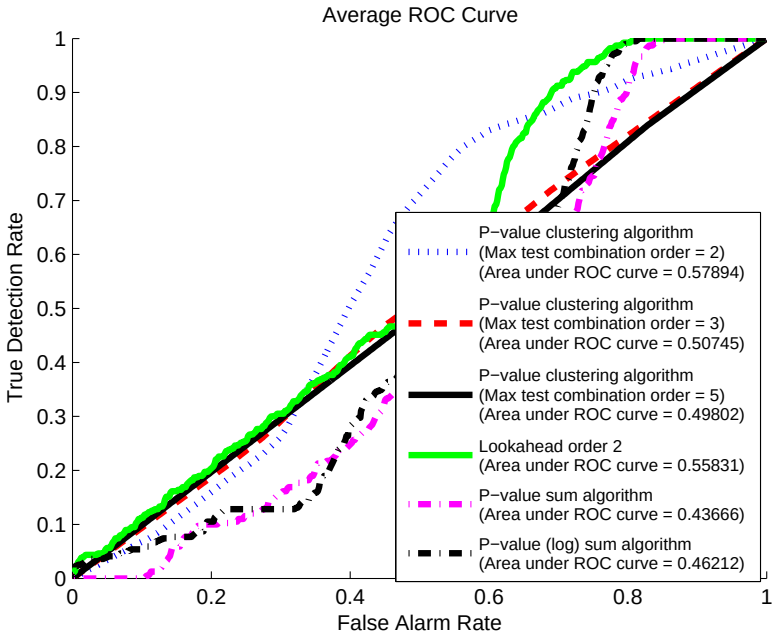

Figure 4.57. ROC curves (File 6 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (variance lower bound=1)

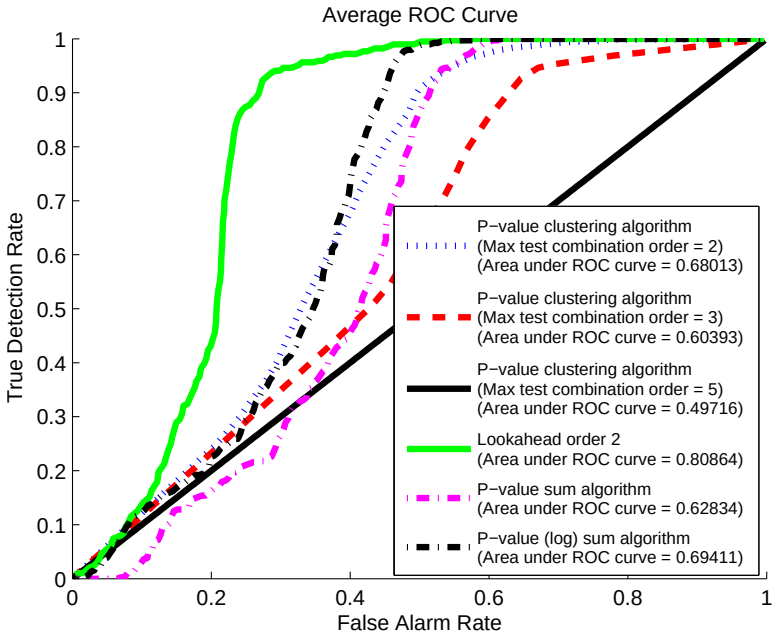

Figure 4.58. ROC curves (File 7 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (variance lower bound=1)

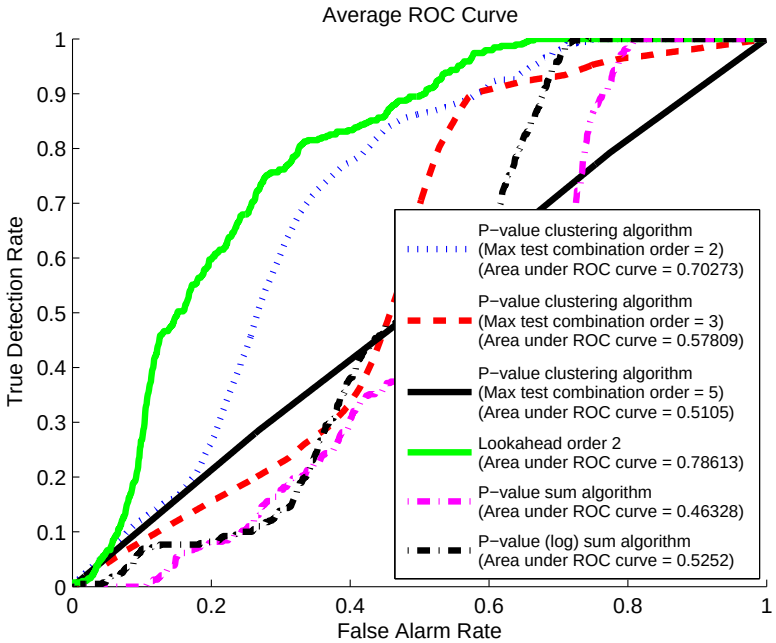

Figure 4.59. ROC curves (Combined File Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (variance lower bound=1)

4.3.2.7 Alternating feature representation results - with categorical features 0 and ACK (separately) (normalized p-values)

These results are obtained by using the feature representation defined in Section 2.5.6. The results in Figures 4.60 and 4.61 belong to this approach.

Effect of normalization: Now, with the normalization applied to p-values obtained by each test, p-value clustering methods got better, making (especially order 2) clustering perform close to Lookahead.

Effect of order increase in P-value clustering: Unlike the previous approaches that employ categorical features, in one result (Figure 4.60) order 5 is better than order 3. But, the other result (Figure 4.61) shows that the performance degrades as order increases.

Positive effect gained by normalizing the p-values, making the maximum value reach 1, and the reason of this being the usage of probabilities corresponding to each conditioning context brings our minds removing the probabilities and seeing the effect of this to the performance. This is investigated in the next subsection.

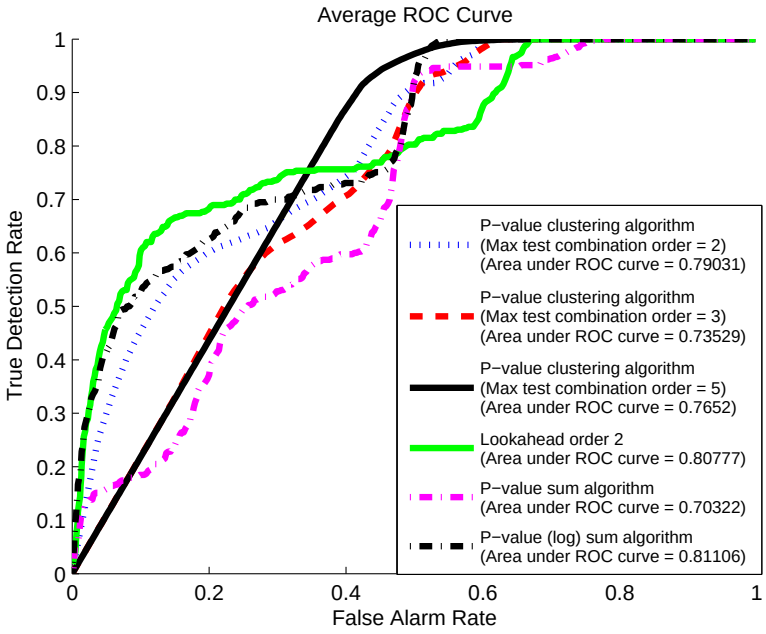

Figure 4.60. ROC curves (File 1 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (p-values for each test normalized) (variance lower bound=1)

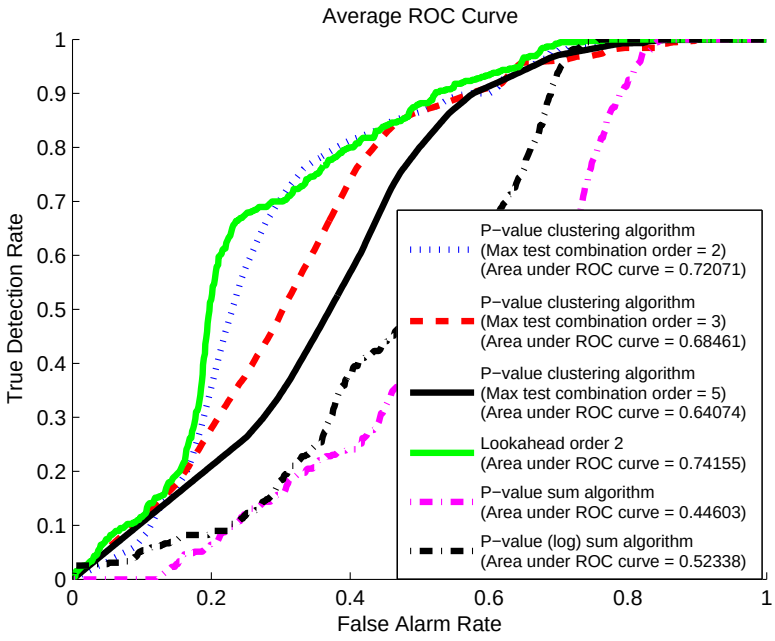

Figure 4.61. ROC curves (File 4 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (p-values for each test normalized) (variance lower bound=1)

4.3.2.8 Alternating feature representation results - with categorical features 0 and ACK (separately)(without probabilities)

These results are obtained by using the feature representation defined in Section 2.5.7. To remind, in this approach, 0's and ACKs are categoricals regarded in different categories, but the probabilities that correspond to each category are not used in the p-value calculations. The results in Figures 4.62 and 4.63 belong to this approach.

Comparison with using probabilities: Section 4.3.2.6 experiments use the probabilities, but here they are not used. Here, we see that not using the probabilities of the conditioning contexts improve the performance of the clustering methods. In comparison with alternating feature representation without any conditioning or modifications (coresponding results are in Section 4.3.2.1), the p-value clustering performance for one file is better here (File 4), and it is worse for another file (File 1).

Effect of order increase in P-value clustering: This has a slight degrading effect on the AUC performance for both order transitions ($2 \rightarrow 3$ and $3 \rightarrow 5$) in Figure 4.62 and same effect only in $3 \rightarrow 5$ transition in Figure 4.63.

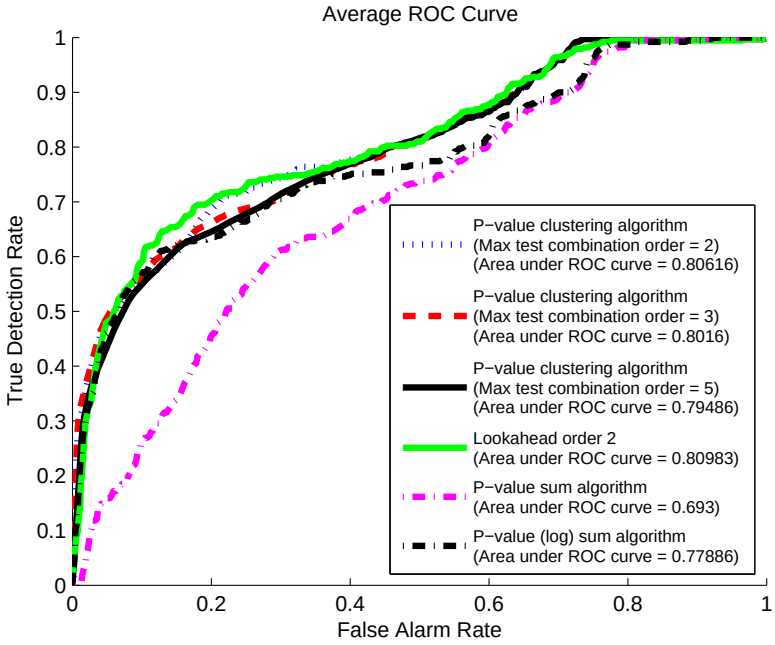

Figure 4.62. ROC curves (File 1 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (without probabilities) (variance lower bound=1)

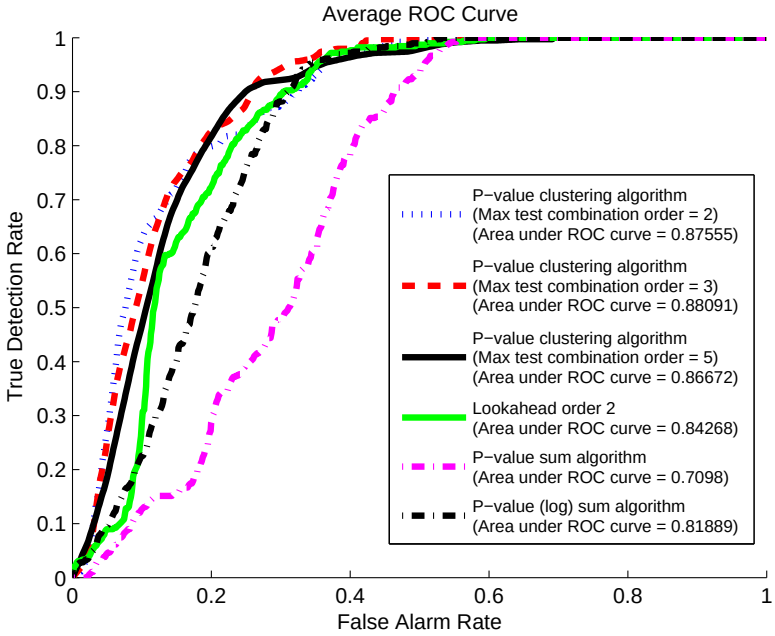

Figure 4.63. ROC curves (File 4 Web - Zeus) (alternating feature representation) (categorical features 0 and ACK, which are considered in the separate categories) (without probabilities) (variance lower bound=1)

4.3.3 Summary

In this chapter, experimental results are provided that enable us to make comparisons between different feature representations and anomaly detection approaches. According to these comparisons, the feature representation that achieves the best discrimination is the alternating feature representation. It is observed that the other representations, which use different conditioning contexts, tend to suffer from poorly informed null in comparison to the alternating representation to which all of the training set is available.

Among the methods, there is not a method that uniformly achieves the best performance in every case. Performances of the methods depend on the dataset. Properties of the datasets that affect the performance are physical port number that the file is captured, dataset size, and capture time of day. It is observed that traffic in certain ports tend to have less diversity and more separable from anomalous traffic. Also, dataset size is directly proportional to training set size in our experiments, which determines how well-informed null is. Time of day is another factor that has effect on the performance. The files that are captured early in the morning tend to have less diversity and more separable from anomalies, similar to the effect that is observed for some ports.

When null is well-informed, certain features have more discrimination power, which are selected by the feature selection methods (p-value clustering approaches). But, when null is poorly informed, collective decision of all of the features are more successful in discrimination. In the latter case, p-value sum and p-value (log) sum methods perform better than p-value clustering approaches.

The above comments are mostly based on area under ROC assessment. It is worth mentioning that, on the average, p-value clustering (order 2) is the best when early detection performance is evaluated by using true positive rate in the first 40 detections.

Chapter 5

Background – Network Neutrality, Games, and Internet Caching

Another vein of this thesis has to do with the network neutrality debate. Mainly, interactions between ISPs, CPs, and consumers are analyzed. Effects of caching on pricing are investigated. This chapter outlines the revenue and demand models that are used in Chapters 6 and 7. Reader must keep in mind that the notation used in the rest of this thesis is unrelated with the previous chapters.

