ANALYSIS OF ALTO PROTOCOL OVER P2P NETWORKS

A Thesis

by

Koray Kökten

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Electrical and Electronics Engineering

Özyeğin University
October 2012

ANALYSIS OF ALTO PROTOCOL OVER P2P NETWORKS

Approved by:

Associate Prof. Dr. M. Oğuz Sunay, Advisor
Department of Electrical and Electronics Engineering
Özyeğin University

Assistant Prof. Dr. İsmail Arı
Department of Computer Science
Özyeğin University

Professor Dr. Tanju Erdem
Department of Electrical and Electronics Engineering
Özyeğin University

Dr. Ahmet Serdar Tan
Turk Telekom R&D

Date Approved: 22 October 2012

Assistant Prof. Dr. Ali Özer Ercan
Department of Electrical and Electronics Engineering
Özyeğin University

To my beloved Grandpa,

My first friend and teacher...

ABSTRACT

Currently, for distributed applications such as Peer-to-Peer (P2P) and Content Delivery Networks (CDNs), the most important challenge is to determine the optimal peer or node selection process, since it sometimes causes low Quality of Experience and affects the economics of the Internet Service Providers (ISPs) negatively. Therefore, in recent years significant research has been conducted in this area. However, since these protocols operate on Layer 7 (Application Layer) according to the Open Systems Interconnection (OSI) model, they are network-oblivious. Therefore, without a co-operation between service or network providers (*i.e. ISPs*) and applications, obtaining an optimal solution is not easy or even possible. Since P2P applications are one of the most widely used applications on the Internet, Internet Engineering Task Force (IETF) has started to work on a protocol, commonly referred to as the Application Layer Traffic Optimization (ALTO) Protocol, in order to enable a standartized interface between applications and network providers. With this protocol, both applications and service providers have a chance to interact using a common interface via the ALTO Server and share the necessary information in order to calculate the costs of communication between peers, so that a better-than-random peer or node selection is possible.

In this thesis, we implement an ALTO Server for several ISPs, describe several (either novel or from the literature) cost calculation methods and try to improve the performance of the communication network from the perspectives of both applications and service providers. We analyse the implemented ALTO Server using a P2P BitTorrent-Like file sharing application, a P2P real-time scalable video streaming application and a CDN application running on a Software Defined Network (SDN) with an OpenFlow Controller. Simulation results and demos show that, with carefully designed peer selection algorithms used in the ALTO Service, the performance of the applications can be sustained (even can be improved), while the inter-ISP traffic rates can be reduced dramatically.

ÖZETÇE

Günümüzde eş görevli (P2P) ve içerik dağıtım ağlarında (CDN) en önemli problem uygulamaların servis kalitesini ve servis sağlayıcıların ekonomilerini kötü yönde etkilemesi nedeniyle en iyi eş ya da düğümün seçilmesi problemidir. Bu nedenle son yıllarda bu problemi çözmek ya da iyileştirebilmek adına araştırmalar yapılmakta çeşitli öneriler sunulmaktadır. Ancak bu uygulamalar açık sistemler arabağlaşımı (OSI) modeline göre 7. katmanda çalıştıkları için, üzerinde çalıştıkları ağın bilgilerine sahip değildirler. Dolayısıyla, uygulamalar ile servis sağlayıcılar arasında bir işbirliği olmadan en uygun çözümü bulmak çok zor hatta imkansızdır. Tüm bu nedenlerden ve ayrıca bu uygulamaların internet üzerinde en çok trafik yaratan uygulamalar olmasından dolayı İnternet Mühendisliği Çalışma Grubu (IETF), Uygulama Katmanı Trafik Eniyilemesi (ALTO) protokolü adı verilen ve, uygulamalar ile servis sağlayıcılar arasında bir arayüz tanımlayan bir protokol oluşturmaya başlamıştır. Bu protokolle birlikte uygulamalar ile servis sağlayıcılar, ALTO Sunucusu olarak adlandırılan ortak bir arayüzde haberleşme ve bilgi paylaşma imkanı bulacak ve paylaşılan bu bilgilerle potansiyel eşler arasındaki maliyetler hesaplanarak, rastgele bir eş eşleştirmeden daha iyi ve hatta en iyi eş eşleştirme yapılabilecektir.

Bu tezde servis sağlayıcılar için, literatürde var olan ve yeni önerilen maliyet hesaplama metodları geliştirerek, bir ALTO Sunucusu gerçeklendi. Bu maliyet hesaplama yöntemleri ile hem uygulamaların performanslarının iyileştirilmesi, hem de servis sağlayıcıların ağ kaynaklarının daha verimli kullanılması amaçlanmıştır. Geliştirilen bu ALTO Sunucusu sırasıyla, BitTorrent benzeri bir dosya paylaşımı uygulamasında, P2P Gerçek-Zamanlı Ölçeklenebilir Video Dağıtımı uygulamasında ve son olarak OpenFlow Kontrolör ile kontrol edilen Yazılım Tabanlı Ağ (SDN) üzerinde gerçeklenen

CDN uygulaması üzerinde analiz edilmiştir. Benzetim sonuçları göstermiştir ki, ALTO servisinin bu uygulamalar için aktif hale getirilmesi servis sağlayıcılar arası trafiği önemli ölçüde azaltmış, aynı zamanda uygulamaların performanslarını da korumuştur.

ACKNOWLEDGEMENTS

First and foremost, I am grateful to Assoc. Prof. Dr. M. Oğuz Sunay, my advisor, for his understanding and courteousness during this two years. Through my graduate school, i have realized so many times that he is a brilliant and innovative academician and this thesis would not be completed wihtout his helps.

I would like to thank to all of my instructors, especially Assist. Prof. Dr. Ali Özer Ercan for his motivation and effort on lectures and us during the semesters.

I also would like to thank to İ. Serkan Kırkgül, my friend and colleague, for his friendship and helps through this mostly tough and stressful two years.

Last of all to you my precious family, who always support and encouraged me to reach the best i can. Love you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Internet is a giant network that is composed of thousands (38,2901 as of October 2011 [1]) of active networks. It has been changing the way of people communicate to each other, as well as their social life in a dramatical way. Therefore, the amount of data consumed by Internet users has been growing continuously during the last decades and reaching massive amounts. According to an Internet report [2], the world wide traffic in 2015 will reach 116,539 petabytes per month. The rapid increase in bandwidth and computational powers as well as reduced prices of both hardware and software technologies, all helped a huge amount of content flow on the Internet. This increased demand paved the way towards a need for a better Internet infrastructure. There are countless number of emerging applications that require transmission of web page content, multimedia content *etc.* If we look at the distribution of this traffic in terms of applications we see that the Peer-to-Peer (P2P) file sharing and multimedia transmission (either Real-Time and Video-on-Demand (VoD)) applications are among the dominant types of traffic on the Internet, as illustrated in Figure 1 [3]. Also, among several types of multimedia applications, the most bandwidth consuming ones are related with video transmission. Recent studies estimate the share of video traffic over the Internet and it is expected to be the main source of the Internet traffic [4][5], in the near future. YouTube has a data volume exceeding 45 terabytes and it has reached over 3 billion views per day [6]. Since this amount of data is not possible to being served with just one or multiple servers, these type of applications works on commercial Content Delivery Networks (CDNs). Therefore at that point, it is important to define what P2P and Content Delivery Networks are and what they

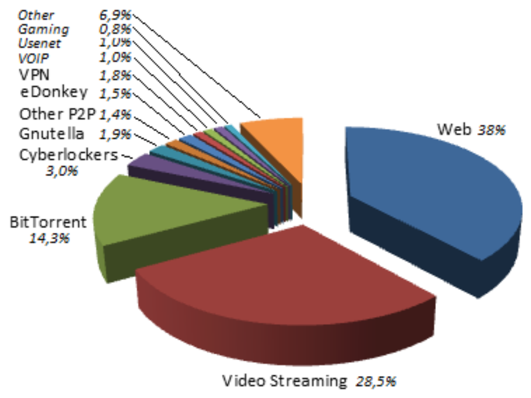


Figure 1: Percentage Traffic Distribution of Different Applications on Internet [3]

both stand for in todays Internet. P2P networks are distributed systems where the software running at each node provides equivalent functions. A formal definition of peer-to-peer networking is "a set of technologies that enable the direct exchange of services or data between computers" [7]. These systems enable a virtual or logical network connection between end-points in which end-point are addressable and provides connectivity, routing and messaging between them. Therefore, these type of systems are also called overlay networks that run on top of the Internet [8]. The logical connection on top of a physical connection between users is show in Figure 2. These networks provide a decentralized, scalable, self-organized and stable system. In

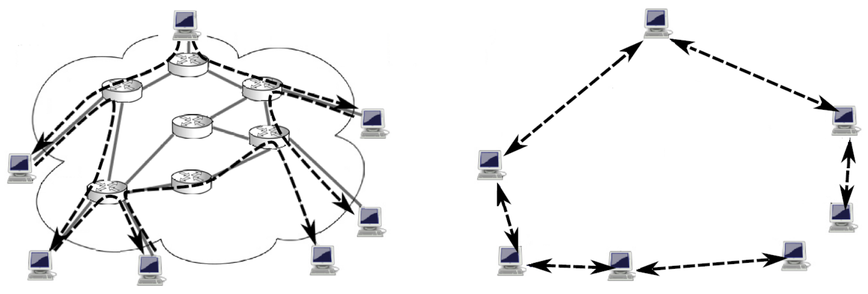


Figure 2: a)Physical Connection b) Logical Connection Between End-Users

contrast to the traditional Client-Server (CS) connection model, peers are equal, and P2P systems emphasize sharing among these equals. A pure P2P system runs without any centralized control or hierarchical organization, called unstructured overlay. An unstructured overlay is “an overlay in which a node relies only on its adjacent nodes for delivery of messages to other nodes in the overlay. Example message propagation strategies are flooding and random walk” [9]. On the other hand, a structured overlay is: “an overlay in which nodes cooperatively maintain routing information about how to reach all nodes in the overlay” [7]. Compared to unstructured overlays, structured overlays provide a limit on the number of messages needed to find any object in the overlay. The details for both of the overlaying methods will be given in Chapter V. In both of the approach, peers can represent clients, servers, routers, or even networks.

On the other hand, a CDN represents a group of geographically distributed servers deployed to simplify the distribution of content generated by Web publishers in a timely and efficient manner [10]. These servers include the same content which is formed by the customers of the CDN enterprises. Although the prior definition of a content delivery network is simplistic, it tells us a significant amount of information about what a CDN represents. To accomplish this task, servers must be located closer to the ultimate consumers or potential clients since it provides customers to experience a good QoE.On the other hand, just like any other network architectures, there are both advantages and disadvantages of CDN architectures. One of the most significant advantage of it is to minimize (in theory) the amount of traffic on overall Internet, by placing the replica content near to end-users, as well as reducing latency and providing scalability. However, deploying or purchasing such a service may come to enterprises cost-inefficient, especially to small compaines due to high costs of it.

As we mentioned above, P2P and multimedia streaming applications which are installed on a CDN infrastructure generate the significant part of the overall traffic. One of the advantage of P2P and CDN systems comes from redundancy in resource availability and network scalability. However, in these applications inefficient source selection has been causing cross-domain (or Inter-ISP) traffic at high volumes. This excessive cross-domain traffic rates may lead to serious disruption on Internet service provider (ISP) economics. In other words, a typical internet flow inevitably travels over multiple networks from its source to destination [11]. Although the capacity at the edges allow high communication rates, travelling over multiple networks results in congestion at peering points and this congestion substantially decreases the overall capacity of the network. Moreover, when TCP flows couple with congestions at peering points, TCP retransmissions caused by multiple round trips impose a serious performance bottleneck, particularly for Real-Time applications. Due to these factors, locality of peers promises a significant potential for the optimization of the network resource utilization. For example, in recent studies [12][13] on Skype, the authors found that many universities (aka edge ISPs) are hosting a large number of Skype super nodes. Thus, they handle a large amount of transient traffic from and then to their providers, violating valley-free routing and leading to substantially higher operational cost.

1.1 Background

All of these problems related with selecting optimal peers have directed the researchers to develop several smart peer selection algorithms, those usullay include a new entity for supporting additional informations to the applications. The results of these studies show that, with these additional provided informations for use in peer selection can improve P2P performance and lower ISP costs [14][15][16].

Currently, in the literature there are several proposals which are considering with this problem from the point of ISPs view and applications view, respectively. For instance in [17], authors suggest a P2P Inter-AS traffic limiting algorithm focusing on the PoP points of the ASes (typically ISPs). However, this method depends on the packet labeling (or packet inspection), but since every P2P application uses different packet structure and since they encrypt the packets, a general solution or algorithm is not easy to develop if not impossible.

Also several studies [18][19][20] demonstrate by looking at the point of ISPs view that, deploying caches by the network providers in the variable locations of the network may reduce the traffic volume routed in the overall topology. Although the results of these studies are promising, P2P traffic caching can be application specific, and many ISPs are reluctant to get involved in the distribution of contents of some P2P applications, due to some copyright issues.

Additionaly in [21], authors suggest a new and scalable method to biased peer selection without requiring any new infrastructure and any co-operation between ISPs and applications, and it depends on on the fact that CDNs aim to minimize the latency for any transmission[22]. In real life, CDN infrastructures contain the same replica of a content more than one geographically different locations, so any node is redirected to only one replica of the content via dynamic Domain Name System (DNS) service which is the closest one to the corresponding end-point. Therefore, from the observations of CDN redirections, if two peers exhibit similar redirection behavior, they are likely to be close to one another. Further it can be said that, these peers will be mostly within the same ISP, thus avoiding cross-ISP traffic and optimizing clients performance by avoiding most network bottlenecks as well. Finally, when a P2P application needs to make a peer selection, first it needs to obtain these informations from all of the peers. Having received these informations any peer has an ability to make a geographically close peer selection to reduce the latency and cross-domain traffic. The redirection informations are saved in a map called *ratio-map* by all of the peers. The structure of this *ratio-map* for a peer P_a is formed as folows;

$$\mu_a = < r_1 = f1,\ \ r_2 = f2\ \ ...\ \ r_i = fi >$$

where r_i is the replica server name and f_i is the ratio of time that a peer is redirected to the replica server r_i. Therefore, if two peers have the same or similar *ratio-map* informations, then the distance between them should be short. Similarly, if two peers have different *ratio-map* values, then the autonomous system number between these peers should be much more. It is important to note that, *ratio-map* has a vector structure. Therefore, the structure of this map can be used for determining the nearby peers based on the *cosine similarity* formulation which is shown in (1). *Cosine similarity* is a measurement of how similar two vectors to each other [23], in a scale [0, 1]. Finally, given two peers a and b the *cosine similarity* formula can be revised as;

$$cos\_sim(a,b) = \frac{\sum\limits_{i \in I_a} (\mu_{a,i} \cdot \mu_{b,i})}{\sqrt{\sum\limits_{i \in I_a} \mu_{a,i}^2 \cdot \sum\limits_{i \in I_b} \mu_{b,i}^2}} \tag{1}$$

where I_a and I_b represents the set of replica servers to which peer a and b has been redirected over the time window. The *cosine similarity* metric is analogous to taking the dot product of two vectors and normalizing the result. When two maps are identical, their resulting *cosine similarity* value is equal to 1, and when they are orthogonal to each other (i.e., have no replica servers in common), the result is going to be equal to 0 [21]. According to the performance results, this proposed approach commits a better performance than non-revised application's in terms of Inter-ISP traffic. However and unfortunately, while this solution reduces the Inter-ISP traffic volumes, it on the other hand worsened the application's performance since it reduces the average download rate. The main reason why this problem occured is that, this approach does not consider any congestion issues of the peers and the corresponding links. In other words, this approach only considers with the locality of the peers in order to select the optimal ones.

In [24], authors propose with an innovative approach that depends on the cooperation of both network providers and applications. In this approach, authors define a new entity which is called *iTracker* and it is implemented by the network providers.

This entity is an interface between network providers and applications for sharing the necessary informations in order to make an efficient peer selection. The main framework of this entity is shown in Figure 3 below.

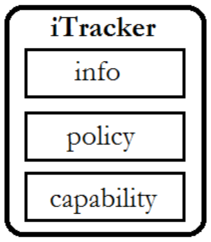


Figure 3: iTracker Framework

In this framework *info* interface allows application trackers or (in pure P2P systems) peers get the topological informations and mappings. For instance, for a given IP addresses *info* interface maps an IP address to *ASID, PID, LOC* tuple, where *ASID* is the ID of the network provider, *PID* is an abstract ID of the peer and LOC is the topological information of the peer such a geographical coordinate of it. On the other hand, the *policy* interface allows application trackers or peers to obtain policies and guidelines of the network. Policies specify how a network provider would like its network to be utilized at a high level, typically regardless of P2P applications. Finally, the *capability* interface allows peers or application trackers, to request network providers capabilities. For example, a network provider may provide different classes of services or on-demand servers in its network. Then, an application tracker may ask iTrackers in popular domains to provide such servers and then use them as peers to accelerate P2P content distribution. After this brief expalanation of the framework, let us define the working mechanism of the overall system with a sample topology shown in Figure 4, below.

In this example there are two separate network providers which are Network Provider A and Network Provider B, and they employ two different iTackers which

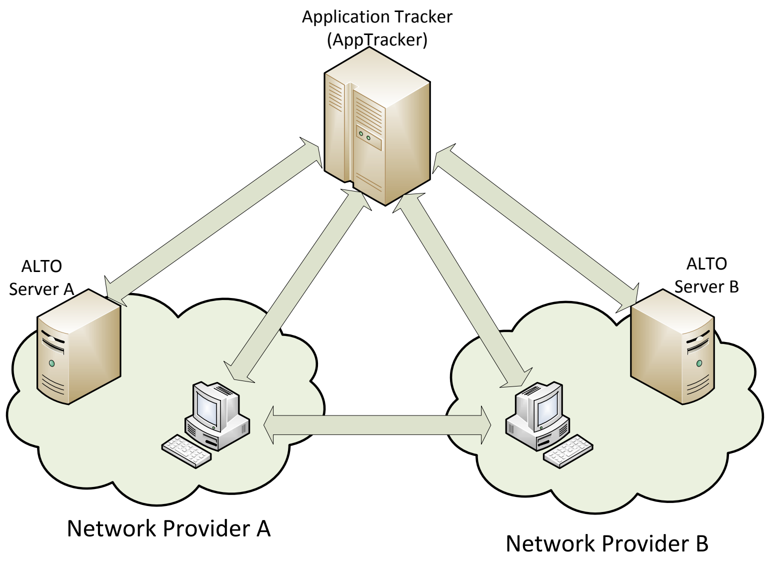


Figure 4: An Example of Obtaining *info* Guidelines from iTrackers

include all of the necesaary information to share with the P2P overlay. In this architecture, first of all, peers get the *info* informations from iTrackers' *info* interface. When a peer wants to join and request a peer list in order to start downloading a content, it shares the corresponding information obtained from the corresponding iTracker, with the tracker of the application. Once getting these informations, application tracker creates a peer list by using these informations to create a biased peer list.

In this model, assigning *info* attributes to the peers is done by considering with a bi-level optimization problem that minimizes the maksimum link utilization (MLU) from the point of service providers' view as well as maximizes system throughput in term of applications. According to the results of this study, not only the Inter-ISP traffic rates are reduced, but also applications' performances are also enhanced by reducing the data download completion time as well. At this point it is important to note that, although this solution seems promising, it depends on the cooperation of both ISPs (with deploying iTrackers) and applications (by using *info* guidelines) in any way.

1.2 Contributions of the Thesis

After all of these efforts, Internet Engineering Task Force (IETF) has started to work on a protocol under a group which is called Application Layer Traffic Optimization (ALTO) for defining a new entity called ALTO Server on the existing network to support necessary topology and other important informations about the underlying network, to the distributed applications such as P2P and CDN to make a better-than-random peer or node selection.

In order to achieve that goal, ALTO protocol defines two types of map service in the ALTO service, which are called Network Map and Cost Map. Under these directories distributed applications can achieve topology informations of the corresponding network as well as the cost values between end-points. It is important to note that, the way of forming network map and the corresponding cost map is left to the companies that implement and sell it in order to develop their innovative ideas and create a difference. By regarding these informations in mind, in this study we implemented a location based Network Map interface and a minimum delay based Cost Mapping service in order to enhance the performance parameters of the applications and reduce the inter-ISP traffic rates as well. The details about these map interfaces are given the following chapters. We simulate our ALTO service in a BitTorrent-Like file sharing applications, a P2P real-time scalable video streaming application and a CDN application that operates on a Software Defined Network (SDN) with OpenFlow controller, respectively. The simulation and demo results show us that, using ALTO protocol not only decrease the cross-domain traffic rates of the service providers dramaticaly, in the mean time it also provides quite similar performance results in terms of applications.

1.3 Summary

This thesis is organized as follows. In Chapter II, ALTO protocol is introduced with its different entities and functions. In Chapter III we will focus on the design issues and challenges about foming an optimal Map Service in ALTO service, and a novel cost calculation method which is based on the delay parameter is going to be explained. Then in Chapter IV, we propose our BitTorrent-Like file sharing simulator and also by enbaling ALTO service for it, we discuss the performance results both in terms of underlying network and the application. Similarly, In Chapter V we evaluate a P2P real-time scalable video streaming application and discuss the performance result of it with different ALTO service enabled and ALTO-free scenarios. Then In Chapter VI, we introduce Software Defined Networking (SDN) explain a use case of ALTO service on this type of networking with a CDN application. Finally in Chapter VII, conclusions of the thesis are drawn and future research directions are proposed.

CHAPTER II

APPLICATION LAYER TRAFFIC OPTIMIZATION (ALTO) PROTOCOL

ALTO is one of the on-going protocol specification process under a IETF Working Group which has the objective of defining “an information-sharing service that enables applications to perform better-than-random peer selection” [25]. This working group started to to study on ALTO, by first defining the problem statement in October, 2009 and the protocol is highly and precisely formed currently. The main focus of ALTO protocol is actually shaping the traffic of P2P applications, since they generate a considerable amount of network traffic in todays Internet. However, ALTO may be useful for non-P2P applications as well. For example, vendors of CDN applications may use information provided by ALTO to select one of several servers or information replicas. As an another example, ALTO information can also be used to select a media relay needed for NAT traversal. The goal of these informed decisions is to improve performance in the application while reducing resource consumption in the underlying network infrastructure. Some applications that can benefit from ALTO are:

-**File Sharing Applications:** In P2P file sharing applications, the content which is interested, generally available more than one user. Therefore, ALTO service may provide information either users and applications in order to make more efficient user selection according to underlying network.

-**Cache/Mirror Selection:** Most of the web content is usually served by the geographically distributed caches and mirrors for scalability and load balancing issues. However, todays Internet, selection of the proper mirror/cache for a given user is based on inaccurate geo-location data, on proprietary network-location systems, or often delegated to the user herself. Therefore, ALTO protocol can be an option for an automated way to choose the propoer cache or mirror.

-**Live Media Streaming Applications:** P2P applications for live video streaming allow users to receive multimedia content produced by one source and targeted to multiple destinations, in a real-time or near-real-time way. Peers often participate in the distribution of the content, acting as both receivers and senders. The goal of an ALTO solution is to help a peer to find effective communicating peers that exchange the media content.

- **Real-Time Communication Applications:** P2P real-time media streaming is an emerging technology and it enables real time audio, video and text. In this approach, just like P2P file sharing, real time content flows between users continuously. Unfortunately, most of the users on the Internet is behing NATs, firewalls *etc.*. Therefore, some other elements are (*e.g.* relays) are made active in the different locations of the topology. Thus an ALTO server could help the users to select best relay in order to achieve desirable QoE performance.