5.1 Network Neutrality

Network neutrality has been supported by the Federal Communication Commission in the United States (with the possible exception of the cellular wireless access context [11, 50]). Basically, network neutrality stipulates that

- two hypothetical sessions that are identical in terms of transmission patterns (bitrates), should be treated the same irrespective of the applications in play for the sessions (*i.e.*, application neutrality), and
- each end-host of a bidirectional session should pay only once to their own ISP for Internet access (*i.e.*, no side payments to remote ISPs).

A communication network is said to be neutral if satisfies both of the above concepts (it is both application neutral and does not require side-payments for use by remote content providers). Application neutrality means that the network does not handle packet-traffic differently based on the application type, *e.g.*, videos from Netflix are

handled the same as the ISP's own managed streaming video service over commodity IP. It is worth mentioning that application neutrality allows discrimination based on traffic volume and end-user specified priorities. So, differentiated services among application types is neutral if requested by end-users themselves, whereas application differentiation implemented unilaterally by an ISP is not application neutral.

So, departures from application neutrality are permitted at the request of the end-users, *e.g.*, if the end-user requests a higher quality-of-service (QoS) for a specific session. Also, neutrality permits ISPs to act on *aggregate traffic volume* or to limit *aggregate traffic bandwidth*. As an example of the former, the ISP could enforce a quota stipulated in an end-user access agreement; such quotas are more tolerable by cellular wireless customers owing to the convenience of mobile access. Note that a discussion of how the presence of such access quotas (and other types of usage-priced overages) raises additional security concerns over flat-rate priced access without traffic volume quotas [77][1] is given in, *e.g.*, [51]; *i.e.*, the departure from flat-rate pricing *incentivizes* more secure end-hosts.

Network neutrality continues to be debated as its core economic issues as described in, *e.g.*, [45], have not been resolved. The debate concerns all participants in the enormous and growing Internet economy: Internet service (access) providers (ISPs), content providers (CPs, including providers of computing services), end-user consumers, and government regulators.

5.2 Games

In Chapters 6 and 7, games between ISPs, CPs, and end-users are analyzed. In Section 5.2.1, the demand model that is used in the following chapters is explained and motivated. Revenue model is also explained here. In Section 5.2.2, the (interior) Nash equilibrium is explained, since this is investigated in the games that will be encountered in the following chapters, under different scenarios. Although this section includes the basics, depending on the specific scenario, the models provided here will be modified when it is necessary.

5.2.1 Revenue and Demand Models

Suppose there is a provider (or providers having common consumers) whose revenue from its subscribers due to its local content is

$$U = pD, \tag{5.1}$$

[1]Flat-rate (wired) residential-broadband end-user contracts typically do involve traffic *bandwidth* limitations that are highly asymmetrical favoring the downlink.

where p is a usage-based price and D is the total demand at that price. Note that ISPs are continuing to depart from pure flat-rate pricing (based on access bandwidth) for unlimited monthly volume, *e.g.*, [95, 16].

Following [52], suppose that there are two broad classes of applications, one of which is significantly sensitive to congestion of access bandwidth, *e.g.*, delay-sensitive interactive real-time applications. Assume that applications of the other, best-effort type are unlikely to engage in usage based-pricing for access bandwidth. As pricing reduces, the demand for access-bandwidth reservation increases, so causing additional congestion so that best-effort service will be increasingly inadequate for congestion-sensitive applications. Therefore, the demand for usage-priced access-bandwidth reservation may *accelerate* with reduced price. More specifically, say there is positive threshold

$$D_\theta \;<\; D_{\max}$$

such that overall demand sensitivity to price is greater when $D \geq D_\theta$ than when $D < D_\theta$. That is, for

$$d_{\max} \;>\; d_\theta,$$

a *convex*, piecewise linear model for access bandwidth would be

$$D(p) \;=\; \max\{D_{\max} - d_{\max}p,\ \hat{D}_\theta - d_\theta p\}, \tag{5.2}$$

where

$$\begin{aligned} \hat{D}_\theta &= D_\theta + (D_{\max} - D_\theta)d_\theta/d_{\max}, \\ p_\theta &= (D_{\max} - D_\theta)/d_{\max}, \\ p_{\max} &= \hat{D}_\theta/d_\theta \;=\; p_\theta + D_\theta/d_\theta, \end{aligned}$$

so that $D(p_\theta) = D_\theta$, see Figure 5.1.

So, in this model, in the price range $[p_\theta, p_{\max}]$ (equivalently, demand range $[0, D_\theta]$) corresponds to low demand sensitivity to price, d_θ. The pricing range $[0, p_\theta]$ (demand range $[D_\theta, D_{\max}]$), when delay-sensitive applications typically need to adopt usage-priced (reserved or priority) access-bandwidth service, corresponds to higher demand sensitivity to price, $d_{\max}$.

Alternatively, suppose a convex, differentiable demand model that can approximate

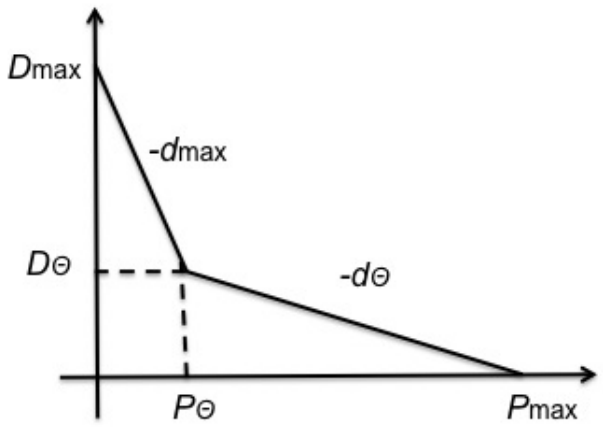

Figure 5.1. Convex, piecewise-linear demand response

(5.2), specifically

$$D(p) \quad = \quad D_{\max}(1 - p/p_{\max})^{\alpha}. \tag{5.3}$$

Here, $\alpha \geq 1$ and given $d_{\max} > d_\theta > 0$ and $0 < D_\theta < D_{\max}$, $p_{\max}$ may be found using $D'(0) = -d_{\max}$ and $D'((D)^{-1}(D_\theta)) = D'(p_\theta) = -d_\theta$. The specific forms of demand in Eq. (5.2) and (5.3) are studied herein because they are tractable.

In [52], we explored the interior Nash equilibria resulting from such convex demand responses. Note how the above models reduce to linear demand response (*e.g.*, by taking $\alpha = 1$), *i.e.*, revenue quadratic in prices, as assumed in many prior papers, *e.g.*, [34].

In the following Chapters (6 and 7), the revenue and the demand models will be based on Eqs 5.1 and 5.3, respectively. The pricing and demands will be changing due to the differences in the models used in those chapters, which will lead to modifications in revenue and demand formulations.

5.2.2 Nash Equilibrium

In Chapters 6 and 7, we will find the Nash equilibrium for the corresponding game in each chapter. In these games, players aim to maximize their utility by changing their pricing strategies.

The Nash equilibrium is a "stalemate" pricing point at which *neither* players' utility will improve by a strategy change. In the following chapters, a player can be an ISP, eyeball ISP, or CP depending on the context. The strategy of a player is only the price that is determined it. Here, to find the Nash equilibrium point, we need to find the point

where none of the players' utility will improve when it changes its pricing. So, at Nash equilibrium point, the following must be satisfied for each player:

$$\arg\max_{p_i} U(p_i, \bar{p}_i^*) = p_i^* \tag{5.4}$$

where U is the utility, p_i is the price of the ith player, and $\bar{p}_i^*$ is the prices of the other players at the equilibrium.

For two players whose utilities and prices are indexed by a and b, the Nash equilibrium point (p_a^*, p_b^*) needs to satisfy

$$\arg\max_{p_a} U_a(p_a, p_b^*) = p_a^* \quad \text{and} \tag{5.5}$$

$$\arg\max_{p_b} U_b(p_a^*, p_b) = p_b^*. \tag{5.6}$$

5.3 Internet Caching

Especially for the sensitive traffic that require high quality-of-service (QoS), the proximity of the physical location of requested content is crucial for decreasing delay experienced by the end-users [41]. Caching significantly reduces the average response time for Web data requests (of course, the fraction of cached content plays a big role on how much improvement is obtained) [90]. Hence, keeping the data close to the users by caching data is of high importance. ISPs are close to the end-users, which makes them a good candidate for caching data. In fact, some large content providers (CPs) cache their content around the world on their own servers, while smaller CPs often use intermediary content distributors, such as Akamai, that have caching agreements with local ISPs [39]. If there are highly dedicated partnerships between ISPs and CPs, ISPs participating in these partnerships can be named as eyeball ISPs. So, as well as the scenario where ISPs cache content of CPs, the scenarios in which eyeball ISPs cache a CP's content or another eyeball ISPs content are also possible.

The contribution of caching to the QoS, for especially the premium services, raises the necessity to determine the amount of content to be cached. Web caches obviously require investment in memory. But, being able to meet the query of the end-user locally saves on the bandwidth that would otherwise be needed to bring the content that the end-user requested (as well as improvement on QoS). In addition to the decision on the size of the cache needed, another decision to make is the policy on how to use the available cache memory. A cache hit means that a request made to the cache is already

in the cache. Otherwise, if the requested data is not in the cache, then a cache miss is said to occur. The cache replacement policies are crucial for effective management of the caches. They basically aim to maximize the proportion of the cache hits among the total queries made to the cache. Many policies can be employed as a cache replacement policy. The more notable one among these is the Least Recently Used (LRU) [90, 17]. As the name implies, in this policy, if there is a cache miss, the requested object replaces the object that hasn't been used for the longest time.

Chapter 6

Network Neutrality – Effect of Caching in a Network with Two Eyeball ISPs

In this chapter, we first give a model involving two different eyeball ISPs connected at peering point(s), where revenue is generated corresponding to net traffic transmitted, is initially considered in Sections 6.1 and 6.2. We consider a caching model captured by a single parameter, Φ, affecting the revenue generated by transit traffic. We assume that there is no limit on the throughput downstream to the users of each ISP. In Section 6.3, we modify the model so that there is an upper bound on the throughput that the users can receive via their ISP. So, two possible mechanisms to distribute the allowed throughput among the types of demands (local or remote content) are introduced. We next consider the scenario where there are multiple providers competing for the same group of users (without the throughput limit condition, as in the initial model). User/customer migration among competing ISPs due to the price difference between them is modeled by their "loyalties" to the ISPs. In Section 6.4, consideration of two ISPs competing for the same set of users is added to the model described in Section 6.2. We provide the results of numerical experiments on performance at Nash equilibrium in Section 6.5.

6.1 Two different eyeball ISPs

We consider a game focusing on two different eyeball ISPs, indexed a and b, on a platform of users and CPs, *i.e.*, the ISPs also serve as CPs so no separate pricing by CPs is

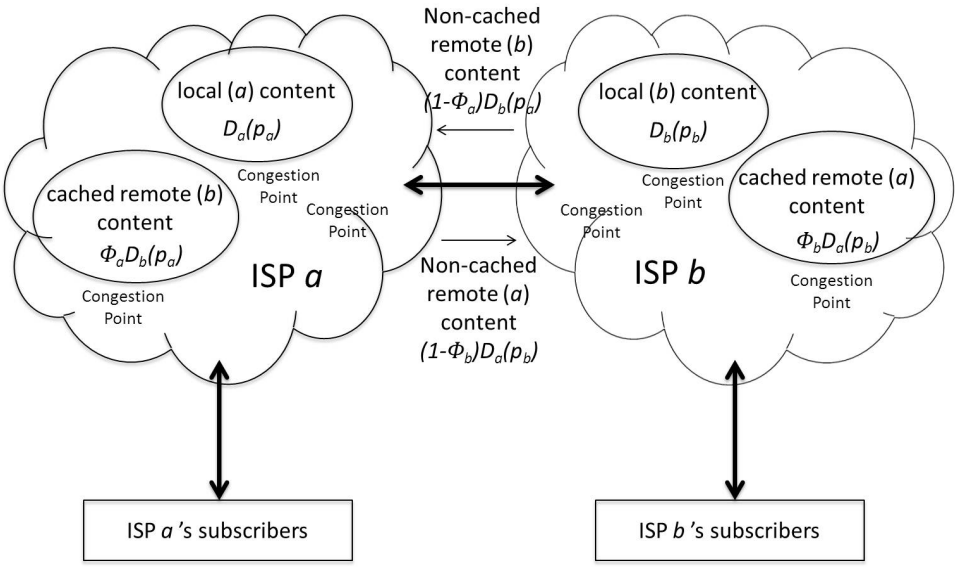

Figure 6.1. Caching remote content

modeled. For $k, j \in \{a, b\}$, the demand for ISP k's content is $D_k(p_j)$ when it is based on ISP j's access-bandwidth price p_j. In the following, the same price p_j will be used by ISP j irrespective of content source, *i.e.*, content is neutrally priced in this sense.

Suppose there are peering points between these two ISPs where net transit traffic flow in one direction will correspond to net revenue for the (net) receiving ISP at rate p_t from the (net) transmitting ISP. For example, France telecom charges p_t =\$3/Mbps, whereas pricing from the digital subscriber line access multiplexer (DSLAM) to core, *i.e.*, access bandwidth, for their *content providers* is \$40/Mbps [86]. This said, many existing peering agreements among non-transit ISPs have no transit pricing, *i.e.*, $p_t = 0$. See [30, 105] for recent studies of models of transit pricing for a network involving a transit ISP between the content providers and end-user ISPs.

Without caching, transit traffic volume is obviously maximal and remote content may be subject to additional delay possibly increasing demand (reducing demand sensitivity) for usage-priced bandwidth reservations. However, poorer delay performance may instead reduce demand for remote content or cause subscribers to change to ISPs that cache remote content. So, caching will result in reduced demand for premium services by transit traffic; in the following, we will model this with a caching factor Φ_k. We assume fixed caching factors for each of the ISPs, which means the selected caching factors by the ISPs do not change no matter how their demand changes.

6.2 Three different congestion points per ISP, fixed caching factors

By simply separately accounting for the demand for premium-access service by two different user populations with similar content preferences, we take the utilities as:

$$\begin{aligned} U_a(p_a, p_b) =& D_a(p_a)p_a + \Phi_a D_b(p_a)p_a \\ &+ [(1-\Phi_a)D_b(p_a) - (1-\Phi_b)D_a(p_b)]^+ p_t, \\ U_b(p_a, p_b) =& D_b(p_b)p_b + \Phi_b D_a(p_b)p_b \\ &+ [(1-\Phi_b)D_a(p_b) - (1-\Phi_a)D_b(p_a)]^+ p_t \end{aligned}$$

where $[x]^+ := \max\{x, 0\}$ in the second (transit revenue) terms. Note that $\Phi_k \leq 1$ will be chosen by ISP k at its *minimal* value, which we here assume to be strictly positive again because an ISP that does not cache any remote content may lose subscribers, or demand for remote content may be reduced owing to poor delay performance, *cf.*, Section 6.4. We will also assume that p_t is fixed and, by volume discount, $p_t < \min\{p_a, p_b\}$. Also, we have assumed different "upstream" congestion points for local and remote traffic and no revenue from cached (best-effort) traffic. Moreover, for $\alpha > 1$ (*i.e.*, not linear demand response) note how this model assumes three different congestion points, one at the peering point, one at the local content source, and the last one at the cached content source, but *not* a single one further downstream toward the users, *cf.*, next section. That is, in this section, we consider three separate congestion points per ISP for an example of convex demand (assumptions that include the linear demand-response scenario as a special case).