-**Distributed Hash Tables:** In pure P2P systems, lookup and search functionalities are done with Distributed Hash Tables (DHTs). ALTO protocol can also provide significant informations for various DHT algorithms as well.

2.1 Protocol Scope

It is important to remind that, ALTO protocol is a mechanism, that conveys the information about the underlying network. Therefore, ALTO service serves an "my-Internet View" of the network. In this terminology, the network can be regarded as an Autonomous System, an ISP, a smaller region or set of ISPs. In particular, ALTO service defines the network endpoints and generates generic costs among them. Figure 4 shows the overall system architecture of the framework. In this architecture, an ALTO Server prepares ALTO Information; an ALTO Client uses ALTO Service

Discovery to identify an appropriate ALTO Server; and the ALTO Client requests available ALTO Information from the ALTO Server using the ALTO Protocol.

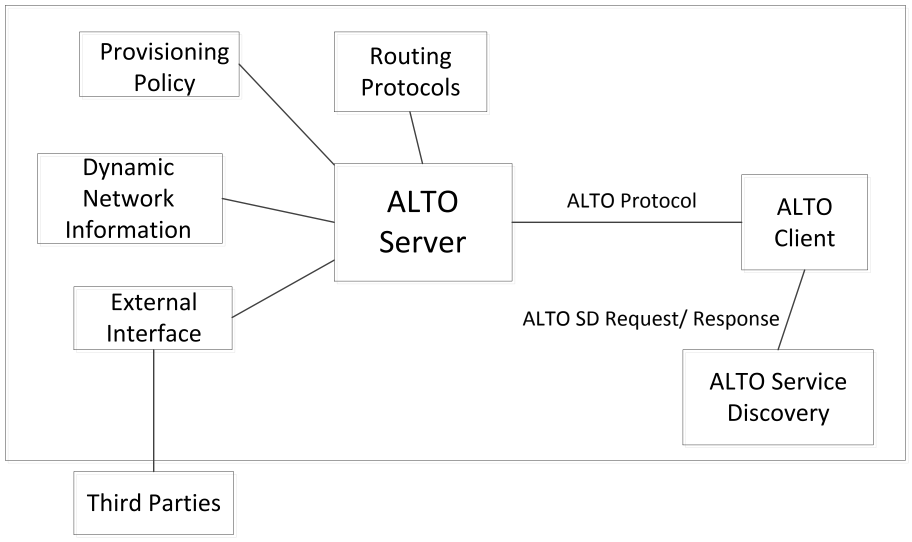


Figure 5: Basic ALTO Architecture

It is a fact that, Internet is a fastly evolving network that the state and connection scheme of that enormous AS has been changing permanently. Therefore, the ALTO information provided by the ALTO Server should be updated as well. At that point, the important question is how often this update mechanism should be processed, and it is left to the developers. More specifically, the ALTO information provided by an ALTO Server may be affected by other systems such as (but are not limited to) static network configuration databases, dynamic network information, routing protocols, provisioning policies, and interfaces to outside parties. These components are shown in the Figure 5 for completeness, but it is important to note that ALTO protocol does not specify how ALTO protocol is going to use such these informations obtained from these third-parties. Note that it may also be possible for ALTO Servers to exchange network information with other ALTO Servers (either within the same administrative domain or another administrative domain with the consent of both parties) in order to adjust exported ALTO Information.

2.2 *Protocol Structure*

By considering with extensibility issue, ALTO protocol employs a simple and single framework to obtain the network informations. As we mentioned before, the main aim of the ALTO service is to provide network location information and path information between the corresponding locations to the ALTO clients. With that in mind, in Figure 6, the general ALTO protocol structure is shown below. Furthermore, as can be understand from the name of the protocol, ALTO works at the application layer of standard OSI model. Therefore, it is built on a common transport protocol, messaging structure and encoding, and transaction model. The protocol is subdivided into services of related functionality. ALTOCore provides the Server Information Service and the Map Service to provide ALTO Information. Other ALTO Information services can provide additional functionality. There are three such services defined in [26]; Map Filtering Service, Endpoint Property Service, and Endpoint Cost Service.

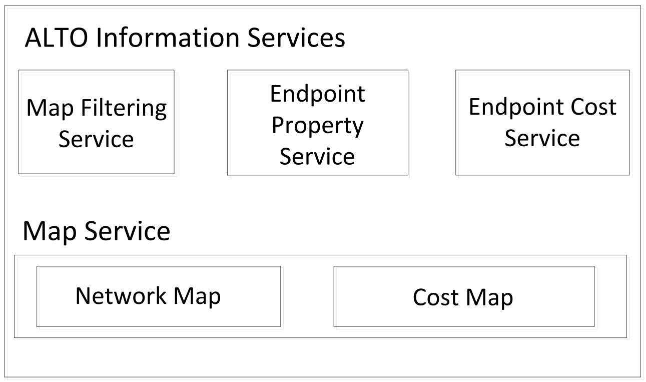


Figure 6: ALTO Protocol Structure

ALTO Information Services: Multiple, distinct services are defined to allow ALTO Clients to query ALTO Information from an ALTO Server.

- Map Service: This service provides a few informations to ALTO Clients in the form of Network Map and Cost Map. The Network Map provides the full set of network location groupings defined by the ALTO Server and the endpoints contained with each grouping. The Cost Map provides costs between the defined groupings. These two maps can be realized as a simple files with an encoding mechanism provided by the ALTO Server.

- Map Filtering Service: ALTO Clients may query for the Network Map and Cost Map informations based on couple of additional parameters. This service allows ALTO Service to serve to taht clients with an appreciate way.

- Endpoint Property Service: This service allows ALTO Clients to look up properties for individual endpoints. An example endpoint property is its network location (its grouping defined by the ALTO Server) or connectivity type (*e.g.*, ADSL, Cable, or FTTH).

- Endpoint Cost Service: This service allows to ALTO Server to calculate and return the corresponding costs (either numerical and ordinal) between different endpoints.

At that point let us give a couple of detailed information about the Map Service of the protocol since the most important interface of the protocol is Map Service.

2.3 Network Mapping

The first subdirectory under Map Service is called Network Map. As we mentioned before, Network Map service addresses all of the users of the underlying service provider's network to different set of groups or clusters. In other words, if we talk in terms of IP addresses rather than end-users, any possible public IP address must be addressed to a cluster whether the corresponding IP address is belong to the corresponding service provider or not. Having grouped all of the IP addresses into several clusters, every cluster is assigned to a unique identification number, called *PID*. In Figure 7 below, we see a sample Network Map, which has addressed all of the possible public IP addresses to different PID numbers. Finally, this network map information should be requested by the ALTO client, periodacally, since it has a potential to being updated.

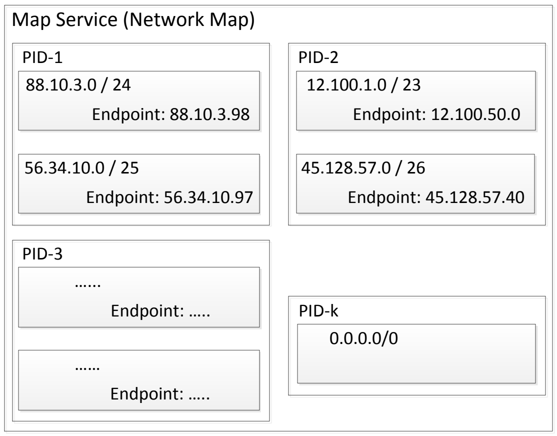
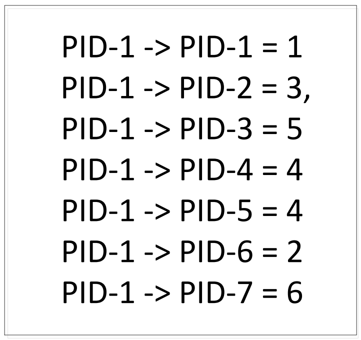


Figure 7: Map Service Structure

According to this Network Map, it is seen that for example, the IP adrresses between 88.10.3.0/24 and 88.10.3.98, which corresponds to 98 IP address in total and, the IP addresses between 56.34.10.0/25 and 56.34.10.97 that corresponds to 97 IP addresses are assigned to PID-1 cluester by the ALTO service. By following this way and notation, other public IP addresses are assigned to other PID clusters as well. As an important detail we see in figure that, PID-k cluster has a notation like 0.0.0.0/0. This entry demonstrates that, if there is any possible IP address which is not assigned to any PID cluster, then the PID cluster of that interested IP address is considered as PID-k, and it can be called as default PID cluster. In every Network Maps, which are generated in different ALTO Servers, it is likely to see this default PID cluster. At that point, it is necessary to indicate that, there is not any restriction about how to determine the IP address ranges for assigning to different PID clusters or what should be the number of IP addresses which are addressed to each PID, or the algorithm that determines the PID number of each IP address. All of these open issues are the design parameters of the protocol's implementors, in practice.

2.4 Cost Mapping

Cost Map service on the other hand defines and calculates the "cost values" between all of the PID clusters, as can be intuitively understood from its name. There are two important attributes of this service which are "Cost Type" and "Cost Mode". First, Cost Type indicates what the cost represents for an ALTO client (*e.g.* hop-count, air miles, routing cost). On the other hand, Cost Mode specifies how the cost is shared in Cost Map interface. There are two types of definition of the cost mode which are "numerical" and "ordinal". In numerical cost mode, PID clusters are ranked (not necessarily unique) from the point of a specific source PID cluster, and these ranked values are used for selecting convenient PID clusters and so peers. Similarly, in the numerical cost mode, the calculated cost values are sent to the ALTO client those are safe to perform numerical operations (*e.g. summation of the costs*). In both of the value types, lower cost value represents higher preference for an ALTO client. After all of these informations, let us give an example about how a cost map is created and represented. For instance, we have seven PID groups which are PID-1, ..., PID-7 and they include variable number of IP addresses (not important for that case). Under this scenario, when an ALTO client (*e.g.* tracker of a P2P application) requests a ordinal Cost Map information for PID-1 cluster and if it makes a request without no Cost Map filtering, Cost Map service returns with a repsonse like shown in Figure below. In this example, we see that all of the PID groups are ranked according to PID-1 as source. Also, as can be seen in figure again, PID-1$\rightarrow$PID-4 and PID-1$\rightarrow$PID-5 costs are equal to each other, which is possible according to protocol as we mentioned above, as well.



PID-1 -> PID-1 = 1
PID-1 -> PID-2 = 3,
PID-1 -> PID-3 = 5
PID-1 -> PID-4 = 4
PID-1 -> PID-5 = 4
PID-1 -> PID-6 = 2
PID-1 -> PID-7 = 6

Figure 8: An Ordinal Cost Map Example

2.5 Use Case of ALTO Service for Tracker-Based P2P Application

In this section, we propose an example use case of ALTO protocol. Since P2P applications are very popular today's Internet, we give our use case for this application. Although native P2P overlaying idea depens on a non-centralized model, today's most of the P2P applications employ a centralized entity which is called *tracker* (*e.g.* BitTorrent). What tracker does is, it keeps the registry of the all peers in the overlay as well as sends candicate peer list to the reuester peers. By regarding this information, in this type of P2P applications we can easily say that, the entity which communicates with ALTO Server as an ALTO client is going to be tracker. Hereby, a P2P tracker obtains and locally stores ALTO information (the Network Map and Cost Map) from the ISPs containing the P2P clients, and benefit from the same aggregation of network locations done by ALTO Servers.

Figure 9 shows an example use case where the P2P tracker is an ALTO Client and applies ALTO information when selecting peers for its P2P clients. The example proceeds as follows: (1). First, as an ALTO client, P2P tracker requests the Network Map that includes all PID numbers from the ALTO Server. Since this Network Map incudes all of the IP prefixes, P2P tracker can easily address all of the peer IP

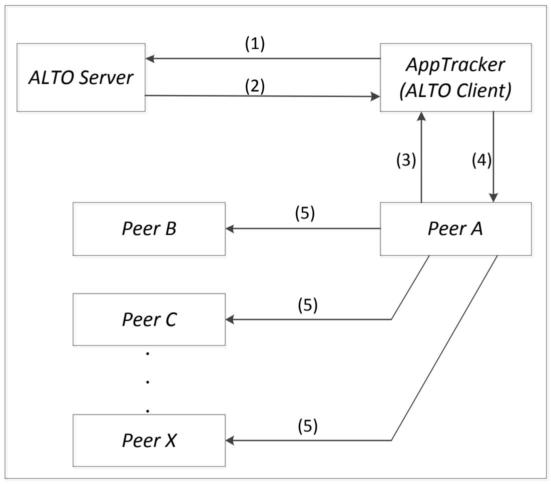


Figure 9: ALTO Client embedded in P2P Tracker

addresses into PID numbers.

(3). When a P2P client joins the swarm, it first registers and requests a peer list from P2P tracker.

(2).Then, P2P Tracker requests the Cost Map from the ALTO Server, for a specific source PID cluster.

(4). Having received the peer list request, P2P tracker sends a peer list to the P2P client which is created by considering with Network Map and Cost Map informations. It is also an alternative for P2P tracker to use only Network Map infomation in order to choose peers that are belong to the same ISP with the requester peer as well.

(5). Finally, P2P Client connects to a subset of the received peers.

In real life, it is likely for a P2P tracker to choose peers that are connected to multiple ISPs. In such a case, P2P tracker needs to connect and request the Network Map and Cost Map informations from the corresponding ALTO Servers as well.

CHAPTER III

PID GROUPING, COST CALCULATION AND PEER SELECTION ISSUES ON ALTO PROTOCOL

3.1 Design Trade-Offs of Map Service

The key issue in ALTO service is how to constitute the Map Service which includes Network Map and Cost Map subdirectories in it, since it is the most effective entity on the overall protocol in terms of performance of it, load on the service *etc.*. and since different vendors can create and implement their algorithms for those mapping services by obeying the general rules of the protocol. Therefore, in this chapter we give several important details about what are trade-offs of choosing different design parameters. After explaining these issues, we propose the Network Mapping and Cost Mapping algorihms that we have used in our simulations.

First of all, there are several fundamental questions that are needed to be answered about the constitution of the PID clusters and calculation of the cost values between PID clusters. These are for instance;

1) What should be the number of PID clusters?

2) What should be the size of PID clusters in terms of IP address number?

3) What should be the clustering algorithm which is going to be used to put the IP's into different clusters? (*e.g.* location based, bandwidth based, mobility based *etc.*)

4) What should be the cost calculation algorithm based on ? (*e.g.* delay, end-to-end bandwidth, delay jitter *etc.*)

5) How should the cost be calculated between PID's that have several IP addresses inside them, with the determined algorithm?

6) What should be the period of obtaining Cost Map and Network Map by an ALTO Client?

As can be remembered, PID clustering method is recommended for obtaining the scalability issue instead of using the unique IP addresses of all of the end-users. Therefore, if we look at the issue in terms of scalablity, the most optimal solution is to put all of the IP addresses into just one PID cluster or a couple of PID clusters. However this approach brings additional problem with itself. For instance, with lower number of PID cluster number, the cost calculation process may be more complitaced and non-accurate. Let us explain this issue with an example. Suupose that we have 2 PID clusters which are determined by looking at the locations of the end-users, and each of them has 3 and 4 end-users in them, as shown in Figure 10 below.

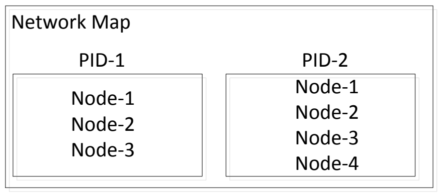
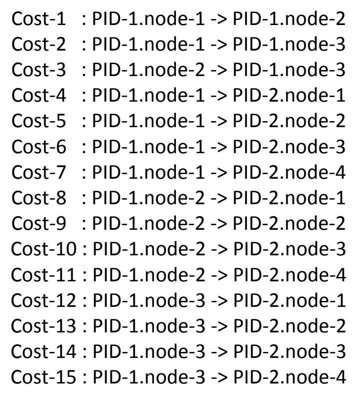


Figure 10: Sample Network Map

At that stage, suppose that a cost map is going to be calculated with a cost calculation function (*e.g.* end-to-end bandwidth), regarding as the PID-1 is the source PID. Therefore, the following cost values PID-1→PID-1, PID-1→PID-2 need to be calculated. However, since all of the end-users do not have the same attributes even those in the same cluster, there may be two approach for calculating the cost value between clusters. First, all of the potentially possible connection costs between all of the peers in PID-1 and PID-2 are calculated and some fucntion of these costs can be redarded as the final cost value. This idea can be regarded as a ideal approach for cost accuracy, but it needs too much processing time. Because even in this small network we need to calculate 15 different cost values in order to calculate the PID-1→PID-1, PID-1 → PID-2 cost values. In brief, in order to obtain high accuracy, we need to sacrifice from our time and processing power.



Cost-1	: PID-1.node-1 -> PID-1.node-2
Cost-2	: PID-1.node-1 -> PID-1.node-3
Cost-3	: PID-1.node-2 -> PID-1.node-3
Cost-4	: PID-1.node-1 -> PID-2.node-1
Cost-5	: PID-1.node-1 -> PID-2.node-2
Cost-6	: PID-1.node-1 -> PID-2.node-3
Cost-7	: PID-1.node-1 -> PID-2.node-4
Cost-8	: PID-1.node-2 -> PID-2.node-1
Cost-9	: PID-1.node-2 -> PID-2.node-2
Cost-10	: PID-1.node-2 -> PID-2.node-3
Cost-11	: PID-1.node-2 -> PID-2.node-4
Cost-12	: PID-1.node-3 -> PID-2.node-1
Cost-13	: PID-1.node-3 -> PID-2.node-2
Cost-14	: PID-1.node-3 -> PID-2.node-3
Cost-15	: PID-1.node-3 -> PID-2.node-4

Figure 11: Possible Cost Values Between End-Points

As an another and less accurate approach, we can randomly choose only one end user from both of the PID clusters and by calculating the cost values between end-users, we can use these cost values as if these are the cost values between PIDs. In this approach as we can guess, since we make an assumption, the accuracy is not high, but on the other hand we gain from the processing power by just calculating 2 cost values. Also in this approach, if the size of the clusters made high, then the non-accuracy on cost values is going to be more error-prone.

In this study, since we do want to simulate an error-free cost calculation and the size of the simulated network is not to high, we set the size of the PID clusters as 1. Therefore, every peer actually corresponds to a PID cluster in the topology and the actual cost values can be canculated with that way. Also next section below, we give the detailed information about our novel cost calculation function as well. In addition, iin Chapter V, we run our simulations with different PID sizes in order to see and discuss the effect of Network Mapping for our netwrok topology.

3.2 Implemented Cost Calculation Methods in Cost Map Service

Under this title, we will propose the evaluated and novel cost calculation method which is going to be realized and used in our simulations. Moreover, in order to make a comparison in our simulations, an additional cost calculation algorithm is also going to be explained which is proposed in literature before.

3.2.1 A Novel Minimum Delay Based (MDB) Cost Calculation Method

It is important to note that, delay plays a significant role on Internet, especially for the delay critic applications(*e.g.* VoD applications, real-time video steaming applications). Therefore, in this section we propose a minimum delay based (MDB) cost calculation method. In packet-switched networks, the delay that a packet experiences on a simple link and a router is modeled with the following equation [27].

$$d_{nodal} = d_{proc} + d_{queue} + d_{prop} + d_{trans} \tag{2}$$

where d_{proc}, d_{queue}, d_{prop} and d_{trans} denote the processing, queuing, transmission and propagation delays, respectively. In practice, d_{proc} is often negligible, since the network devices(*e.g.* peers, routers) have reached considerably high processing speeds. Here, for simplicity in our simulations, we assume that the lengths of the links that connect all of the devices in the network are distributed uniformly. In other words $d_{prop_i} = d_{prop_j}$ for every i, j, where $i \neq j$. Also, in our simulations we guarantee that the traffic intensity [27] of whole network is not bigger than 1. Therefore, under all of these assumptions we can rewrite the equation as;

$$d_{nodal} \cong d_{prop} + d_{trans} \tag{3}$$

In this equation d_{prop} and d_{trans} are equal to

$$d_{prop} = \frac{l}{v} \quad , \quad d_{trans} = \frac{s}{b} \tag{4}$$

where l stands for the length, v is the propagation speed of the link which is close to the speed of light, s is the size of a packet on link and the b is the bandwidth of the link.

Here, Eq.(3) represent the delay on only a single link. Therefore, the total delay between PID_i and PID_j can be calculated as;

$$d_{total_{i\to j}} \cong \sum_{\eta=i\to j} d_{nodal_\eta} = \sum_{\eta=i\to j} d_{prop_\eta} + d_{trans_\eta} \tag{5}$$

$$\cong \sum_{\eta=i\to j} \frac{l_\eta}{v_\eta} + \frac{s_\eta}{c_\eta} \tag{6}$$

In this equation, c_η represents the maximum bandwidth value of the corresponding link which means while it is fully empty. However, it is required to consider with the available bandwidth value of ant link at any time t. Therefore, the equation should be revised as in Eq.(7).

$$d_{total_{i\to j}} \cong \sum_{\eta=i\to j} \frac{l_\eta}{v_\eta} + \frac{s_\eta}{c_\eta(101 - \%\mu_{\eta(i\to j)})} \tag{7}$$

where, $\%\mu_{\eta(i\to j)}$ denotes the percentage utilization on any link from PID_i to PID_j. The reason why we put 101, rather than 100 is to eliminate the zero denominator situation at the denominator of the equation while at least a link is being fully utilized.

3.2.2 Distance Based (DB) Cost Calculation Method

As an alternative to proposed cost calculation method above, an existing peer scoring and cost assigning algorithm is also examined. In this algorithm ALTO Server does nothing but calculates the physical hop number between the requester peer and the candicate peers until reaching it and then assigns an ordered costs according to the method explained in [28].

CHAPTER IV

P2P FILE SHARING APPLICATION WITH ALTO SERVICE

In this chapter, we add and analyze ALTO Service in a BitTorrent-Like P2P file sharing application. The reason why we implement a BitTorrent-Like protocol is, it is the leading P2P file sharing application, and it generates the 50% of in whole Internet traffic on the average [29][30]. Therefore, improving the performance of this protocol with ALTO Service, also affects the operational costs of the ISPs in terms of bandwidth demand in a good way as well.

4.1 BitTorrent Protocol

BitTorrent [31] is a tracker-based P2P file sharing tool which is used most of the internet users today. In BitTorrent, users not only download content from the server but also serve it to the other peers. Thus the serving capacity of the system grows with the number of nodes, making the system potentially self-scaling. The idea behind BitTorrent is to divide entire content into small pieces called *chunks* (typically 256 Kbyte size) and share and upload them the other users. There are two different user type defined in BitTorrent which are seed and peer. Seed is used for the nodes who has the entire content and only upload it to the requested users. Peer on the other hand is called for the user who has none or non-entire of the content. As can be understood from the definitions, seeds only upload the corresponding content like servers, and peers upload the corresponding content to requester peers and download the non-existing chunks from the other peers, simultaneously. The general working mechanism of BitTorrents is as follows;

1) First, a user in BitTorrent overlay, creates an encrypted metadata file called *torrent file* of the content it wants to share, with other users in the overlay and then publish it via Web. It contains no information about the content of the file. The only data that the torrent holds is, the information about the location of different pieces of the target file. It also contains the URLs of many central, determined control servers' (called *trackers*) informations and integrity metadata about all the pieces.

2) Peers who want to download the content first obtain the corresponding *torrent* file from Internet and run this file via the application.

3) Since the *torrent* file includes information (URL Address) about the *tracker* peer connects the this server and request for a peer list in order to start downloading.