Again suppose, for $k \in \{a, b\}$, that

$$D_k(p) = D_{\max,k}\left(1 - \frac{p}{p_{\max}}\right)^{\alpha}, \tag{6.1}$$

where the maximal price $p_{\max} > 0$ and $\alpha \geq 1$ are also assumed to be common parameters for both ISPs to simplify the following expressions for Nash equilibria. Without loss of generality, assume the demand ratio

$$\delta := \frac{D_{\max,b}}{D_{\max,a}} \;\leq\; 1, \tag{6.2}$$

i.e., demand for ISP a's content is generally higher than that of ISP b.

The first order Nash equilibrium conditions and the solutions of these for 3 cases are provided below.

Case 1: $(1-\Phi_a)D_b(p_a^*) > (1-\Phi_b)D_a(p_b^*)$.

$$\begin{aligned}\frac{\partial U_a(p_a,p_b)}{\partial p_a} =& D_a'(p_a)p_a + D_a(p_a) + \Phi_a[D_b'(p_a)p_a + \\ & D_b(p_a)] + (1-\Phi_a)D_b'(p_a)p_t = 0 \\ \frac{\partial U_b(p_a,p_b)}{\partial p_b} =& D_b'(p_b)p_b + D_b(p_b) + \Phi_b[D_a'(p_b)p_b \\ & + D_a(p_b)] = 0\end{aligned}$$

The solution is as follows:

$$p_a^* = \frac{p_{\max}}{1+\alpha} - \frac{p_t(1-\Phi_a)\delta\alpha}{(1+\alpha)(1+\Phi_a\delta)}, \tag{6.3}$$

$$p_b^* = \frac{p_{\max}}{1+\alpha}. \tag{6.4}$$

The requirement $p_t < p_a^* < p_b^* < p_{\max}$ gives the following condition on p_t for an interior Nash equilibrium:

$$\frac{p_{\max}}{p_t} > 1 + \frac{\alpha(\delta+1)}{1+\delta\Phi_b}. \tag{6.5}$$

Another way to put the case condition $(1-\Phi_a)D_b(p_a^*) > (1-\Phi_b)D_a(p_b^*)$ is:

$$1 < \frac{(1-\Phi_a)\delta}{1-\Phi_b}\left(\frac{p_{\max}-p_a^*}{p_{\max}-p_b^*}\right)^{\alpha}, \text{ and} \tag{6.6}$$

$$1 < \frac{(1-\Phi_a)\delta}{1-\Phi_b}\left(1 + \frac{(1-\Phi_a)\delta p_t}{(1+\Phi_a\delta)p_{\max}}\right)^{\alpha}. \tag{6.7}$$

Case 2: $(1-\Phi_a)D_b(p_a^*) < (1-\Phi_b)D_a(p_b^*)$.

$$\begin{aligned}\frac{\partial U_a(p_a,p_b)}{\partial p_a} =& D_a'(p_a)p_a + D_a(p_a) + \Phi_a[D_b'(p_a)p_a \\ & + D_b(p_a)] = 0 \\ \frac{\partial U_b(p_a,p_b)}{\partial p_b} =& D_b'(p_b)p_b + D_b(p_b) + \Phi_b[D_a'(p_b)p_b \\ & + D_a(p_b)] + (1-\Phi_b)D_a'(p_b)p_t = 0\end{aligned}$$

The solution is as follows:

$$p_a^* = \frac{p_{\max}}{1+\alpha}, \tag{6.8}$$

$$p_b^* = \frac{p_{\max}}{1+\alpha} - \frac{p_t(1-\Phi_b)\alpha}{(1+\alpha)(\delta+\Phi_b)}. \tag{6.9}$$

The requirement $p_t < p_b^* < p_a^* < p_{\max}$ imposes the following condition on p_t:

$$\frac{p_{\max}}{p_t} > 1 + \frac{\alpha(\delta+1)}{\delta+\Phi_b} \tag{6.10}$$

The case condition $(1-\Phi_a)D_b(p_a^*) < (1-\Phi_b)D_a(p_b^*)$ can be rewritten as:

$$1 > \frac{(1-\Phi_a)\delta}{1-\Phi_b}\left(\frac{p_{\max}-p_a^*}{p_{\max}-p_b^*}\right)^{\alpha}, \text{ and} \tag{6.11}$$

$$1 > \frac{(1-\Phi_a)\delta}{1-\Phi_b}\left(1+\frac{(1-\Phi_a)\delta p_t}{(1+\Phi_a\delta)p_{\max}}\right)^{\alpha}. \tag{6.12}$$

Case 3: $(1-\Phi_a)D_b(p_a^*) = (1-\Phi_b)D_a(p_b^*)$.

$$\begin{aligned}
\frac{\partial U_a(p_a,p_b)}{\partial p_a} =& D_a'(p_a)p_a + D_a(p_a) \\
&+ \Phi_a[D_b'(p_a)p_a + D_b(p_a)] = 0 \\
\frac{\partial U_b(p_a,p_b)}{\partial p_b} =& D_b'(p_b)p_b + D_b(p_b) \\
&+ \Phi_b[D_a'(p_b)p_b + D_a(p_b)] = 0
\end{aligned}$$

The solution of above equations is as follows:

$$p_a^* = p_b^* = \frac{p_{\max}}{1+\alpha} \tag{6.13}$$

The case condition reduces to

$$\frac{1-\Phi_b}{1-\Phi_a} = \frac{D_{\max,b}}{D_{\max,a}} = \delta \tag{6.14}$$

6.3 One congestion point per ISP, fixed caching factors

In this scenario, at ISP a, the demands $D_a(p_a)$ (demand for local content) and $D_b(p_a)$ (demand for remote content) share a common, significant congestion point proximal to

the users, *e.g.*, in a wireless-access setting. Again, we consider a system where the players (eyeball ISPs) select access prices (plays) $p_a, p_b > p_t$.

Given the prices p_a for local content, we want an expression for demand $\hat{D}_{aa}$ (local content at ISP a) and $\hat{D}_{ba}$ (remote content at ISP a) that has the following intuitive property:

$$\lim_{D_{\max,b}\to 0} \hat{D}_{aa} = D_a(p_a) \text{ and } \lim_{D_{\max,a}\to 0} \hat{D}_{ba} = D_b(p_a). \tag{6.15}$$

And similarly for ISP b regarding $\hat{D}_{bb}$ and $\hat{D}_{ab}$ as a function of p_b.

The following assumed property is also intuitive because the presence of remotely originated traffic will congest locally originated traffic and vice versa:

$$\hat{D}_{aa} \leq D_a(p_a) \quad \text{and} \quad \hat{D}_{ba} \leq D_b(p_a) \tag{6.16}$$

and similarly for the other ISP b.

Proportion Rule: Suppose that the throughput limit downstream to the users is L_k for ISP $k \in \{a, b\}$. Then, at ISP a, the demands are as follows:

$$\hat{D}_{aa} = \begin{cases} \dfrac{D_a(p_a)}{D_a(p_a) + D_b(p_a)} L_a, & \text{if } D_a(p_a) + D_b(p_a) > L_a \\ D_a(p_a) \qquad\qquad , & \text{else.} \end{cases}$$

and

$$\hat{D}_{ba} = \begin{cases} \dfrac{D_b(p_a)}{D_a(p_a) + D_b(p_a)} L_a, & \text{if } D_a(p_a) + D_b(p_a) > L_a \\ D_b(p_a) \qquad\qquad , & \text{else.} \end{cases}$$

And similarly for ISP b.

Critical Price Rule: Another way to split the throughput among the demands is as follows. For ISP a, when $D_a(p_a) + D_b(p_a) > L_a$, a new price p_a^* is chosen so that

$$D_a(p_a^*) + D_b(p_a^*) = L_a. \tag{6.17}$$

If $p_a < p_a^*$, then congestion will occur.

So, the expressions for the ISP revenues here can be taken as

$$U_a(p_a) = \hat{D}_{aa} p_a + \Phi_a \hat{D}_{ba} p_a$$

$$
\begin{aligned}
&+ [(1-\Phi_a)\hat{D}_{ba} - (1-\Phi_b)\hat{D}_{ab}]^+ p_t \\
U_b(p_b) =& \hat{D}_{bb}p_b + \Phi_b \hat{D}_{ab} p_b \\
&+ [(1-\Phi_b)\hat{D}_{ab} - (1-\Phi_a)\hat{D}_{ba}]^+ p_t.
\end{aligned}
$$

6.4 Three different congestion points per ISP, fixed caching factors, multiple providers of one of the types

In this scenario, ISP a in Figure 6.1 is replaced by two ISPs, namely ISP $a1$ and $a2$, which compete for the same group of subscribers. So, we need to consider three utility functions; U_{a1}, U_{a2}, U_b; three demand functions, D_{a1}, D_{a2}, D_b; and three access prices for each of the ISPs' own subscribers, p_{a1}, p_{a2}, p_b. But the number of caching factors increases to four: $\Phi_{a1,b}$, $\Phi_{a2,b}$, $\Phi_{b,a1}$, and $\Phi_{b,a2}$ ($\Phi_{m,n}$ meaning willingness of ISP m to cache the content of ISP n). And, there are 2 transit prices, that are p_{t1} (for the traffic between ISP $a1$ and ISP b) and p_{t2} (for ISPs $a2$ and b).

$$
\begin{aligned}
U_{a1}(p_{a1}, p_b) =& \sigma_{a1} D_{a1}(p_{a1}) p_{a1} + \sigma_{a1}\Phi_{a1,b} D_b(p_{a1}) p_{a1} \\
&+ [\sigma_{a1}(1-\Phi_{a1,b}) D_b(p_{a1})) \\
&- (1-\Phi_{b,a1}) D_{a1}(p_b)]^+ p_{t1} \\
U_{a2}(p_{a2}, p_b) =& \sigma_{a2} D_{a2}(p_{a2}) p_{a2} + \sigma_{a2}\Phi_{a2,b} D_b(p_{a2}) p_{a2} \\
&+ [\sigma_{a2}(1-\Phi_{a2,b}) D_b(p_{a2})) \\
&- (1-\Phi_{b,a2}) D_{a2}(p_b)]^+ p_{t2} \\
U_b(p_{a1}, p_{a2}, p_b) =& D_b(p_b) p_b + \Phi_{b,a1} D_{a1}(p_b) p_b \\
&+ \Phi_{b,a2} D_{a2}(p_b) p_b + [(1-\Phi_{b,a1}) D_{a1}(p_b) \\
&- \sigma_{a1}(1-\Phi_{a1,b}) D_b(p_{a1})]^+ p_{t1} \\
&+ [(1-\Phi_{b,a2}) D_{a2}(p_b)) \\
&- \sigma_{a2}(1-\Phi_{a2,b}) D_b(p_{a2})]^+ p_{t2}
\end{aligned}
$$

where

$$
\sigma_{ai} = \frac{1/p_{ai}}{1/p_{a1} + 1/p_{a2}}, \qquad \forall\, i \in \{1, 2\}
$$

represents customer stickiness (loyalty, inertia) to the ith ISP (*e.g.*, [21]); *i.e.*, since $\sigma_{ai} \propto 1/p_{ai}$, the subscribers will not completely switch to the ISP with the lowest price.

The demand-response model provided in (5.3) is used here, now with $k \in \{a1, a2, b\}$.

6.5 Numerical experiments

First, numerical results were obtained for the scenario where there are three congestion points per ISP (with fixed caching factors, as explained in Section 6.2) with: $\alpha \in \{1, 2\}$, $D_{\max,a} = 20$, $D_{\max,b} = 10$, $p_{\max} = 5$, $p_t = 1$, $\Phi_a = 0.5$, and $\Phi_b = 0.3$ as the selected parameter values.

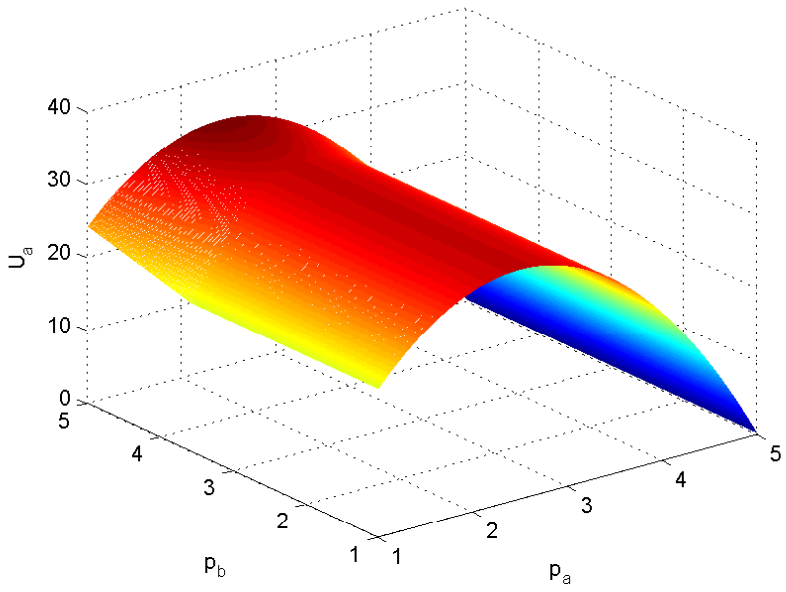

Figure 6.2. $U_a(p_a, p_b)$ (3 congestion points for each ISP, fixed caching factors) ($\alpha = 1$)

By using $U_a(p_a, p_b)$ (Figure 6.2) and $U_b(p_a, p_b)$ (Figure 6.3), the Nash equilibrium point (p_a^*, p_b^*) were found in the following way:

1. Uniformly at random over $(p_t, p_{\max})$ select an initial point $\gamma^{(0)} = (p_a^{(0)}, p_b^{(0)})$.

2. $\forall\, k \geq 1$, find the updated point $\gamma^{(k)} = (p_a^{(k)}, p_b^{(k)})$ by synchronous best-response updates, which are

$$p_a^{(k)} = \arg\max_{p_a} U_a(p_a, p_b^{(k-1)})$$
$$p_b^{(k)} = \arg\max_{p_b} U_b(p_a^{(k-1)}, p_b).$$

3. (a) If $\gamma^{(k-1)} \approx \gamma^{(k)}$, stop.

   (b) Else, return to step 2).