4) Once receiving this request *tracker* sends a randomly generated peer list that have the *chunks* corresponding content.

5) Having received the peer list response peer starts downloading the *chunks* from the other peers. Figure 12 summarizes the overall chronological progress below as well.

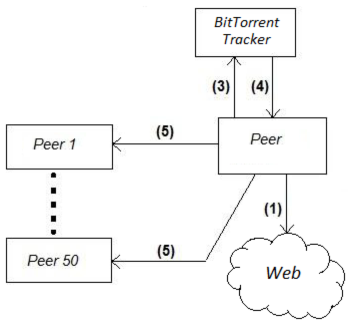


Figure 12: BitTorrent Working Mechanism

4.2 Proposed Network Model

In this study, we simulated two differenet BitTorrent-Like file sharing scenarios with and without ALTO Service, those include 780 peers, one *tracker* and three ALTO Servers each of them are placed to each ISP. In the first scenario, when a peer wants to join the overlay, it first sends a request to the *tracker*. While the peer is registering to the overlay, it also sends an information about which *chunks* of the file it has or has not as well. Then, when the *tracker* receives the request from the corresponding peer, it registers the peer and sends a randomly created peer list that includes 20 peers' IP addresses. The peer that receives the peer list start to establish TCP connections with the peer that are in peer list. Besides, every peer in the overlay sends an updated information about which chunks it has to the tracker with a 10 seconds period of time. In the second scenario, three ALTO Servers are made active in each ISP to provide Network Map and Cost Map informations. Therefore, before creating and sending a peer list, *tracker* obtains the Cost Map informations from the corresponding ALTO Server, as shown in Figure 9. Once getting the related cost values between *PIDs* which are calculated by using the MDB cost calculation method detailed in section 2.4.1, *tracker* creates a ranked peer list and sends it to the requester peer for better performance. As a detail, in both of the scenarios we assume that, initially all of the peers have the *.torrent* file which is needed to connect to the *tracker*. Also upload and download bandwidth rates are distributed to the peers according to Table I, below [32]. It is important to note that every ISP has a direct link to other two ISPs as shown in Figure 13. Besides, the internal physical topology of an ISP is shown in Figure 14, as well. In that topology model we see that, for simplicity the connections between routers are made according to tree based model all in three ISPs. In other words, in overall topology, there is only one path from any peer to any peer. Therefore, this situation lets us avoiding multiple routing path issues for this study. As a note, we are going to use the same topology with a minor change in the next study, again. The size of the content is set 16 Mbytes and the simulation is run for 6 minutes.

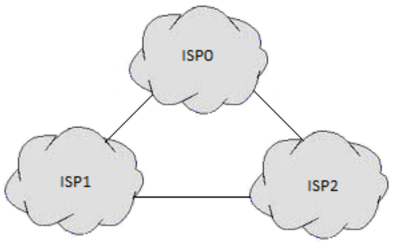


Figure 13: Proposed Network Model

Table 1: Bandwidth Distributions of Peers

Percentage(%)	Upload Bandwidth(Mbps)	Download Bandwidth(Mbps)
56	0.5	0.25
21	3	0.4
9	1.5	0.9
3	20	2
11	20	5

4.3 Simulation Results

During simulations in both of the scenarios, initially 30 peers in the overlay are assigned as seed (who have entire content) and they are equally distributed to the three ISPs. In this study we focus on two results, that are Inter-ISP traffic rate and content downloading completion time of the peers. In Figure 14, 15 and 16 we see the average traffic rates flowing between ISP1-ISP3, ISP1-ISP2 and ISP2-ISP3 respectively. In all of the graphics we see that, the average cross domain traffic rates are reduced approximately half of them by using ALTO service. As an interesting note, in Figure 15 we see a different behaviour than Figure 16 and 17. As we see threre is a dramatical decrease at time about 4 min. The reason why such a decrease has occured in ALTO free scenario is, or in other words the reason why the Inter-ISP traffic is so high in the beginning of this scenario is, random peer selection process causes inefficient peer lists in terms of cross-domain traffic and also since the bandwidth distribution is not the same of the peers. However, after a considerable time Inter-ISP traffic has reached to it stable value as like in Figure 16 and 17. Since

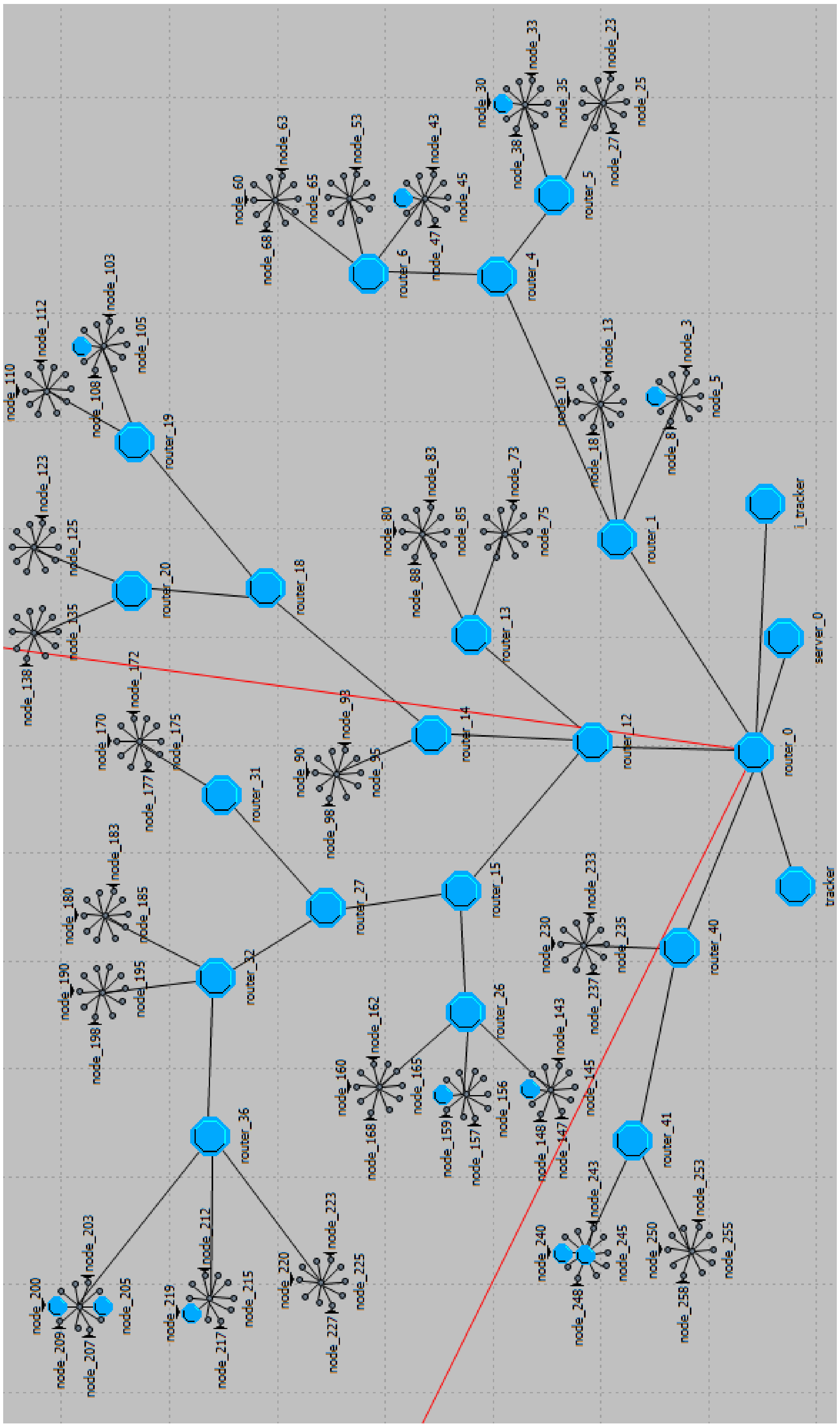


Figure 14: Intenal Pyhsical Topologies of ISPs

all the topology is symetric we see similar results in all figures.

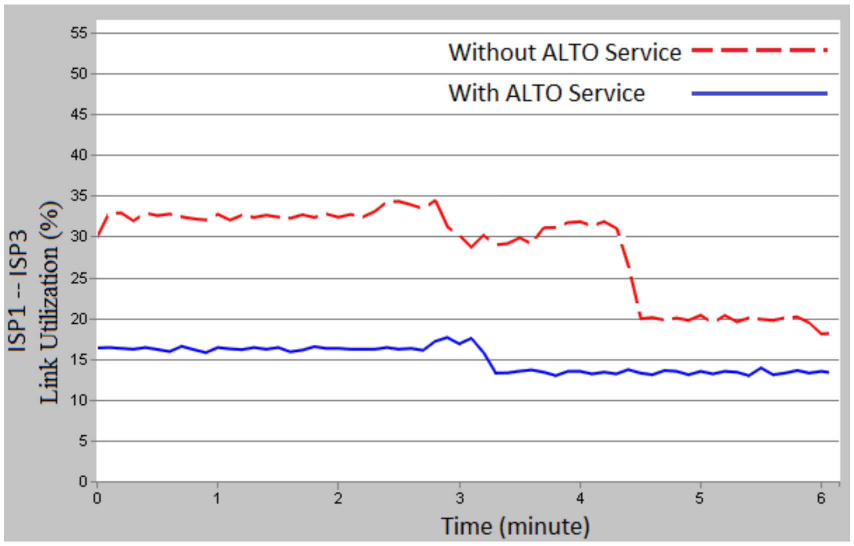


Figure 15: Average Traffic Rates Between ISP1 and ISP3

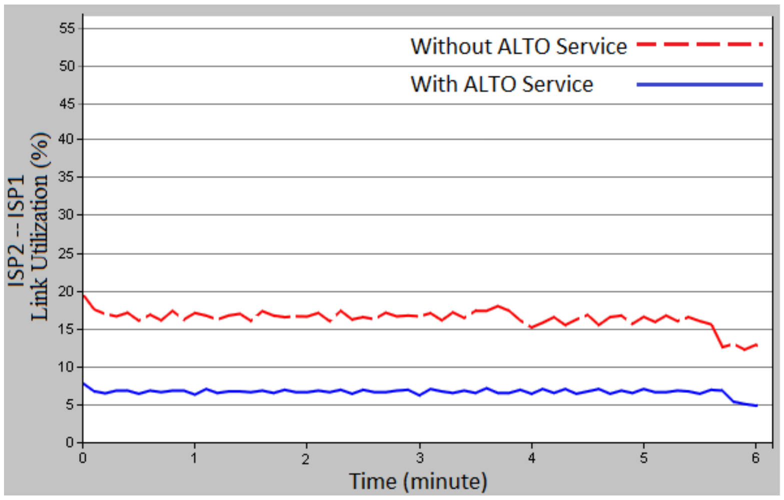


Figure 16: Average Traffic Rate Between ISP1 and ISP2

Similarly, if we look at the content downloading completion time in Table 2, we see a small improvement, as well, since we focused on the minimum delay while creating peer list for any peer. Here, since there is o congestion between any link, except (peers' link) we don't see such a big gap in terms of content downloading comletion time.

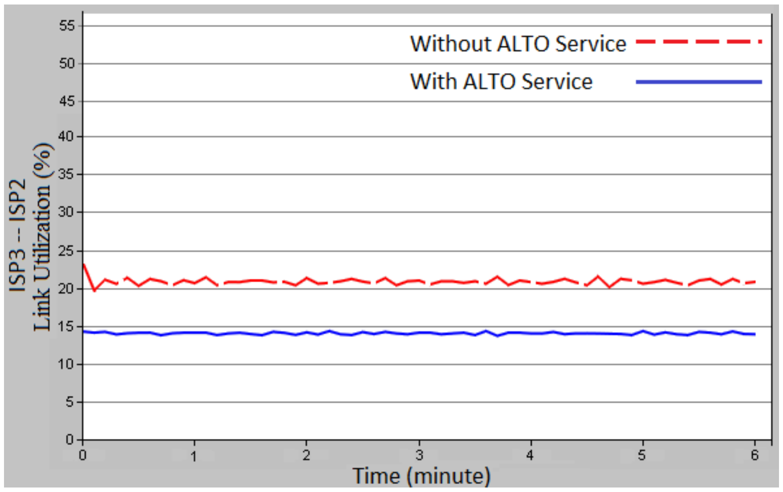


Figure 17: Average Traffic Rates Between ISP2 and ISP3

Table 2: Average Download Completion Time of Peers(10 min of simulation)

With ALTO Service (sec.)	Without ALTO Service (sec.)
375	383

CHAPTER V

P2P REAL-TIME SCALABLE VIDEO STREAMING WITH ALTO SERVER

5.1 Introduction

Recently, there has been a noticable interest on deploying P2P technology for real-time video streaming on Internet [33][34][35]. The main reason of this interest is, P2P systems do not require any additional entity to deploy on the currently existing Internet. Secondly, in such a system peers not only download the content, but also upload it to another peers exist in the overlay. Therefore, such a system defines a self-scalable architecture by itself. However these type of overlaying comes with another significant and challenging problems. Rather than sending a non-real time content in P2P overlay (*e.g.* BitTorret), P2P real-time video streaming applications include much more control messages and scheduling of the peers is very complex [36][37]. Specifically, video broadcasting imposes stringent real-time performance requirements in terms of bandwidth and latency [38]. These distinguishing and stringent requirements of real-time video streaming necesiate fundementally different design desicions and approaches.

The most fundamental question need to be answered in P2P real-time video streaming is how to construct the overall overlay. There are two types of existing approaches proposed in liteature which are called *tree based*, and *mesh-based* approaches. In the tree-based approach, there is an overlaying construction proccess which organizes the participating peers into multiple trees [39][40][41]. The number of trees that any peer is going to connect is determined by its upload and download badwidth attributes. Therefore, in order to minimize the effect of peer churn and to effectively utilize the available resources in the system, peers are distribued to the multiple trees. In these multiple trees, each peer is placed as an *internal* peer or *external* (or leaf) peer [42]. Also, the distribution of the content is realized with a push mechanism where each peer sends the pieces of the content it has, to its *child* peers while it receives the required pieces from its *parents*. As we can understand from the description, the main component of the tree-based approach is the tree construction process.

On the other hand, in the mesh-based P2P streaming, peer in the overlay form a randomly connected structure or *mesh*. Similar to tree-based approach, each peer maintains a certain number of peers (either *parent* or *child*) according to its both upload and download bandwidth values. In this architecture when a peer join the overlay, first it connects to a *bootstraping* node whose IP is static and known initially and request a peer list to connect [42]. Once receiving the request bootstraping node sends a randomly created peer list to the corresponding peer. Upon arrival of peer list, the peer starts to connect a subset of the list in order to start streaming. Unlike the tree-based approach these is not a structured tree topology in this system. Therefore, any peer can send and receive content from the peers it connects at the same time. For that reason, this approach is called a pull-based approach as well, since the content is requested from antoher peers before receiving it. Figure 18 shows an example of overlays, created with both tree-based and mesh-based approaches.

In this study, we evaluate a P2P real-time scalable video streaming protocol by using ALTO service. Our scalable video streaming protocol based on a mesh-based approach. To evaluate the performance results of the overall system with different cost calculation algorithms we describe the in chapter 3.2.1 and 3.2.2, we carry out the experiments over a simulated network, in OPNET [43] environment, where all the peers run our proposed real-time scalable video streaming application. Simulations are run over a determined network topology, with 3 ISPs and 780 peers in

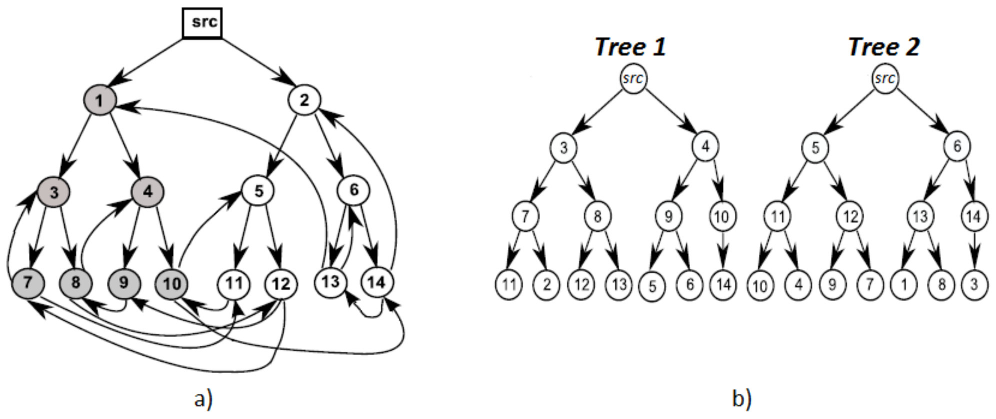


Figure 18: Overlay Structures a) Mesh-Based Approach b) Tree-Based Approach

total. Also, the backbone links are sufficiently provisioned so that congestion may only occur on the links connecting the peers to the network. Therefore, in the backbone video packets experience no (or very short) queuing delays since because no congestion occurs. This situation also explain why we ignore the queuing delay while calculating the end-to-end total delay in chapter 3.2.1. Additionaly, there are three streaming servers each of them is placed to each ISP. As can be remembered, our streaming system depends on a Tracker-Based model. Therefore, the Application Tracker(AppTracker) is also made available in one of the ISP which is ISP0. Finally, in each ISP an ALTO Server is enabled in order to obtain the network map of the corresponding ISP and create the cost values between PIDs. In our simulations we use reliable communication links that have zero error probability in order to avoid to quality degradations due to the packet losses. Also the control and transmission protocol is implemented over UDP/IP protocol stack, and we ignore any network address translator (NAT) or firewall issues that may limit the connectivity of the peers.

In our simulations peers have variable and fixed downlink an uplink bandwidth, which they have measured and know accurately. The bandwidth distribution is derived by using the latest report of Akamai, which is the leading content delivery network company [44]. Table 3 shows the download and upload bandwidth values and the corresponding percentage of them.

Table 3: Bandwidth Distribution of Peers

Download(Mbps)	Upload(Mbps)	Percentage(%)
0.5	0.25	5
1	0.5	10
8	1	20
20	3	50
20	5	15

In our model, peers are distributed into the ISPs with different hop numbers. At that point, it is important to remind that, we use the same network topology that is used in the previous study. However, at that time we also push the simulator to distribute the bandwidth values of peers, in each ISP, exactly the same, unlike the previous work. In each ISP, there are 260 peers distributed with different hop numbers, and in total there are 780 peers in three ISPs, as indicated before. The distance of the peers to the central router of the corresponding ISP, in terms of hop number is mentioned in Table 4.

Table 4: Peer Numbers with Different Hop Distances

Hop Number	Peer Number
3	90
4	150
5	240
6	210
7	90

In the experiments, all of the peers are set active during the simulations. It means that, when a peer join the system it doesn't leave from the overlay until the end of the simulation. In order to get the performance results of different scenarios, we simulate three different experiments on the same network topology. In the first two scenarios, we simulated an ALTO service enable P2P real-time scalable video streaming application, by using the cost calculation methods detailed in chapter 3.2.1

and 3.2.2, respectively. In our last experiment, we disable ALTO Server information for the application in order to show the benefits of ALTO information. Since the ALTO Service is not available, the application generates random peer lists and sends them to the requester peers. The dynamic behavior of the scenarios in all three cases is as follows; For the scenarios in which ALTO Service enabled, after completing the phases for JOIN REQUEST and JOIN REPLY, before creating and sending the peer list to the corresponding peer, AppTracker receives the cost informations between all of the *PIDs* and creates the most convenient peer list. On the other hand, in other scenario where ALTO Service is disabled, AppTracker sends a randomly created peer list to all peers. The chronological messaging processes are shown in Figure 19, below. We run the experiments and show the results for a 52 sec. video with the 854x480

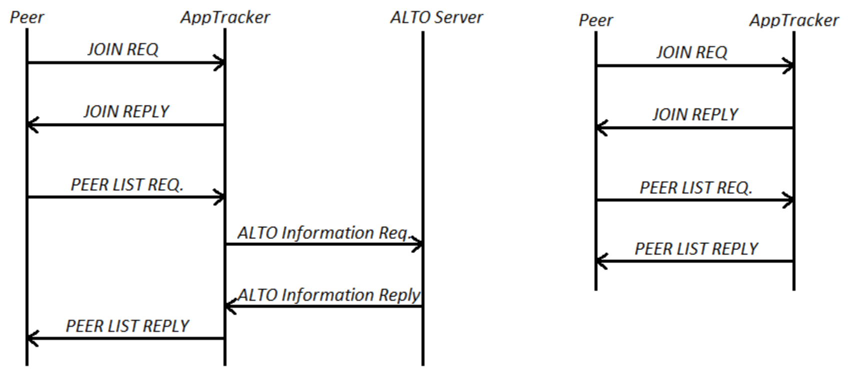


Figure 19: Flowchart of Messaging

(or 480p) format . It is encoded at 30 frames/s according to H.264/SVC by using the reference software JSVM [45] and bit rates of scalable layers are shown in Table 5. The video quality is recorded for all of the peers and it is measured in terms of average *peak signal-to-noise ratio (PSNR).*

5.2 Simulation Results

In this section, we present the simulation results obtained in OPNET Network Simulator. In our system we particularly focus on the average video download and upload

Table 5: Bit rates of Scalable Layers

Layer	Bit rate (kbits/s)
0	199.02
$0+1$	349.85
$0+1+2$	385.10
0+1+2+3	415.83

rates, video pause duration of the peers, streamed video quality of the peers in terms of *PSNR* and inter-ISP traffic rates, respectively.

5.2.1 Bandwidth Utilization of Peers

In P2P Video Streaming systems, the most important statistics are the video download and upload rates of the peers, since they give a a lot of information about QoE experienced by the peers as well as the continuity index of the stream. Therefore, we first discuss the average download and average upload speeds of all peers in Figure 20 and Figure 21, respectively. As can be seen in Figure 20, the average video download rates of the peers are similar to each other, especially as time passes. Besides, at the beginning of the streaming, ALTO-free and MDB peer ranking algorithms employ a slightly better performance than the DB peer ranking algorithm which are about 700 Kbps. However, since this graphics mirrors an average rate of the results, it is better to analyze the corresponding result with CDF of the starting times of the peers.

In Figure 21 similar to Figure 20, average video upload rates are given. According to this graphics, using any cost calculation method with ALTO service reduces the average upload rates 10% approximately. This situation occurs due to ALTO service's strict policy, especially in MDB peer ranking algorithm. Because in MDB peer ranking algorithm, we push ALTO server to rank the candicate peers (or PIDs) by not only looking at the end-to-end delay between end-points, but also looking the the location of the peers. In other words, if a peer (or PID) belongs to other ISP, ALTO Server assigns a considerably high cost value which in result, put that end-point to the

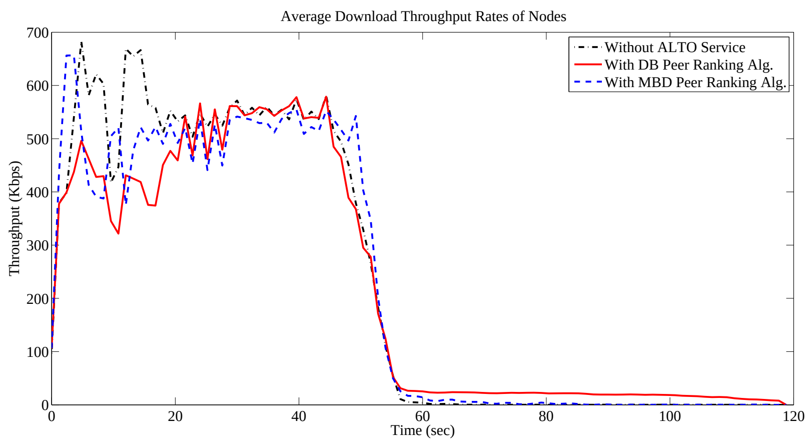


Figure 20: Average Video Download Rates of Peers

bottom of the peer list even that this end-point is has a better end-to-end delay value. Similarly, in DB peer ranking algorithm we just consider with the end-to-end distance between end-points and we ignore any another metric that has a potential to affect the performance of the streaming.