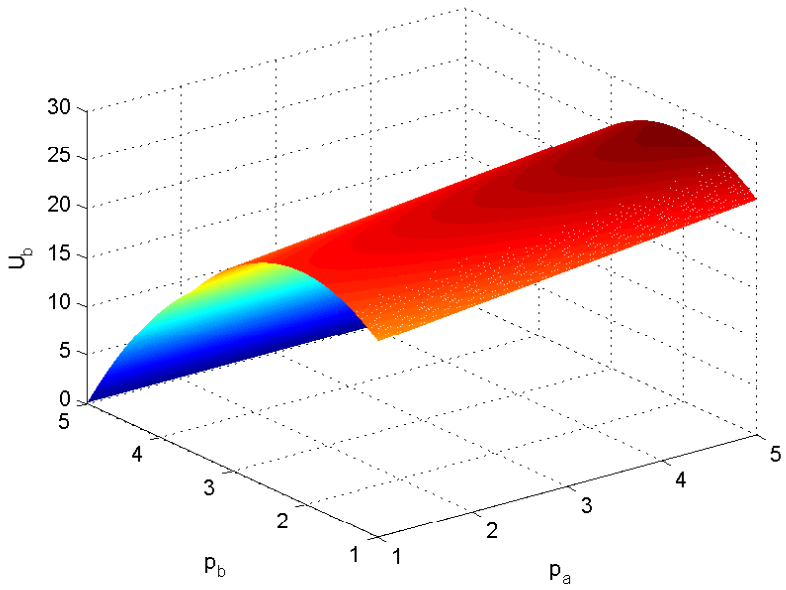

Figure 6.3. $U_b(p_a, p_b)$ (3 congestion points for each ISP, fixed caching factors) ($\alpha = 1$)

It was observed that the Nash equilibrium point found by using the above procedure is the same as the equilibrium point corresponding to the proper case solution provided in Section 6.2 (regardless of the randomly selected starting point) and it was found in just a few iterations.

It can be observed in Figures 6.6 and 6.7 that $p_a^* > p_b^*$ and $U_a(p_a^*, p_b^*) > U_b(p_a^*, p_b^*)$ for both values of α. This is intuitive since $D_{\max,a} > D_{\max,b}$, which implies that the demand for ISP a's content will be larger than ISP b's at the same price. This immediately implies larger gain for ISP a, which also means that ISP a might have some margin for increasing p_a in order to gain even more utility. Therefore $p_a^* > p_b^*$ in this setting.

Next, numerical results were obtained for the model defined in Section 6.3, where one congestion point per ISP and fixed caching factors assumptions are used. Here, the throughput limit is split among the ISPs according to the proportion rule, *cf.*, Section 6.3. $\alpha \in \{1, 2\}$, $D_{\max,a} = 20$, $D_{\max,b} = 10$, $p_{max} = 5$, $p_t = 1$, $\Phi_a = 0.5$, $\Phi_b = 0.3$, $L_a = 50$, and $L_b = 5$ are the selected parameters values. Notice that one of the throughput limits (L_a) is selected significantly larger than the other one (L_b) to analyze the scenario where congestion does not occur downstream to the users of ISP a, whereas it does occur for ISP b. If both of the throughput limits are selected very large, then the problem reduces to the three congestion points scenario (Section 6.2), since there will be no distribution of the throughput limit between the two different kinds of demand at the congestion

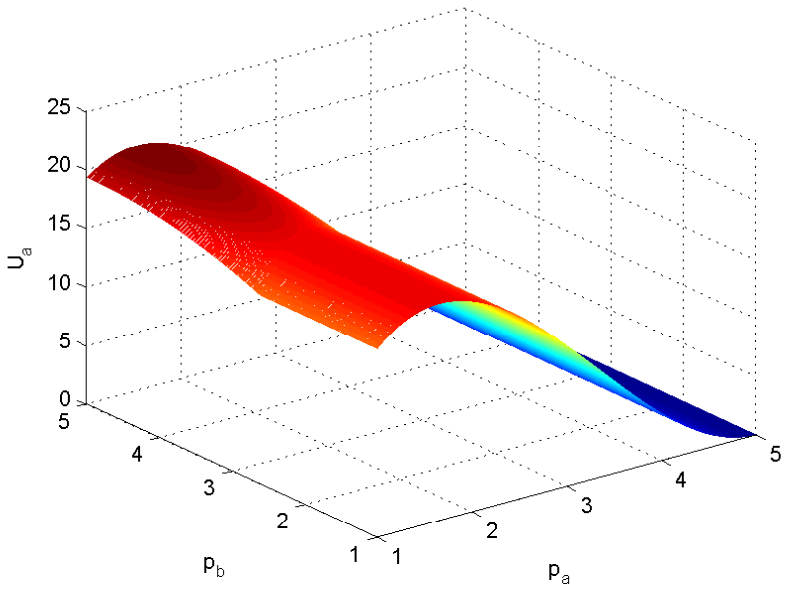

Figure 6.4. $U_a(p_a, p_b)$ (3 congestion points for each ISP, fixed caching factors) ($\alpha = 2$)

point (of each ISP).

The Nash equilibrium point was again quickly found by using synchronous best-response updates.

In Figures 6.8 and 6.9, similar behaviors are observed compared with Figures 6.6 and 6.7. But, it is worth noting that in Figure 6.9, for values of p_b where U_b is increasing (for both $\alpha \in \{1, 2\}$), the capacity L_b is fully utilized. In this region, increasing p_b does not lead to a decrease in the demand, which means there is a linear increase in the utility of ISP b. But, after the peak, the total demand at ISP b is smaller than L_b, therefore the increase in price p_b leads to decreases in both demand and utility.

Finally, numerical results were obtained for the case where there are multiple providers competing for the same group of subscribers (Section 6.4). Again, synchronous best-response updates are used, but for three utility functions ($U_{a1}(p_{a1}, p_{a2}, p_b)$, $U_{a2}(p_{a1}, p_{a2}, p_b)$, and $U_b(p_{a1}, p_{a2}, p_b)$) depending on the corresponding three access pricing parameters (p_{a1}, p_{a2}, and p_b). So, generally, for n competing ISPs ($n = 2$ in our case of ISPs $a1$ and $a2$), the synchronous best-response update step ($n + 1$ player synchronous updates) will be as follows:

$$p_i^{(k)} = \arg\max_{p_i} U_i(p_i, \underline{p}_{-i}^{(k-1)}), \quad \forall\, i$$

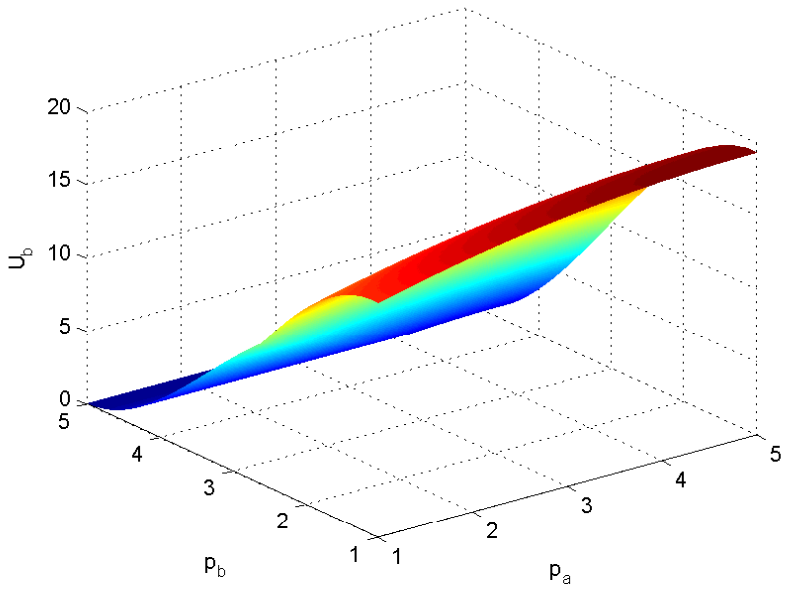

Figure 6.5. $U_b(p_a, p_b)$ (3 congestion points for each ISP, fixed caching factors) ($\alpha = 2$)

where i is the index of the ISP (including the non-competing ISP (in our case, ISP b)), p_i is the price used by ISP i, and $\underline{p}_{-i}$ is the set of prices used by the other ISPs.

The parameter values can be selected in various combinations. We used the parameters $D_{\max,a1} = 20$, $D_{\max,a2} = 20$, $D_{\max,b} = 10$, $p_{\max} = 5$, $p_{t1} = 1$, $p_{t2} = 1$, $\Phi_{a1,b} = 0.2$, $\Phi_{a2,b} = 0.8$, $\Phi_{b,a1} = 0.5$, and $\Phi_{b,a2} = 0.5$. These were selected so as to analyze the effect of (static but different) caching factors of competing ISPs (ISPs $a1$ and $a2$) on the utilities. It can observed from Figures 6.10 and 6.11 that the ISP with smaller Φ ($a1$) also has (again following intuition) a smaller utility compared to its competitor ISP ($a2$). The effect of α on the utilities and the equilibrium prices are the same as the previous cases.

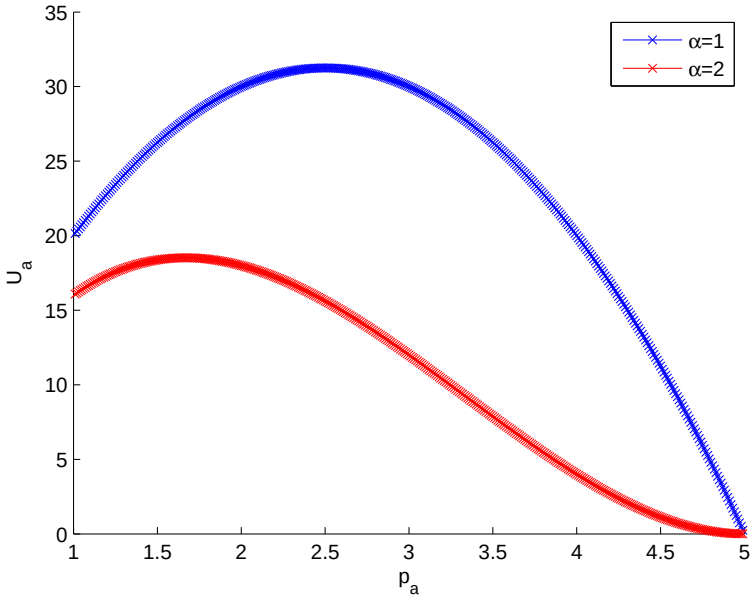

Figure 6.6. $U_a(p_a, p_b^*)$ (3 congestion points for each ISP, fixed caching factors)

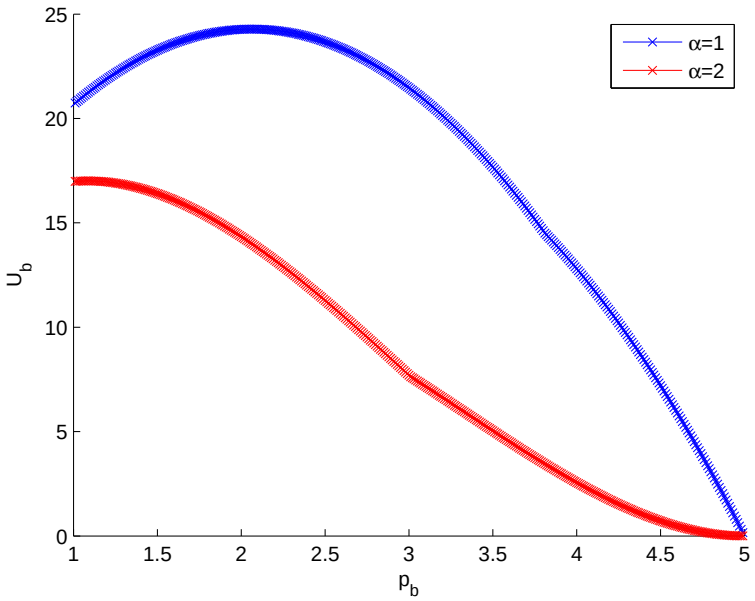

Figure 6.7. $U_b(p_a^*, p_b)$ (3 congestion points for each ISP, fixed caching factors)

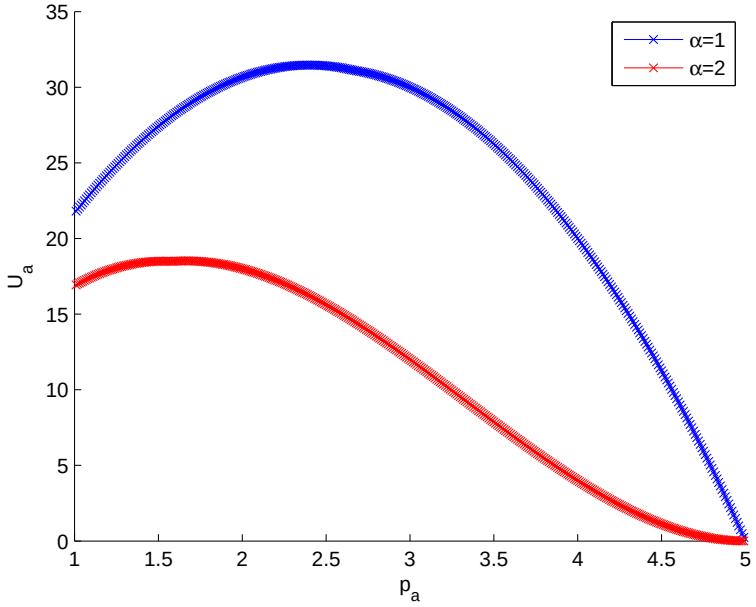

Figure 6.8. $U_a(p_a, p_b^*)$ (1 congestion point for each ISP, fixed caching factors)

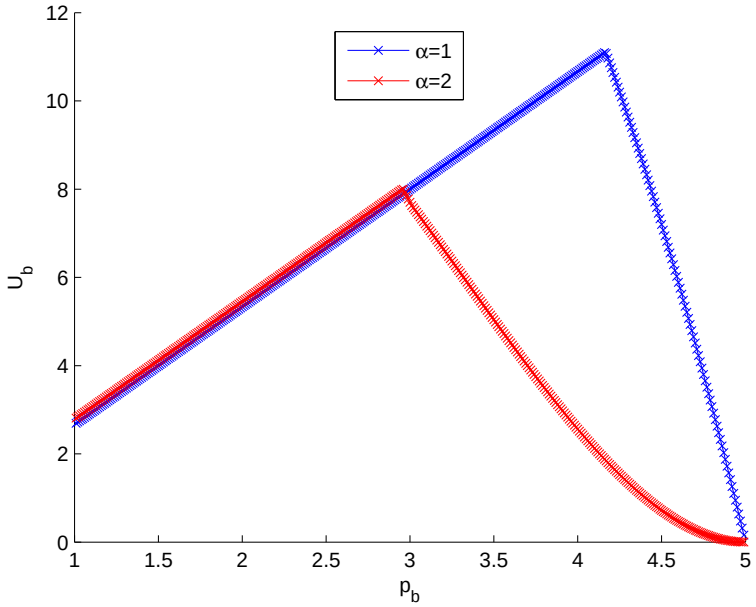

Figure 6.9. $U_b(p_a^*, p_b)$ (1 congestion point for each ISP, fixed caching factors)