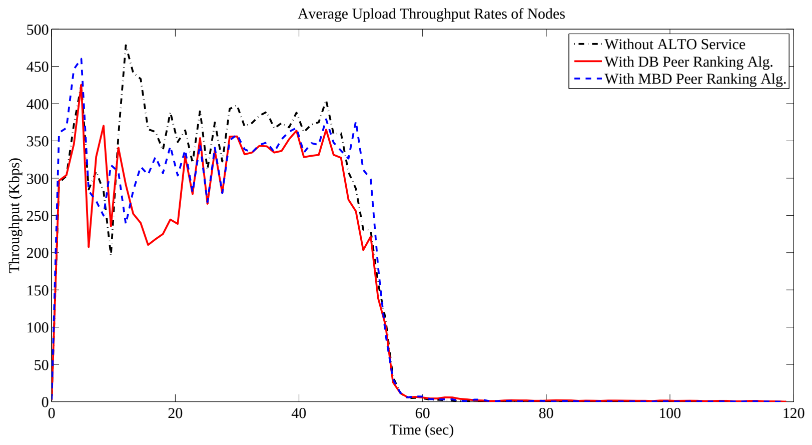


Figure 21: Average Video Upload Rates of Peers

5.2.2 Streaming Start-Up Time Distribution

In the following figure, Figure 22, CDF of the streaming start-up times is represented for all of the scenarios. While we are discussing about the average video download rates of the peers, we have mentioned that, it is better to analyze the result with the starting times distribution of the peers. In this figure, we see the reason of that conclusion. Because, although the average video download rate of the peers is maximized by not using an ALTO service, we see that enabling ALTO service improves the performance of starting time distribution. This situation is because, enabling ALTO service may reduce the average download rate, but all of the peers have the similar download rates. Although in ALTO-free scenario, the average video download rate is higher than other two scenario, due to a high variance between peers in terms of downloading, peers are starting to streaming a little later. Also, with ALTO service wee see that approximately 90% of the peers start streaming less than 8 seconds, which is quite acceptible and much more welcome than the other protocols exist in literature.

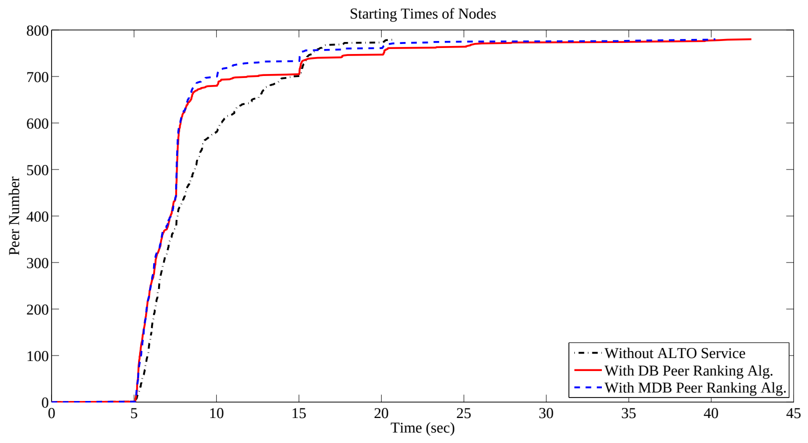


Figure 22: CDF of the Stream Starting Times of the Peers

5.2.3 Pause Duration Distribution

In this section we present the CDF of the pause duration of the peers which is shown in Figure 23. According to this graphics we see that, all of the scenarios show a similar behaviour, especially ALTO enabled streaming scenarios. In ALTO free scenario, we can claim a slightly better performance in terms of the number of the peers that experience zero pause. Here, it is important to mention that, more start-up delay sometimes can provide less pause durations, since there is more chance (time) for peers to fullfil their streaming buffers. Therefore, while some peers experience a few seconds of pausing, in ALTO-free scenario, these peers may not experience any pause since they have received frames of the stream, that cause pausing, before starting streaming. Also, at time t ¿ 20 sec. we see a constant gap (corresponds to approximately 50 peers) between ALTO-free and ALTO enabled scenarios. The reason of that difference is because of the peers that could not receive a base layer packet of one frame, which is not no more exist (available) at any peer. Since these peers will not be able to request and receive that packet from any of the peers (including servers), these peers experience an infinite pause. Therefore, they will not be able to finish streaming at any time.

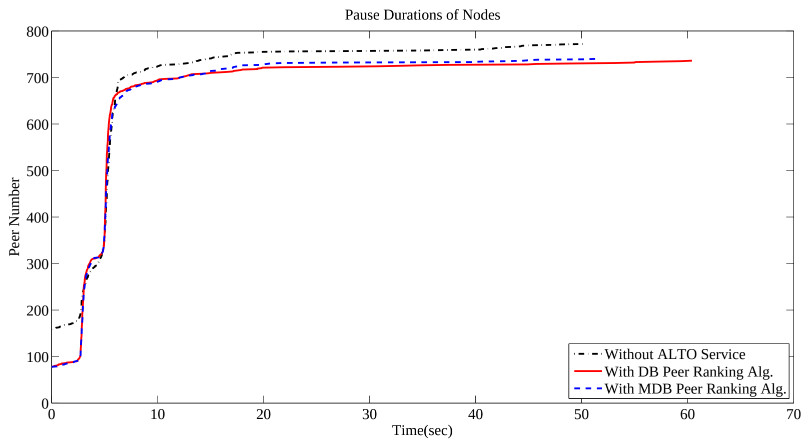


Figure 23: CDF of the Pause Durations of the Peers

5.2.4 Streaming Completion Time Distribution

Next, we examine the average streaming completion time of the peers in overall overlay. According to Figure 24, until t = 68 sec. we see a slightly better performance in ALTO-assisted scenarios, while after t= 68 sec. the number of the peers that has finished streaming is getting more in ALTO-free simulation since because the reason that we have just explained (infinite pause situation). Also, in all of the scenarios most of the peers finish streaming before 70 sec. which is quite acceptable and ALTO policy is not affecting the performance of the application in a bad way in terms of video completion times.

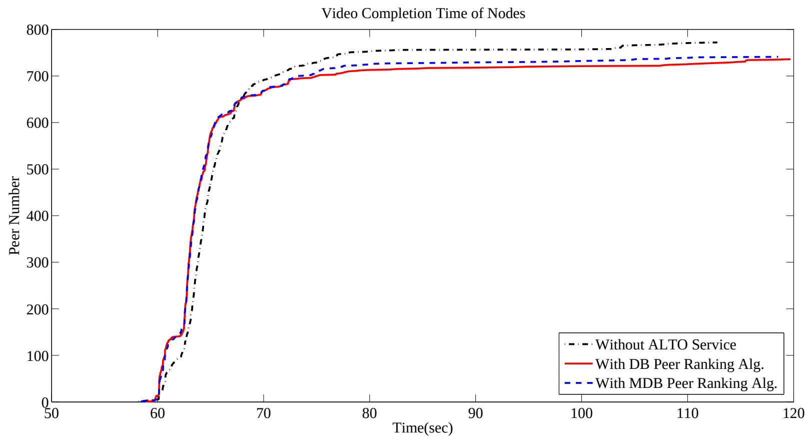


Figure 24: CDF of Video Completion Times of the Peers

5.2.5 Received Video Quality

For QoE purposes, we calculate the Peak-Signal-to-Noise-Ratio (PSNR) values, in order to compare the different scenarios. In Table 6 maximum, minimum and average PSNR values of the peers that finished streaming are given respectively.

Table 6: PSNR Value Statistics

	Witout ALTO Service (dB)	With DB Peer Ranking Alg. (dB)	With MDB Peer Ranking Alg. (dB)
Maximum	42.1	42.1	42.1
Minimum	36.49	32.91	32.89
Average	40.23	39.45	39.57
Variance	1.79	2.67	2.44

Since the table is filled with the statistics of the peers that finished streaming, we can on the other hand guarantee that, minimum PSNR value can be the base layer PSNR value since the streaming cannot be finished without receiving all of the base layers of the video, which is about 33dB which can be seen in both of the ALTO-assisted scenarios. However, in ALTO-free scenario, minimum PSNR value is about 36dB which is quite better than ALTO assisted scenarios that means a strict ALTO service has a potential to reduce the video PSNR values of some peers. On the other hand, maximum PSNR value of the video is exactly 42.1dB which can be obtained by receiving all of the layers of the whole video. With that in mind we also see that, in all of the scenarios at least one peer takes all of the layers. In that table, the average PSNR value can give us a good idea for comparison of the scenarios. According to table, the average PSNR values are quite similar to each other which means that, ALTO service is also can be regarded as acceptable for the applications in terms of video quality as well. As an additional information, these PSNR values are calculated by ignoring pause durations of the peers.

5.2.6 Inter-ISP Traffic

Last of all, in this work we analyze the average traffic rates between ISP, those show the effect of ALTO server and the corresponding ALTO policies mostly. In Figure 25, all of the inter-ISP traffic rates are given below. According to this result, we can assert that, ALTO protocol has a big potential for reducing the unnecessary data flow between ISPs, since it reduces the traffic from 50% to 5% and 3% in DB and MDB peer ranking algorithms, respectively. By reducing these corresponding traffic rates, ISPs can also have a change to strengthen their economics, since they will not be charged by other ISPs any more. On the other hand, this result also endorses the main motivation behind ALTO protocol, which is to reduce the unneccessary traffic caused by distributed applications by not affecting the performance of them. If we look at the graphics closely, we also see a peak about 5 seconds. This peak is because of the buffering of all of the peers. Since almost all of the peers has not started streaming before 5 sec., peers maximize their download throughput as much as they can, in order to fill their empty buffer. Once starting the streaming, peers can only download a content at a maximum rate of 400Kbps which is the data rate of the video. Therefore the average inter-ISP traffic rate is reducing just after 5 sec.

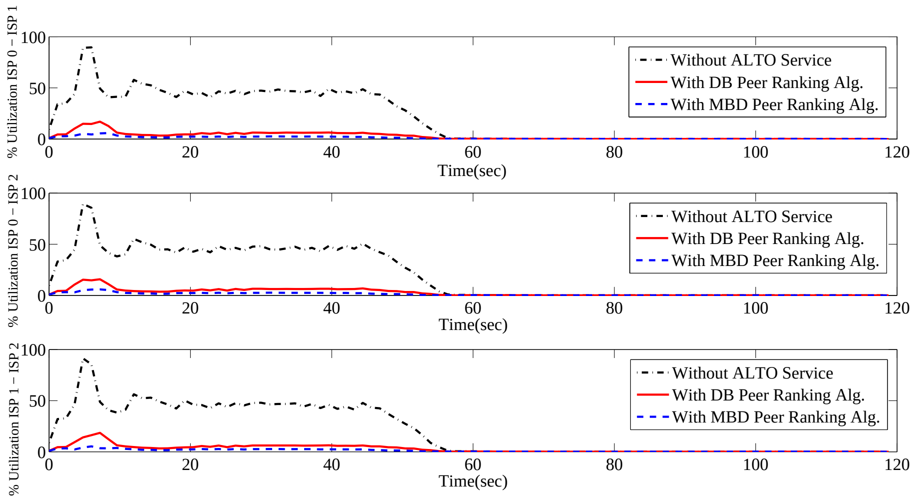


Figure 25: Average Inter-ISP Traffic Rates

In conclusion, in this work we figure out that, pushing ALTO service to the applications may have a possibility to reduce the performance of the applications even if just a bit while it dramatically reduces the cross domain traffic, enabling a softer ALTO policy can have a possibility to improve the performance of both the applications and service providers.