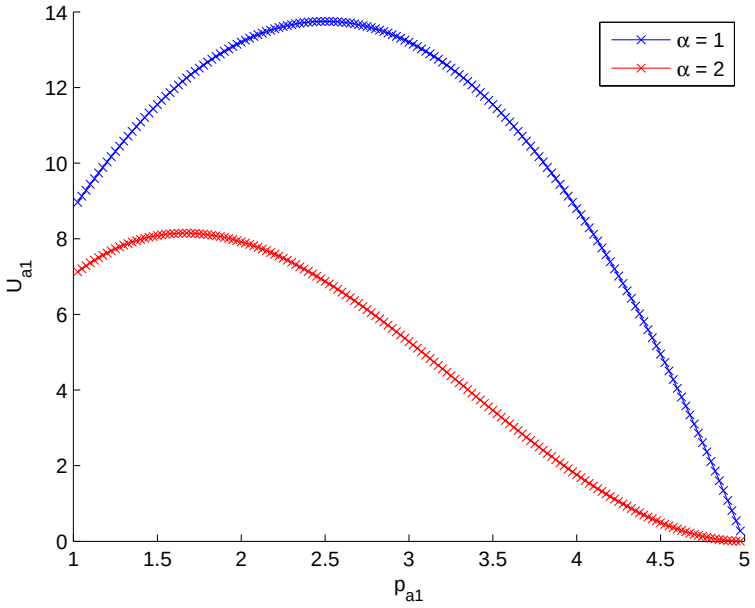

Figure 6.10. $U_{a1}(p_{a1}, p^*_{a2}, p^*_b)$ (3 congestion points for each ISP, fixed caching factors, competing ISPs)

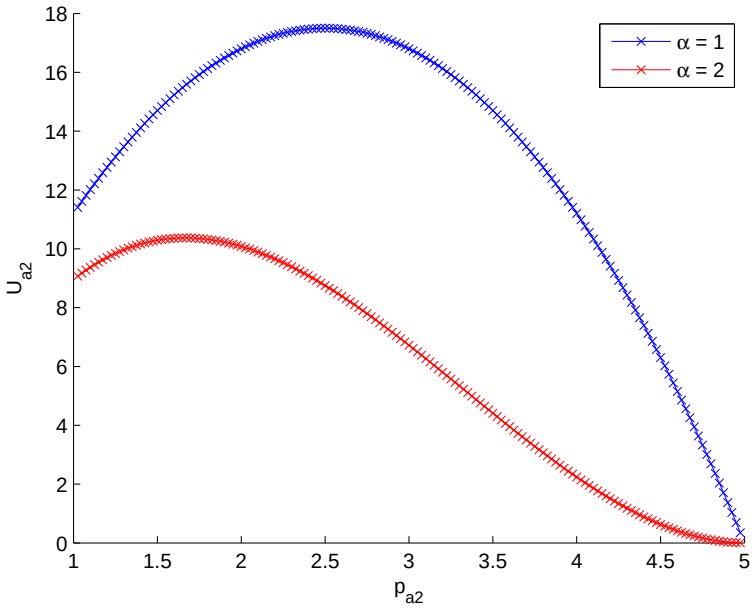

Figure 6.11. $U_{a2}(p^*_{a1}, p_{a2}, p^*_b)$ (3 congestion points for each ISP, fixed caching factors, competing ISPs)

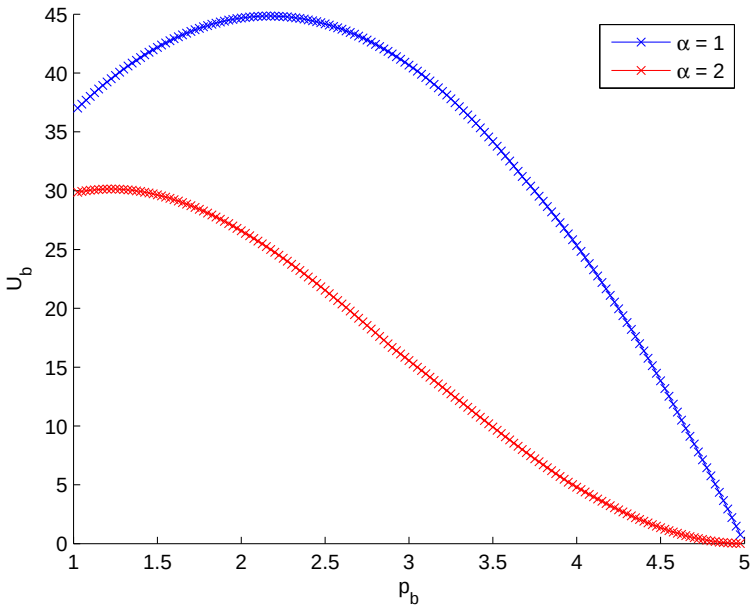

Figure 6.12. $U_b(p^*_{a1}, p^*_{a2}, p_b)$ (3 congestion points for each ISP, fixed caching factors, competing ISPs)

Chapter 7

Network Neutrality – Effect of Caching in Information-Centric Networks

This chapter is organized as follows. Section 7.1 provides a motivational discussion on the connection between network neutrality, ISP-level content caching, and future Internet architectures. In Section 7.2, we summarize prior results on a simple ISP-CP game for the "Internet" setting. In Section 7.3, we adapt these results to the ICN setting and extend the model to account for content caching by the ISP. A key element of the extension is a price-convex demand-response motivated by delay-sensitivity of the applications/content under consideration. In Section 7.4, we give the results of a numerical study.

7.1 Background discussion

7.1.1 Network neutrality and ISP-level content caching

In Chapter 5.1, concerns about flat rate pricing are provided. Moreover, there may be penalties for asymmetric (net) traffic-aggregates at inter-ISP and/or ISP/transit-provider peering points [37, 105]. In some important instances, these penalties amount to a side-payments between CPs and remote ISP. For example, a large CP may team with a transit-provider (TP) and the peerings between that TP and an ISP may result in traffic volumes that are naturally much higher from TP to ISP than vice versa. This traffic asymmetry will generate revenue for the ISP from the TP, costs that the TP will

naturally try to recover from the CP[1] So for the "Internet" setting, we assume a net side-payment from CP to ISP in the following.

Note that the presence of such transit costs may logically disincentivize ISP-level content caching, *e.g.*, [53]. However, poor transfer-delay performance due to a lack of content caching by the ISP may diminish end-user demand (including causing end-users to change to a competing ISP)[2] For the "Internet" setting, we assume that ISPs are not incentivized to cache content in the following, but we do model the effect of delay performance on demand.

Here, we generally assume consumers are, to some extent (for some delay-sensitive applications), willing to pay usage-based fees. Providers are then competing to settle on their usage-based prices, their goal being to maximize associated revenues. Note that a null price in the following does not mean a provider has no income, but rather that all their monthly revenues come from flat-rate priced service components. The study of the flat-rate regime is, however, out of the scope of this thesis; see [77, 93] for recent surveys of such issues.

7.1.2 Future Internet Architectures

In the past few years, several NSF Future Internet Architecture (FIA) projects [81] and EU projects (*e.g.*, [2]) have proposed dynamic management of content by the *network* layer, *i.e.*, Content-Centric Networking (CCN) [12, 40]; *e.g.*, eXpressive Internet Architecture (XIA) [10] (via use of their Content IDentifier (CID)[3]), and Named Data Networking [49]. Some of them leverage prior proposals for structured[4] and unstructured peer-to-peer file-sharing systems, and notions of indirection [100]. In CCNs, the end-users query the network with content identifiers that are typically hierarchically arranged (for scalable forwarding) and presume content "providers" (the publishers of a publish-subscribe system) who have sorted out semantic issues associated with content ontology.

[1] Recently, ISPs have also targeted advertising revenue of CPs [79] by filtering-out advertising from delivered content [88], presumably under the premise that such advertising was not explicitly requested (authorized) by the end-user.

[2] A similar trade-off occurs when large mirrored Content Distribution Networks (CDNs) connect to large ISPs: the ISP's customers benefit from increased proximity of content, but the ISP may lose "transit revenue" and anyway want the CDN, or any individual CP, to help pay for infrastructure costs associated with access to their customers, "Now what [the content providers] would like to do is use my pipes free, but I ain't going to let them do that because we have spent this capital and we have to have a return on it" [78]. See also [25, 75] regarding ISP infrastructure investment modeling and analysis.

[3] The XIA framework also includes service identifiers (SIA) and end-host identifiers (HID), the latter similar to existing IPv4 addresses.

[4] Which in turn leveraged DHTs used to manage some data centers.

In addition to (or possibly instead of) a hierarchical CID system, scalability for a publish-subscribe CCN can be achieved via limited scoping. For example, identifiers based-on/learned from local caching will only have local meaning. So, such identifiers could be reused spatially/horizontally, (this scoping could have physical significance in a geospatial wireless social network). Under identifier reuse, the possibility of "collision" could be made small when the identifier-space is large. Locally, the number of CIDs may be small so that forwarding could be feasibly based on a flat identifier space.

In the following for a future "Information" Centric Network (ICN), we will assume a coalition of ISP and content resolver/rendezvous-point, the latter selecting a CP or CDN for each end-user query. If these entities are in fact separate, fairly dividing revenue between them can be argued through the use of Shapley values, *e.g.*, [68, 69]. Since in this setting the ISP is *pulling* content, rather than the CP pushing content as in the (current) "Internet" setting described in the previous subsection, one can, by the same argument, expect that the CP should be compensated for their networking costs. So, for the ICN setting, we assume a reverse in side-payments polarity, from ISP to CP[5].

7.2 Problem Set-Up: The Internet model

Suppose there are two providers, one content (CP indexed 2) and the other access (ISP indexed 1), with *common* consumer demand-response [34][6]. First suppose that the demand response to price is linear:

$$D = D_{\max} - d(p_1 + p_2), \tag{7.1}$$

where d is demand sensitivity to the price, p_1 and p_2 are, respectively, the prices charged by the ISP and CP, and $D_{\max} > 0$ is the demand at zero usage based price[7]. Suppose the revenue of the ISP is

$$U_1 = (p_1 + p_s)D, \tag{7.2}$$

[5]Note that revenue from embedded advertising may be more fully shared in the ICN setting for the same reason.

[6]Leader-follower dynamics, rather than simultaneous play at the same time-scale, are considered in [**?**]. For the problem setting considered here, leader-follower dynamics were considered by us in [8] and provider competition in [21, 53].

[7]Note that ISPs are continuing to depart from pure flat-rate pricing (based on maximum access bandwidth) for unlimited monthly volume, *e.g.*, [95, 16].

where p_s is the side payment from content to access provider. Similarly, the revenue of the CP is

$$U_2 \quad = \quad (p_2 - p_s)D. \tag{7.3}$$

Consider a noncooperative game played by the CP and ISP adjusting their prices, respectively p_2 and p_1, to maximize their respective revenues, with all other parameters fixed. In particular, the fixed side-payment p_s is here assumed regulated. Note that the utilities are linear functions of p_s so that if p_s were under the control of one of the players, it simply would be set at an extremal value.

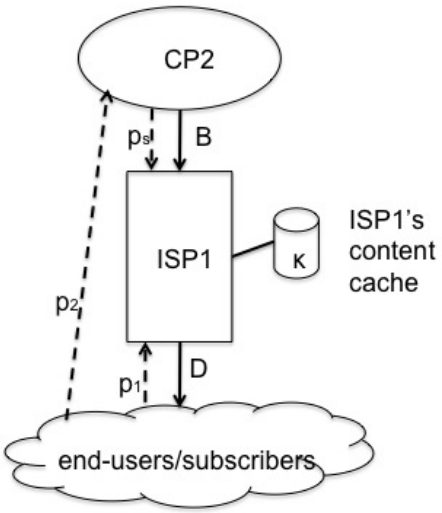

Figure 7.1. ISP and CP game on a platform of end-user demand-response

The following simple result was shown in [8, 21].

Theorem 1. *The interior Nash equilibrium*[8] *is*

$$p_1^* = \frac{D_{\max}}{3d} - p_s \quad \textit{and} \quad p_2^* = \frac{D_{\max}}{3d} + p_s$$

when

$$|p_s| \quad < \quad \frac{D_{\max}}{3d}, \tag{7.4}$$

[8]In this thesis, we do not consider boundary Nash equilibria, where at least one player is selecting an extremal value for one of their control parameters, often resulting in that player essentially opting out of the game, or maximally profiting from it at the expense of the other player. The boundary equilibria are also specified in [8].

with player utilities

$$U_1^*, U_2^* \quad = \quad \frac{D_{\max}^2}{9d}.$$

Note that this result allows $p_s < 0$, *i.e.*, net side payment is from ISP to CP (remuneration for content instead of access bandwidth). But in the Internet setting, we take $p_s > 0$, whether there is direct side-payment from CP to ISP (or, again, indirectly by payment through the peering contract between the residential ISP and the ISP of the CP - a contract that penalizes for asymmetric traffic exchange neutrally based on aggregate traffic volume).

In [21, 52], we showed that the ISP may actually experience a reduction in revenue/utility with the introduction of side payments, using a communal demand model that had different demand-sensitivity-to-price parameters d per provider type and also multiple providers of each type (*i.e.*, provider competition). Such a model was also considered in [9].

In [52], we used a convex, rather than linear, demand response to price, *e.g.*,

$$D \quad = \quad D_{\max}(1 - (p_1 + p_2)/p_{\max})^a, \tag{7.5}$$

where $a \geq 1$ and

$$p_{\max} \quad = \quad D_{\max}/d \;\text{ when } a = 1.$$

This model was motivated in [52] by considering two different types of users, as follows. Suppose that (user-designated) premium class-of-service (CoS) applications are

- delay sensitive,
- given service priority by the ISP over best-effort applications for the bandwidth B available between CP and ISP,
- subjected to usage-based charges by the ISP at price p_1.

Best-effort applications exploit reserved-but-unused bandwidth ($\leq B$) by the premium CoS applications, and unreserved bandwidth if any. So, some delay-sensitive applications may be content with under best-effort CoS when demand for premium CoS is low (hence reserved-but-unused bandwidth is high). Thus, as demand increases for premium CoS applications, say because price $p = p_1 + p_2$ reduces, there may be *additional demand* owing

to migration of delay-sensitive applications by more price-sensitive users who would otherwise tend to assign their delay-sensitive applications to best-effort CoS.

To better motivate this demand model, in Appendix A we derive a (more complex) price-convex demand response based on the delay-sensitivity of the usage-priced applications under consideration.

The following simple extension of Theorem 1 was shown in [52] by summing the first-order conditions $\partial U_i/\partial p_i = 0$, $i \in \{1, 2\}$, *cf.*, (7.7).

Theorem 2. *The interior Nash equilibrium for a strictly convex demand response D is*

$$p_1^* = p^*/2 - p_s \quad \textit{and} \quad p_2^* = p^*/2 + p_s, \tag{7.6}$$

where $p^* = p_1^* + p_2^*$ solves

$$2D(p^*) + p^* D'(p^*) \;=\; 0. \tag{7.7}$$

and $|p_s| < p^*/2$.

For the example of (7.5) with $a > 1$,

$$p^* \;=\; \frac{2}{2+a} p_{\max}, \tag{7.8}$$

$$U_1^*, U_2^* \;=\; \frac{p^*}{2} D(p^*) \;=\; \frac{D_{\max} p_{\max}}{2+a} \left(\frac{a}{2+a}\right)^a. \tag{7.9}$$

Again, under communal demand response with only one provider of each type, neither $p^* = p_1^* + p_2^*$ nor U_1^* depend on the side payment p_s.

7.3 ICN model

Again, in an ICN, residential users request content (or, more generally, information regarding application services) of the ISP/resolver, and the ISP/resolver decides the content provider. Therefore in an ICN, it's reasonable to assume that the side-payment is from ISP to CP, *i.e.*, $p_s < 0$. Also, the ISP is motivated to cache content, unlike for our simple Internet case, to reduce the side payment (*i.e.*, avoid paying for, *e.g.*, the networking costs of the ISP-selected CP to transmit the user-requested content). Suppose that the ISP decides to cache a fraction κ of the content and this results in lower delay between the CP and ISP, and a lower required side-payment to the CP, *cf.*, (7.11). If we model mean delay as $1/(B - D)$, where B is the service capacity between CP and ISP,

then with caching factor κ, this delay is reduced to $1/(B-(1-\kappa)D)$. For the model of Appendix B, the demand response:

- is increasing in caching factor κ,
- tends to convex in price as $\kappa \to 0$, and
- tends to linear in price as $\kappa \to 1$.

In the following for the ICN setting, we take the following simplified form of demand response than that of Appendix B *with these above properties*:

$$\begin{aligned} D &= D_{\max}(1-(p_1+p_2)/p_{\max})^{\kappa+(1-\kappa)a} \\ &= D_{\max}(1-(p_1+p_2)/p_{\max})^{a+\kappa(1-a)}. \end{aligned} \tag{7.10}$$

Note how in this model, neither $D_{\max}$ nor $p_{\max}$ are affected by κ, but *cf.* the linear demand model (7.14). Because of ISP caching, the ISP and CP utilities generalize to

$$\begin{aligned} U_1 &= (p_1+(1-\kappa)p_s)D - c(\kappa), \\ U_2 &= (p_2-(1-\kappa)p_s)D, \end{aligned} \tag{7.11}$$

again with $p_s < 0$, where $c(\kappa)$ is the cost of caching borne by the ISP.

We can use the results of Theorem 2 here, with parameters $(1-\kappa)p_s$ and $\kappa+(1-\kappa)a$ instead of p_s and a respectively, because the caching cost c component of U_1 does not depend on p_2 or p_1, and $|p_s| < p^*/2$ implies $|(1-a)p_s| < p^*/2$. We can conclude that the optimal utilities for ICN are

$$\begin{aligned} &U_1^* + c(\kappa),\ U_2^* \\ &= \frac{D_{\max}p_{\max}}{2+\kappa+(1-\kappa)a}\left(\frac{\kappa+(1-\kappa)a}{2+\kappa+(1-\kappa)a}\right)^{\kappa+(1-\kappa)a}. \end{aligned} \tag{7.12}$$

In the following section on numerical results, we consider performance at Nash equilibria as a function of κ (under the assumption that $|p_s| < p_{\max}/2$).

7.4 Numerical results

In this section, we give some numerical results for the models of communal CP/ISP demand given in the previous sections. Despite the fact that our models do not involve a lot of parameters, our aim is not a comprehensive numerical study over the entire

parameter space. Instead, we give some numerical results for parametric instances to show how optimal caching factors can be identified and comparisons made between the "Internet" and ICN scenarios described above. To this end, Figures 7.2-7.6 depict ISP utility $U_1^*/(D_{\max}p_{\max})$ with demand-exponent parameter $a = 2.0$. Figures 7.3-7.5 assume a caching cost that is polynomial in caching factor, *i.e.*, of the form

$$c(\kappa) \quad = \quad bD_{\max}p_{\max}\kappa^n,$$

where $b > 0$, while Figure 7.6 models caching cost as exponential in caching factor.

In Figure 7.2, $b = 0$ (*i.e.*, no cache cost, $c = 0$) and we see that U_1^* increases with caching factor κ. By (7.12), this figure also represents CP revenue U_2^* for the cases of Figures 7.2-7.6.

Figures 7.3 and 7.4 illustrate how linear cache cost ($n = 1$) leads to optimal $\kappa \in \{0, 1\}$: if $b \leq 0.04$ then optimal $\kappa = 1$, otherwise if $b \geq 0.05$ then optimal $\kappa = 0$ (the "Internet" case).

In Appendix C, we argue how $c(\kappa)$ is convex. Figure 7.5 shows how the ISP utility may be concave in κ for quadratic (convex) cache cost ($n = 2$) - here for $b = 0.05$, optimal $\kappa \approx 0.4$. Alternatively, we could consider a convex, exponential caching cost function

$$c(\kappa) \quad = \quad b_1 D_{\max}p_{\max}(\mathrm{e}^{b_2/(1-\kappa)} - \mathrm{e}^{b_2}), \tag{7.13}$$

where $b_1, b_2 > 0$. Figure 7.6 shows how the ISP utility may also be be concave - here for $b_1 = 0.05$ and $b_2 = 0.2$, optimal caching factor $\kappa \approx 0.5$.

Again, note that under the premise that ISP-level caching is not incentivized in for the (current) Internet setting, we can directly compare against the ISP utilities for the "Internet" case by simply using the ISP utilities at $\kappa = 0$ in these figures.

Finally, consider the simpler case of demand-response that is linear in price. We can take the caching factor κ as simply reducing the demand sensitivity to price (equivalently, increasing the maximum price for which there is non-zero demand):

$$D \quad = \quad D_{\max}\left(1 - \frac{p}{p_{\max}(1 + \sigma\kappa)}\right), \tag{7.14}$$

where $\sigma > 0$. Here, the results of Theorem 2 directly apply with $p_{\max}$ simply replaced by $p_{\max}(1 + \sigma\kappa)$ and $a = 1$ (or Theorem 1 with the demand-sensitivity d replaced by

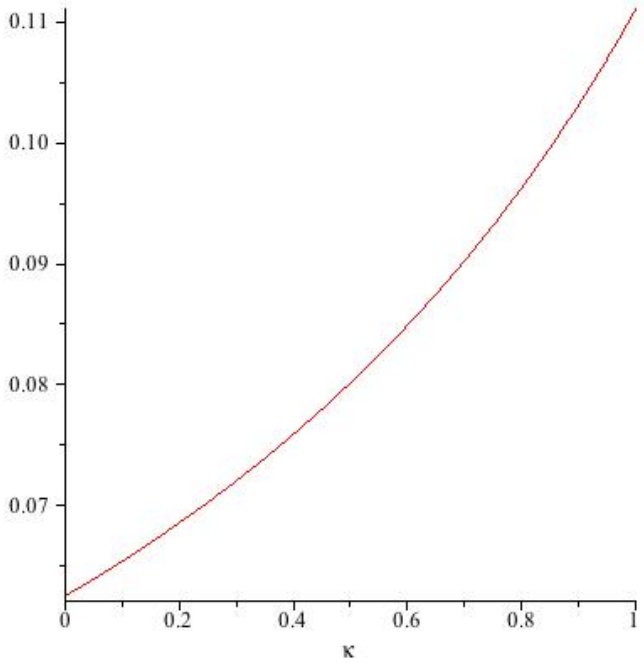

Figure 7.2. $U_1^*/(D_{\max}p_{\max})$ without caching cost

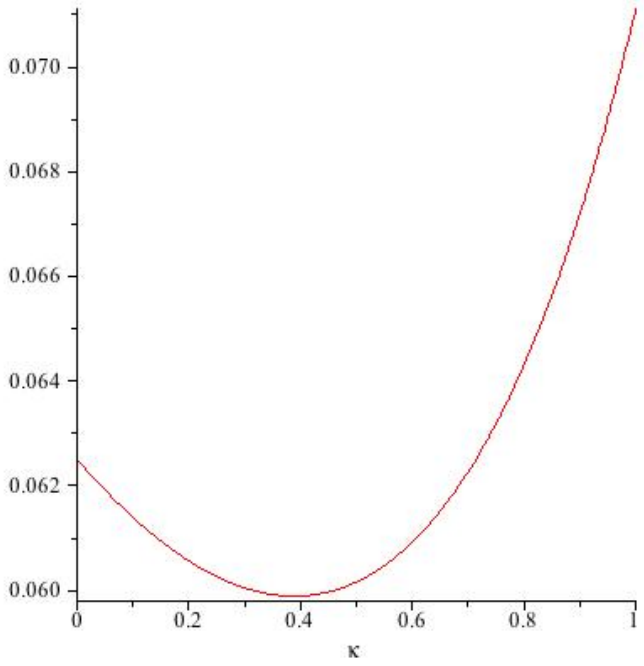

Figure 7.3. $U_1^*/(D_{\max}p_{\max})$ with linear caching cost, $b = 0.04$

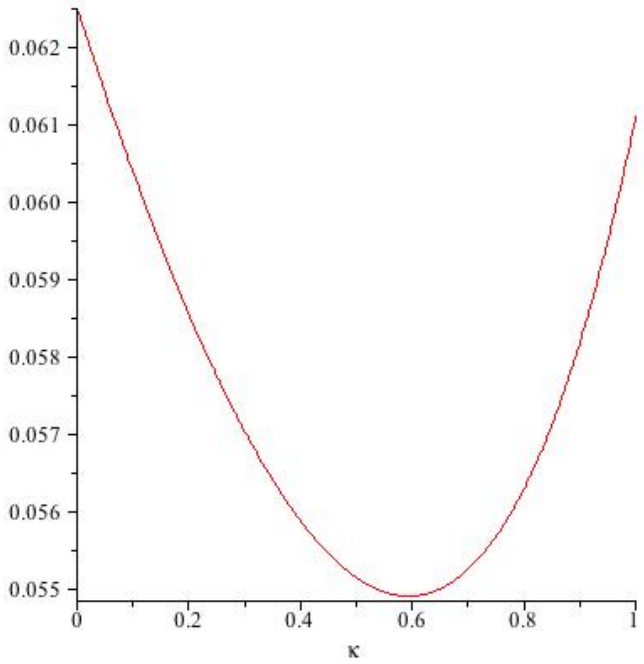

Figure 7.4. $U_1^*/(D_{\max}p_{\max})$ with linear caching cost, $b = 0.05$

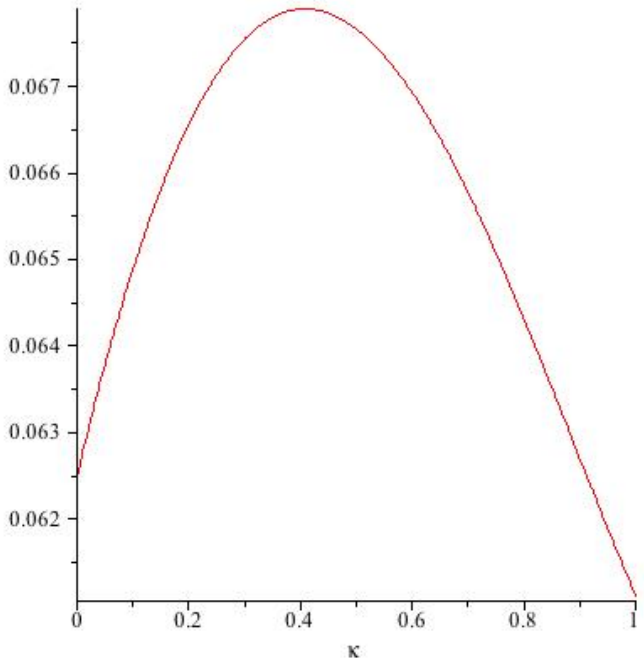

Figure 7.5. $U_1^*/(D_{\max}p_{\max})$ with quadratic caching cost, $b = 0.05$

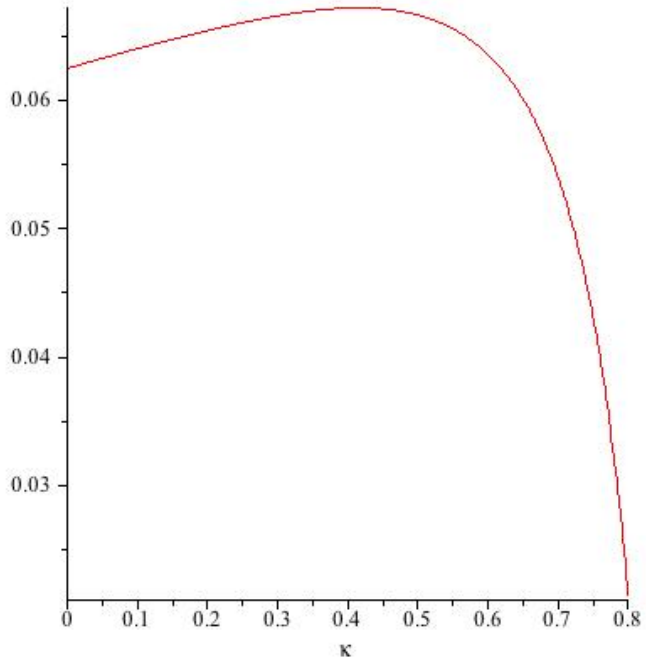

Figure 7.6. $U_1^*/(D_{\max}p_{\max})$ with exponential caching cost, $b_1 = 0.05, b_2 = 0.2$

$d/(1+\sigma\kappa)$). For quadratic caching cost, the ISP utility,

$$\frac{U_1^*}{D_{\max}p_{\max}} = \frac{1+\sigma\kappa}{9} - b\kappa^2.$$

is maximized when the caching factor is $\kappa^* = \min\{\sigma/(18b), 1\}$. So, when $\kappa^* < 1$, the concave, quadratic ISP utility U_1^* has maximal value

$$D_{\max}p_{\max}\left(\frac{1}{9} + \frac{\sigma^2}{18^2 b}\right).$$

Chapter 8

Conclusions

In this thesis, we contributed to two areas, namely anomaly detection and network neutrality. In both of the areas, our main focus was analysis of network traffic flows. The application of anomaly detection part in this thesis was basically about detection of anomalous flows in a network. In the net neutrality part, we dealt with pricing games under the existence of content caching and their implications to the net neutrality, which is a hot debate topic in the recent years. Both areas are related in the sense that the existence of infected end-users may be attracting malicious traffic to the network, which may lead ISPs to take precautions against those users, which in turn may result in non-neutral policies to be applied to them.

In our efforts contributing to anomaly detection, we considered detection of anomalous samples in a batch of collected samples. In our scenario, samples might be high-dimensional. But, the features which were most discriminative are a priori unknown. Our aim was to perform feature selection that will enable to successful detection of anomalies. We basically provided 2 different types of approaches to this problem. In both of the approaches, we utilized p-values for statistical significance assessment of the samples. The fundamental difference between the two approaches was that the first approach detected one sample at a time, whereas the second approach detected the anomalous samples in clusters. But, both of them may be particularly suitable when there is a latent anomalous class present in the data batch, discriminable from the known class using an (albeit unknown) small subspace of the full feature space.

In the sample-wise detection procedure new tests (formed by using the features) are used and included to the existing test set only when they yield lower (corrected) p-values than only using the previously existing set. This means existing test set size is growing

as more detections are made. This approach seeks to maximize the aggregate statistical significance of all detections up until a finite horizon. Before using and including a new test, considering the future detections (looking ahead) leads to better discrimination performance, while keeping the used test set small. And, this means that looking ahead enables the algorithm to perform effective feature selection. Our approach was compared, in area under the ROC curve, with several standard detection strategies for a network intrusion domain, detecting Zeus bot intrusion flows embedded amongst (normal) Web flows.

In the cluster-wise detection approach, we propose a procedure that aims to find the most outlier clusters of samples by assessing an approximate joint p-value (joint significance) for each candidate cluster. Our method effectively selects and uses the most discriminative features (by choosing a subset of the pairwise feature tests) to determine the clusters of anomalous samples in a given batch. We compared our approach with methods that use the p-values of individual samples but without clustering, and with the one-class SVM, which uses the feature vector directly. We proposed multiple feature representations and compared their advantages and disadvantages using different datasets in the experiments. We observed that, in detecting Zeus amongst Web, our p-value clustering algorithm, when used with low maximum test combination orders, with certain feature representations, and with sufficiently large training set, may outperform the tested alternative methods, which all make separate detection decisions for each sample, and which all use all of the features (tests). Limiting the test order improves the AUC performance, keeps the independent test assumption as valid as possible, and keeps the algorithm computationally feasible. P-value clustering method also performs better than the one-class SVM. Regarding to the comparison between our sample-wise and cluster-wise detection approaches, which of these approaches is more successful in discrimination depends on the dataset and (more importantly) feature representation. We observed that dataset dependence is based on 3 basic factors, which are port, training set size, and time of day. Port and time of day affect the type and diversity of the traffic. Training set size, along with these, determine how well-informed the null is. We observed that when the null is well-informed, p-value clustering methods tend to perform better since better informed nulls for the individual tests may give high discrimination power to the tests. When the null is poorly informed, approaches that use the features collectively (p-value sum and log sum) tend to outperform the feature selecting methods (p-value clustering). Also, the performance assessment criterion is crucial. We saw that in early detection performance, p-value clustering methods outperform others on the

average, which was not the case when area under ROC is used.

As future experimental work, there are other datasets available online for experimental purposes. For instance, ISCX datasets include botnet and non-malicious traffic datasets [96]. For background traffic, Ericsson Lab dataset can be used [101].

The second main area that this thesis contributed is network neutrality. We investigated the effects of content caching to the utilities and pricing policies of the entities in the Internet. The entities under consideration are ISPs, eyeball ISPs, CPs, and end-users. Game scenarios with different players are constructed and analyzed.

In the first model, we modeled the interaction of two different eyeball ISPs and explored the effect of differences in remote-content caching and demand on the net revenue from transit-traffic at ISP-to-ISP peering points. We considered slight modifications in this model. We changed the places of the congestion points, where the basic difference became the existence of a throughput limit downstream to the end-users. We found the Nash equilibrium points in these models. We have observed that the eyeball ISP allowing larger maximum demand will have larger demand even if this ISP's content is at the same price as the other ISP's content. This leads to larger gain for this ISP. It also means that this ISP might have some margin to increase its price. Hence, when we compare the prices of the ISPs at the Nash equilibrium, the ISP with larger possible demand has higher price than the other ISP. In addition to this, for the case where the downstream throughput to the end-users is limited, imposing a strict upper bound might lead to a linear increase in the utility until the high prices become so effective that even small throughput limit is not filled due to the users' price sensitivity.

The second model that is considered in this thesis about net neutrality is a game between a CP and an ISP on a platform of end-users served by both, which makes this a two-sided market. Two cases are analyzed here. One of them is the Internet case, where payment is in the same direction as traffic. So, the traffic that goes from CP to the ISP requires CP to pay ISP for carrying this traffic to the end-users. The other case is the Information-Centric Network, where CP transferring content to the ISP deserves payment from ISP to the CP. Here, content and payment are in opposite directions between CP and ISP. But, more importantly, in the ICN case, ISP is incentivized to cache content, whereas in the Internet case, it is not. We also made analyses under different assumptions about the caching cost. It is observed that without the caching cost, the ISP utility function is increasing as the caching factor increases. We also observed that how linear caching cost leads to optimal caching cost taking value either 0 or 1. There are also cases where ISP utility may be concave. These latter cases are

where the optimal values of the caching factors are fractional.

Appendix A

Explanation of convex demand response (Chapter 7)

We "implicitly" model the demand $D = [g(D)]^+$ with

$$g(D) = (D_{\max} - dp)\left(1 - \frac{\lambda}{B - D}\right) / \left(1 - \frac{\lambda}{B}\right), \tag{A.1}$$

where

- B is the bandwidth reserved *between CP and ISP* for delay sensitive applications paying usage-based prices,
- λ is demand sensitivity to mean delay, here modeled as $1/(B - D)$ (an expression for mean delay taken from the M/M/1 queue [111]).

Here, $\lambda > B - D$ results in zero demand D. That is,

$$D = [g(D)]^+.$$

Letting

$$\tilde{D} := (D_{\max} - dp)/(1 - \lambda/B) = D_{\max}(1 - p/p_{\max})/(1 - \lambda/B),$$

and assuming

$$\tilde{D} > 0,$$

we can find the interior fixed-point D of g^+ (*i.e.*, fixed point of g), giving the "explicit" demand response

$$D = \tfrac{1}{2}\left[(B + \tilde{D}) - \sqrt{(B - \tilde{D})^2 + 4\lambda\tilde{D}}\right]. \tag{A.2}$$

It's easy to see that this demand response has the following intuitive properties:

- $D \to D_{\max}$ as $B \to \infty$ and $p \to 0$
- D is a convex function of $\tilde{D}$ when $B > \lambda$, and hence also a convex function of price p (as assumed in [52, 53]).

There are obviously many alternative demand models with similar properties.

Appendix B

Explanation of convex demand response, increasing in caching factor (Chapter 7)

As a result of ISP caching, only a fraction $(1-\kappa)$ of the demand D is transmitted through the bandwidth B between ISP and CP. So, (A.1) is modified to

$$\begin{aligned} g_\kappa(D) &= (D_{\max} - dp)\left(1 - \frac{\lambda}{B-(1-\kappa)D}\right) / \left(1 - \frac{\lambda}{B}\right) \\ &= (D_{\max} - dp)\left(1 - \frac{\lambda/(1-\kappa)}{B/(1-\kappa)-D}\right) / \left(1 - \frac{\lambda/(1-\kappa)}{B/(1-\kappa)}\right) \end{aligned}$$

So, solving $D = g_\kappa(D)$ results in (A.2) with B and λ replaced by $B/(1-\kappa)$ and $\lambda/(1-\kappa)$, respectively:

$$D = \tfrac{1}{2}\left[(\frac{B}{1-\kappa} + \tilde{D}) - \sqrt{(\frac{B}{1-\kappa} - \tilde{D})^2 + 4\frac{\lambda}{1-\kappa}\tilde{D}}\right]. \tag{B.1}$$

So, as $\kappa \to 0$, the demand tends to (A.2), *i.e.*, convex in price. On the other hand, as $\kappa \to 1$, the demand tends to linear in price (7.1).

Since, $g_\kappa(D) = g_0((1-\kappa)D) := g((1-\kappa)D)$, is decreasing in $(1-\kappa)D$ (hence increasing in caching factor κ), the solution

$$D_\kappa = g_\kappa(D_\kappa)$$

is an increasing function of caching factor κ (in particular, $D_\kappa \geq D_0$). To see this, note

that

$$D_0 \;=\; g_0(D_0) \;\;<\;\; g_0((1-\kappa)D_0) \;=\; g_\kappa(D_0).$$

So, if $D_\kappa \leq D_0$, then we would have

$$D_\kappa \leq D_0 \;\;<\;\; g_\kappa(D_0) \;\leq\; g_\kappa(D_\kappa),$$

which contradicts the definition of D_κ in the first display above.

Appendix C

Convexity of cost of caching as a function of caching factor (Chapter 7)

Assume that the cost of caching is proportional to the number of cached items (content), in turn proportional to the (mean) amount of memory required to store them. For a fixed population of N end-users (a proximal group served by an ISP), let $\pi(j)$ be the proportion of the items that will soon be of interest to precisely j end-users. Finally, suppose the ISP naturally prioritizes its cache to hold the most popular content. So, a "caching factor" κ, based on all-or-none decisions to cache content of the same popularity, would satisfy

$$\kappa \quad \propto \sum_{j=N-f(\kappa)}^{N} j\pi(j).$$

for some $f(\kappa) \in \{0, 1, 2, ..., N\}$. The cost of caching would be proportional to the number of cached items, *i.e.*,

$$c(\kappa) \quad \propto \sum_{j=N-f(\kappa)}^{N} \pi(j).$$

Suppose that the great majority of potentially desired content is only minimally popular, *i.e.*, $\pi(j)$ is decreasing[1] We now argue that the caching cost $c(\kappa)$ is convex

[1]Note that this general assumption obviously accommodates the empirically observed Zipf distribution

and increasing for the simplified continuous scenario ignoring the (positive) constants of proportionality:

$$\kappa \;=\; \int_{N-f(\kappa)}^{N} z\pi(z)\mathrm{d}z \text{ and } c(\kappa) \;=\; \int_{N-f(\kappa)}^{N} \pi(z)\mathrm{d}z,$$

with $c(0) = 0$ and $c(1) = 1$. By differentiating successively, we get

$$\begin{aligned} 1 &= (N - f(\kappa))\pi(N - f(\kappa))f'(\kappa) && \text{(C.1)} \\ c'(\kappa) &= \pi(N - f(\kappa))f'(\kappa) \\ \Rightarrow 1 &= (N - f(\kappa))c'(\kappa) \\ \Rightarrow c''(\kappa) &= f'(\kappa)(N - f(\kappa))^{-2} && \text{(C.2)} \end{aligned}$$

Note that $f' > 0$ by (C.1) and therefore $c'' > 0$ by (C.2).

---

for content popularity, *e.g.*, [28].

Bibliography

[1] Comcast v. FCC. 600 F.3d 642 (D.C. Cir. 2010).

[2] Pursuit project. http://www.fp7-pursuit.eu/PursuitWeb/?page_id=177.

[3] Wireshark. `http://www.wireshark.org/`.

[4] In *First Economics and Technologies for Inter-Carrier Services (ETICS) Workshop Proceedings*, 2010.

[5] M. Adler, R. Sitaraman, and H. Venkataraman. Algorithms for optimizing the bandwidth cost of content delivery. Dec. 2011.

[6] P. Agyapong and M. Sirbu. Economic incentives in content-centric networking: Implications for protocol design and public policy. In *Proc. 39th Telecommunications Policy Research Conference*, Arlington, VA, 2011.

[7] T. Ahmed, B. Oreshkin, and M. Coates. Machine learning approaches to network anomaly detection. In *Proc. SysML*, 2007.

[8] E. Altman, P. Bernhard, S. Caron, G. Kesidis, J. Rojas-Mora, and S. Wong. A study of non-neutral networks under usage-based pricing. 2011.

[9] E. Altman, A. Legout, and Y. Xu. Network non-neutrality debate: An economic analysis. In *Proc. IFIP Networking*, 2011.

[10] A. Anand, F. Dogar, D. Han, B. Li, H. Lim, M. Machado, W. Wu, A. Akella, D. Andersen, J. Byers, S. Seshan, and P. Steenkiste. Xia: An architecture for an evolvable and trustworthy internet. In *Proc. ACM HOTNETS*, Cambridge, MA, 2011.

[11] R. Arbogast and D. Kaut. FCC chairman lays out net neutrality plan, avoids title ii, but walks tightrope. Dec. 1 2010.

[12] A. Baid, T. Vu, and D. Raychaudhuri. Comparing alternative approaches for networking of named objects in the future Internet. In *Proc. INFOCOM Computer Communications Workshops*, Mar. 2012.

[13] T. Berners-Lee. Net neutrality: This is serious. June 2006.

[14] J. Binkley and S. Singh. An algorithm for anomaly-based botnet detection. In *USENIX SRUTI 2006*, pages 43–48, July 2006.

[15] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2007.

[16] K. Bode. AT&T to impose caps, overages. Available at: http://www.dslreports.com/shownews/Exclusive-ATT-To-Impose-Caps-Overages-113149, Mar. 13 2011.

[17] J.-C. Bolot and P. Hoschka. Performance engineering of the World Wide Web: Application to dimensioning and cache design. pages 1397–1405, 1996.

[18] F. Bornstaedt, M. Roettgermann, F. Johansen, and H. Lønsethagen. The sending party network pays.

[19] F. Bretz, T. Hothorn, and F. Westfall. *Multiple comparisons using R*. CRC Press, 2011.

[20] B. Caberera, B. Ravichandran, and R. Mehra. Statistical traffic modeling for network intrusion detection. In *Proc. the 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 466–473, San Francisco, CA, 2000.

[21] S. Caron, G. Kesidis, and E. Altman. Application neutrality and a paradox of side payments. In *Proc. ACM ReArch*, Nov. 30, 2010. See also http://arxiv.org/abs/1006.3894.

[22] Z. Celik, J. Raghuram, G. Kesidis, and D. Miller. Salting public traces with attack traffic to test flow classifiers. In *Proceedings CSET USENIX Workshop*, 2011.

[23] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(Article no: 15), 2009.

[24] H. Cheng, S. Bandyopadhyay, and H. Guo. The debate on net neutrality: A policy perspective. 22:60–82, Mar. 2011.

[25] K. Cheng, S. Bandyopadhyay, and H. Gon. The debate on net neutrality: A policy perspective. June 2008.

[26] P. Chhabra, C. Scott, E. Kolaczyk, and M. Crovella. Distributed spatial anomaly detection. In *IEEE INFOCOM*, 2008.

[27] R. Chong. The 31 flavors of net neutrality. 12, 2008.

[28] G. Dan and N. Carlsson. Power-law revisited: A large scale measurement study of P2P content popularity. In *Proc. IPTPS*, 2010.

[29] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[30] A. Dhamdhere and C. Dovrolis. Can ISPs be profitable without violating "network neutrality"? In *Proc. ACM NetEcon*, Seattle, 2008.

[31] Q. Ding and E. Kolaczyk. A compressed pca subspace method for anomaly detection in high-dimensional data.

[32] R. Douville. Etics architecture(s). In *Second Economics and Technologies for Inter-Carrier Services (ETICS) Workshop*, June 2011.

[33] R. Duda, P. Hart, and D. Stork. *Pattern classification*, volume 2. Wiley New York:, 2001.

[34] N. Economides. Net neutrality: Non-discrimination and digital distribution of content through the internet. *I/S: A Journal of Law and Policy*, 4(2):209–233, 2008.

[35] L. Ertoz, E. Eilertson, A. Lazarevic, P.-N. Tan, V. Kumar, J. Srivastava, and P. Dokas. The MINDS - Minnesota intrusion detection system. In *Next Generation Data Mining*. MIT Press, 2004.

[36] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data. In *Data Mining for Security Applications*, 2002.

[37] P. Faratin, D. Clark, P. Gilmore, S. Bauer, A. Berger, and W. Lehr. Complexity of Internet connections. In *Proc. 35th TPRC*, 2007.

[38] M. Feily, A. Shahrestani, and S. Ramadass. A survey of botnet and botnet detection. In *IEEE Third International Conference on Emerging Security Information, Systems and Technologies*, pages 268–273, 2009.

[39] P. Ganley and B. Allgrove. Net neutrality: A user's guide. 22:454–463, 2006.

[40] A. Ghodsi, T. Koponen, J. Rjahalme, P. Sarolahti, and S. Shenker. Naming in content-oriented architectures. In *Proc. ACM SIGCOMM ICN*, Toronto, Aug. 2011.

[41] J. Goldsmith and T. Wu. *Who Controls the Internet: Illusions of a Borderless World*. Oxford University Press, 2006.

[42] M. Graham and D. Miller. Unsupervised learning of parsimonious mixtures on large spaces with integrated feature and component selection. *IEEE Trans. on Signal Processing*, 54:1289–1303, 2006.

[43] C. B. Group. Characterizing network-based applications. `http://www.cl.cam.ac.uk/research/srg/netos/brasil/data/index.html`.

[44] G. Gu, R. Perdisci, J. Zhang, and W. Lee. Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proc. USENIX Security*, pages 139–154, 2008.

[45] R. Hahn and S. Wallsten. The economics of net neutrality. *Economists' Voice*, 3(6):1–7, 2006.

[46] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11:10–18, 2009.

[47] K. A. Heller, K. M. Svore, A. D. Keromytis, and S. J. Stolfo. One class support vector machines for detecting anomalous windows registry accesses. In *Proc. of the Workshop on Data Mining for Computer Security*, 2003.

[48] S. Holm. A simple sequentially rejective multiple testing procedure. *Scandinavian Journal of Statistics*, 6:65–70, 1979.

[49] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard. Networking named content. In *Proc. 5th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, pages 1–12, 2009.

[50] C. Kang. FCC approves net-neutrality rules; criticism is immediate.

[51] G. Kesidis. Congestion control alternatives for residential broadband access by cmts. In *Proc. IEEE/IFIP NOMS*, Osaka, Japan, Apr. 2010.

[52] G. Kesidis. Side-payment profitability under convex demand-response modeling congestion-sensitive applications. In *Proc. IEEE ICC*, Ottawa, June 2012.

[53] F. Kocak, G. Kesidis, and S. Fdida. Network neutrality with content caching and its effect on access pricing. In S. Sen, C. Joe-Wong, S. Ha, and M. Chiang, editors, *Smart Data Pricing*. John Wiley & Sons, 2013.

[54] F. Kocak, G. Kesidis, T.-M. Pham, and S. Fdida. The effect of caching on a model of content and access provider revenues in information-centric networks. *ASE Science Journal*, 2(3), 2013.

[55] F. Kocak, G. Kesidis, T.-M. Pham, and S. Fdida. The effect of caching on a model of content and access provider revenues in information-centric networks. In *ASE/IEEE International Conference on Economic Computing (EconCom)*, Washington D.C., Sept. 2013.

[56] F. Kocak, D. J. Miller, and G. Kesidis. Detection of hidden anomalous classes in network traffic using categorical and continuous features. *journal paper in preparation.*

[57] F. Kocak, D. J. Miller, and G. Kesidis. Detecting anomalous latent classes in a batch of network traffic flows. In *Annual Conference on Information Sciences and Systems (CISS)*, March 2014.

[58] L. B. N. Laboratory and ICSI. LBNL/ICSI Enterprise Tracing Project. `http://www.icir.org/enterprise-tracing`.

[59] S. V. Labs. VRT Labs - Zeus Trojan Analysis. `http://labs.snort.org/papers/zeus/html`.

[60] A. Lakhina, M. Crovella, and C. Diot. Characterization of network-wide anomalies in traffic flows. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 201–206, 2004.

[61] W. Lee and D. Xiang. Information-theoretic measures for anomaly detection. In *Proc. IEEE Symposium on Security and Privacy*, pages 130–143, 2001.

[62] E. Lehmann and J. Romano. *Testing statistical hypotheses*. Springer, 2005.

[63] B. Li, J. Springer, G. Bebis, and M. Gunes. A survey of network flow applications. 36(2):567–581, Mar. 2013.

[64] W. Li and A. Moore. A machine learning approach for efficient traffic classification. In *Proc. of IEEE MASCOTS*, 2007.

[65] K. Limthong and T. Tawsook. Network traffic anomaly detection using machine learning approaches. In *IEEE Network Operations and Management Symposium (NOMS)*, pages 542–545, 2012.

[66] W. Lu, M. Tavallaee, G. Rammidi, and A. Ghorbani. Botcop: An online botnet traffic classifier. In *Communication Networks and Services Research Conference*, pages 70–77, 2009.

[67] E. Lundin and E. Jonsson. Anomaly-based intrusion detection: privacy concerns and other problems. 34(4):623–640, 2000.

[68] R. Ma, D.-M. Chiu, J. Lui, V. Misra, and D. Rubenstein. Interconnecting eyeballs to content: A shapley value perspective on isp peering and settlement. In *Proc. Workshop on Economics of Networked Systems (NetEcon)*, 2008.

[69] R. Ma, D.-M. Chiu, J. Lui, V. Misra, and D. Rubenstein. On cooperative settlement between content, transit and eyeball internet service providers. In *Proc. ACM CoNEXT*, 2008.

[70] J. MacQueen. Some methods for classification and analysis of multivariate observations, 1967.

[71] B. Maggs. Presentation on CDNs (Akamai). http://blog.lrem.net/2013/05/23/rescom-2013-bruce-maggs/, May 2013.

[72] M. Mantere, M. Sailio, and S. Noponen. Network traffic features for anomaly detection in specific industrial control system network. 5(4):460–473, 2013.

[73] D. J. Miller, F. Kocak, and G. Kesidis. Sequential anomaly detection in a batch with growing number of tests: Application to network intrusion detection. In *IEEE Intl. Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2012.

[74] M. Molina, I. Paredes-Oliva, W. Routly, and P. Barlet-Ros. Operational experiences with anomaly detection in backbone networks. 31(3):273–285, 2012.

[75] J. Musacchio, G. Schwartz, and J. Walrand. A two-sided market analysis of provider investment incentives with an application to the net-neutrality issue. *Review of Network Economics*, 8(1), 2009.

[76] P. Njoroge, A. E. Ozdaglar, N. E. Stier-Moses, and G. Y. Weintraub. Investment in two sided markets and the net neutrality debate. In *Columbia Business School DRO (Decision, Risk and Operations)*, Oct 2012.

[77] A. Odlyzko. Network neutrality, search neutrality, and the never-ending conflict between efficiency and fairness in markets. 8, 2009.

[78] B. Online. At SBC, it's all about scale and scope. Nov. 5 2005.

[79] R. C. P. Hande, M. Chiang and S. Rangan. Network pricing and rate allocation with content provider participation. In *Proc. IEEE INFOCOM*, 2009.

[80] K. M. P. OKane, S. Sezer and E. Gyu. SVM training phase reduction using dataset filtering for malware detection. 8(3), March 2003.

[81] J. Pan, S. Paul, and R. Jain. A survey of the research on future Internet architectures. July 2011.

[82] R. Parker, A. Melathopoulos, R. White, S. Pernal, M. Guarna, and L. Foster. Ecological adaptation of diverse honey bee (apis mellifera) populations. 5(6), 2010.

[83] V. Paxson. Bro: a system for detecting network intruders in real-time. 31(23):2435–2463, 1999.

[84] T.-M. Pham, S. Fdida, and P. Antoniadis. Pricing in information-centric network interconnection. In *Proc. IFIP Networking*, Brooklyn, NY, 2013.

[85] L. Portnoy, E. Eskin, and S. Stolfo. Intrusion detection with unlabeled data using clustering. In *Proc. ACM CSS Workshop on Data Mining Applied to Security (DMSA)*, pages 5–8, Philadelphia, PA, 2001.

[86] E. Pouyllau. Presentation at ARC MANEUR meeting, INRIA, Paris, May 2011.

[87] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD Conference*, pages 427–438, Dallas, TX, 2000.

[88] A. Robertson. French government tells ISP to stop installing ad-blocking software on its modems. Jan. 7 2013.

[89] M. Roesch. Snort: Lightweight intrusion detection for networks. In *LISA*, volume 99, pages 229–238, 1999.

[90] K. Ross. Hash-routing for collections of shared web caches. Nov-Dec 1997.

[91] J. Ryan, M.-J. Lin, and R. Miikkulainen. Intrusion detection with neural networks. In *Proc. Workshop on AI Approaches to Fraud Detection and Risk Management, AAAI Press*, pages 72–77, 1997.

[92] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[93] S. Sen, C. Joe-Wong, S. Ha, and M. Chiang. Pricing data: A look at past proposals, current plans, and future trends. 2012.

[94] Z. Shan and X. Wang. Growing grapes in your computer to defend against malware. 9(2), Feb. 2014.

[95] A. Shin. Who's the bandwidth bandit? Available at: http://blog.washingtonpost.com/thecheckout/2006/10/bandwidth_bandit.html, Oct. 4 2006.

[96] A. Shiravi, H. Shiravi, M. Tavallaee, and A. Ghorbani. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. 31(3):357–374, May 2012.

[97] T. Shon and J. Moon. A hybrid machine learning approach to network anomaly detection. pages 3799–3821.

[98] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *IEEE Symposium on Security and Privacy (SP)*, pages 305–316, 2010.

[99] V. Sotiris, P. Tse, and M. Pecht. Anomaly detection through a bayesian support vector machine. 59(2), 2010.

[100] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet indirection infrastructure. In *Proc. ACM SIGCOMM*, pages 73–86, 2002.

[101] G. Szab, D. Orincsay, S. Malomsoky, and I. Szab. On the validation of traffic classification algorithms. In *9th International Conference on Passive and Active Network Measurement*, pages 72–81, 2008.

[102] A. Tanenbaum and D. Wetherall. *Computer Networks.* Prentice Hall, 5th edition, 2010.

[103] D. Trossen and A. Kostopoulos. Techno-economic aspects of information-centric networking. 2:26–50, 2012.

[104] G. Trunk. A problem of dimensionality: a simple example. *IEEE Trans. Patt. Anal. and Mach. Intell.*, 1:306–307, 1979.

[105] V. Valancius, C. Lumezanu, N. Feamster, R. Johari, and V. Vazirani. How many tiers? pricing in the internet transit market. In *Proc. ACM SIGCOMM*, 2011.

[106] G. Venkatesh and N. Nadarajan. Http botnet detection using adaptive learning rate multilayer feed-forward neural network. In I. Askoxylakis, H. Phls, and J. Posegga, editors, *Information Security Theory and Practice. Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems*, volume 7322 of *Lecture Notes in Computer Science*, pages 38–48. Springer Berlin Heidelberg, 2012.

[107] P. Waldmeir. The net neutrality dogfight shaking up cyberspace. Mar. 23 2006.

[108] K. Wang, C.-Y. Huang, S.-J. Lin, and Y.-D. Lin. A fuzzy pattern-based filtering algorithm for botnet detection. 55:3275–3286, 2011.

[109] Y. Wang, J. Wong, and A. Miner. Anomaly intrusion detection using one class SVM. In *Proc. from the Fifth Annual IEEE SMC Information Assurance Workshop*, pages 358–364, 2004.

[110] D. Weller and B. Woodcock. Bandwidth bottleneck: The hardware at the heart of the Internet is not fast enough. Jan. 2013.

[111] R. Wolff. *Stochastic modeling and the theory of queues*. Prentice-Hall, Englewood Cliffs, NJ, 1989.

[112] C. Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

[113] T. Wu. Network neutrality, broadband discrimination. 2:141, 2003.

[114] P. Wurzinger, L. Bilge, T. Holz, J. Goebel, C. Kruegel, and E. Kirda. Automatically generating models for botnet detection. In *14th European Conference on Research in Computer Security (ESORICS)*, 2009.

[115] P. H. M. C. D. T. Y. Wu, H. Kim. Revenue sharing among isps in two-sided markets. In *Proc. IEEE INFOCOM Mini Conference*, Shanghai, 2011.

[116] L. Yan, R. Dodier, M. Mozer, and R. Wolniewicz. Optimizing classifier performance via the Wilcoxon-Mann-Whitney statistics. In *Proc. of Intl. Conf. on Machine Learning*, 2003.

[117] H. Zeidanloo, A. Manaf, P. Vahdani, F. Tabatabaei, and M. Zamani. Botnet detection based on traffic monitoring. In *International Conference on Networking and Information Technology (ICNIT)*, 2010.

[118] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and A. Vasilakos. An effective network traffic classification method with unknown flow detection. 10(2), 2013.

[119] J. Zhang, R. Perdisci, W. Lee, X. Luo, and U. Sarfraz. Building a scalable system for stealthy P2P-botnet detection. 9(1), Jan. 2014.

[120] D. Zhao, I. Traore, A. Ghorbani, B. Sayed, S. Saad, and W. Lu. Peer to peer botnet detection based on flow intervals. In D. Gritzalis, S. Furnell, and M. Theoharidou, editors, *Information Security and Privacy Research*, volume 376 of *IFIP Advances in Information and Communication Technology*, pages 87–102. Springer Berlin Heidelberg, 2012.

[121] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, and D. Garant. Botnet detection based on traffic behavior analysis and flow intervals. 39:2–16, 2013.

[122] G. Zou, G. Kesidis, and D. Miller. A flow classifier with tamper-resistant features and an evaluation of its portability to new domains. *IEEE Journal on Selected Areas in Communications*, 29(7):1449–1460, 2011.

Vita

Fatih Kocak

Education:

- 2010 – 2014: The Pennsylvania State Univ. (PSU), Electrical Engineering
  PhD, supervised by Professors George Kesidis and David J. Miller
- 2007 – 2010: Bilkent University, Electrical and Electronics Eng., Ankara, Turkey
  MS, supervised by Professor Sinan Gezici
- 2003 – 2007: Bilkent University, Electrical and Electronics Eng., Ankara, Turkey
  BS, senior project supervised by Professor A. Enis Cetin
- 1996 – 2003: Adnan Menderes Anatolian High School, Aydin, Turkey

Experience:

- Aug. 2010 – April 2014: Research and Teaching Assistant in PSU
  RA: Machine learning and data mining algorithms design and implementation (applications especially to network intrusion detection), network neutrality, peer-to-peer caching
  TA: Operating Systems (Computer Science), discrete-time linear systems and communication systems courses (Electrical Eng.)
- Sep. 2007 – June 2010: RA and TA in Electrical and Electronics Eng., Bilkent University
  RA: Time-delay estimation in cognitive radio and multiple-input multiple-output (MIMO) systems
  TA: Probability, microprocessors, and analog electronics courses
- July 2007 – Dec. 2008: Engineer in Military Communications, Systems Engineering Department of Communications Division of ASELSAN Inc., Ankara, Turkey
  Design and integration of military communication systems, dealing with medium access control layer, network layer and transport layer issues. Expertise on a variety of basic Internet protocols and concepts
- Dec. 2006 – July 2007: Student (part-time) engineer in Command, Control and Weapon Systems, Systems Engineering Department of Microwave System Technologies Division of ASELSAN Inc., Ankara, Turkey
  Image and video processing applications (image and video fusion)
- June 2006 – July 2006: Intern in Research and Development Dept. of TRT (National public broadcaster of Turkey)
  Optimization of bandwidth and gain of FM transmitters
- June 2005 – July 2005: Intern in Command, Control and Weapon Systems, Systems Engineering Department of Microwave System Technologies Division of ASELSAN Inc., Ankara, Turkey
  Image processing applications (pattern recognition)