5.3 Effect of Network Mapping with Different PID Sizes

As can be remembered, we run our simulations, with PID size 1 in the previous section. However, decreasing PID size causes a high cost computation load at ALTO Server as explained in Chapter III. In this section, we run our simulations with different PID groupings based on the location of the peers. In other words, Network Map Service is going to assign the groups of peers to the same PID clusters, by considering with their geographical locations. During simulations, we set our PID size to 2, 5, 10, 20, 52, 130 and 260, the submultiples of 260 which is the number of peers that each ISP includes. We propose our results for both of the cost calculation methods which are DB and MDB cost calculation algorithms. First, we start with average download and upload bandwidth utilization of the peers for both of the cost calculation methods, that can be seen in Figure 26 and Figure 27, Figure 28 and Figure 29, respectively.

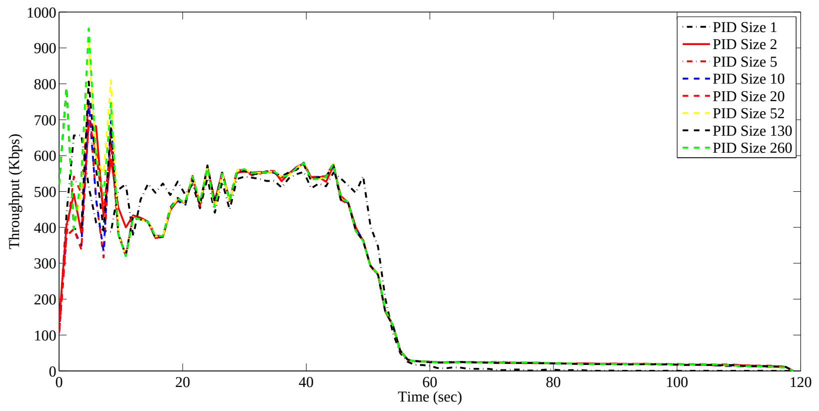


Figure 26: Average Download Utilization of Peers with Different PID Sizes by Using MDB Peer Ranking Alg.

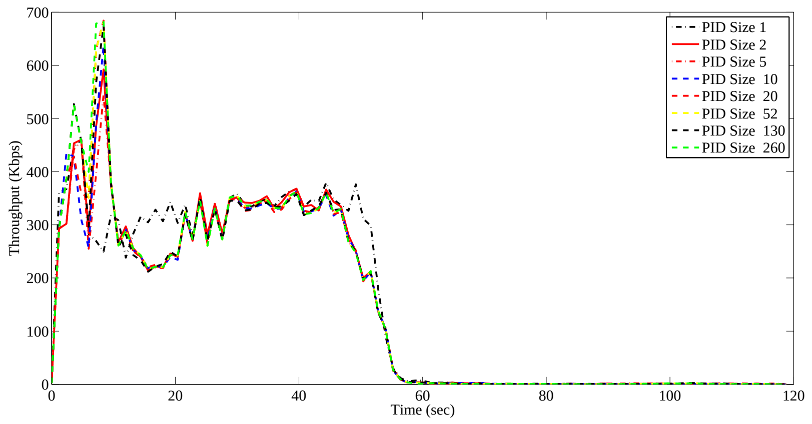


Figure 27: Average Upload Utilization of Peers with Different PID Sizes by Using MDB Peer Ranking Alg.

According to these results, we see that increasing PID size (therefore decreasing the cost calculation load between PIDs) is not affecting the average download and upload rates of the peers noticably. The first reason of that results is due to the

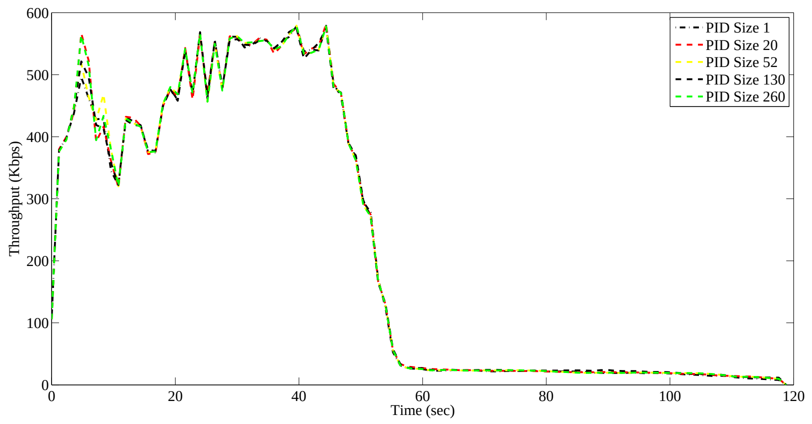


Figure 28: Average Download Utilization of Peers with Different PID Sizes by Using DB Peer Ranking Alg.

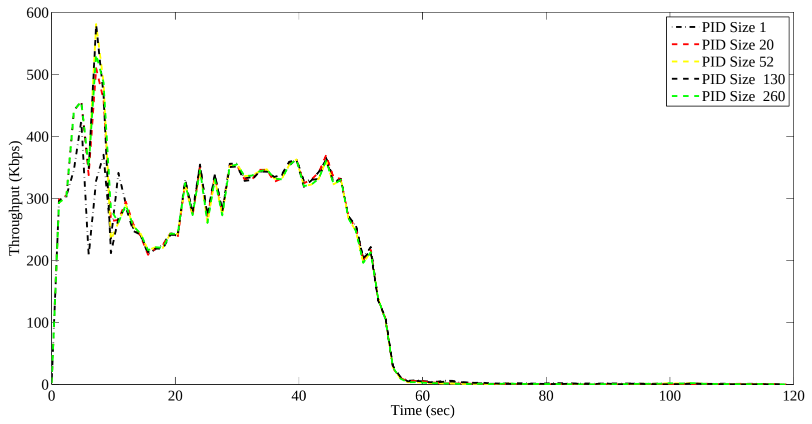


Figure 29: Average Upload Utilization of Peers with Different PID Sizes by Using DB Peer Ranking Alg.

effect of the ALTO service to the application. As we have described before, ALTO service was affecting the performance of the application in a slightly bad way when PID size is equal to 1, and while we are increasing the size of the PID clusters, indeed we are getting closer to a random cost calculation, and at the same time a random peer ranking behaviour. Therefore, the performance results of the application is not changing to much in terms of bandwidth utilizations.

Similar to bandwidth utilizations, we again see the very close (almost the same) results, when we look at the CDF of the starting times, pause durations and video completion times of the peers in the following figures, Figure 30-35. Another reason of these similar results with different PID sizes is because of the overall topology of our network. Accroding to our network topology, most of the peers are distributed at the edges of the network. Therefore, this situation causes in a way that a less-dependent case to PID sizes for ALTO service, since the hop distances of the peers are very close to each other, and the main delay is caused by the backbone links of the network as can be seen in Figure 14.

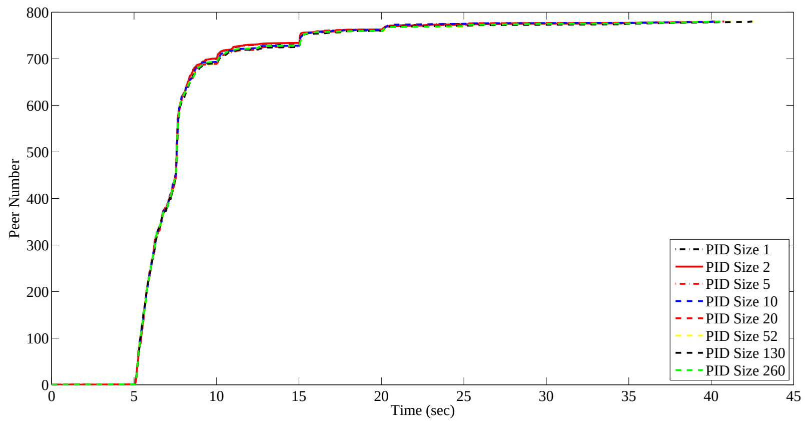


Figure 30: CDF of the Stream Starting Times of the Peers with Different PID Sizes by Using MDB Peer Ranking Alg.

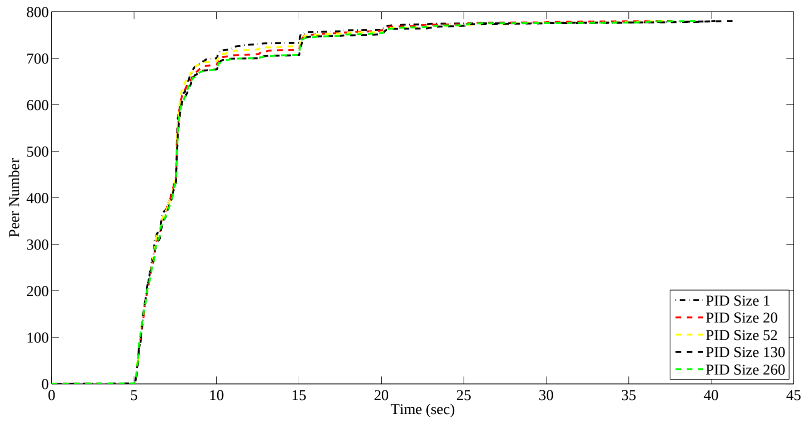


Figure 31: CDF of the Stream Starting Times of the Peers with Different PID Sizes by Using DB Peer Ranking Alg.

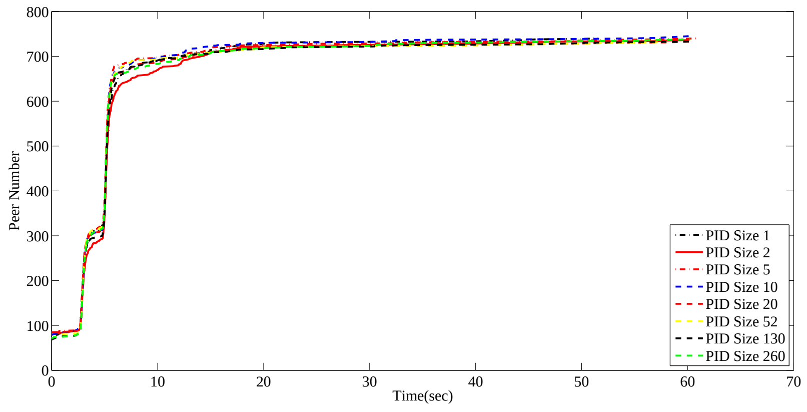


Figure 32: CDF of the Pause Durations of the Peers with Different PID Sizes by Using MDB Peer Ranking Alg.

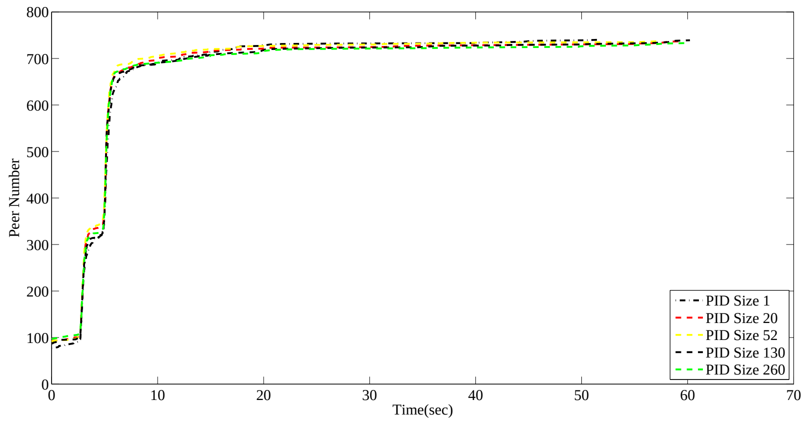


Figure 33: CDF of the Pause Durations of the Peers with Different PID Sizes by Using DB Peer Ranking Alg.

In our first results (for PID size 1 case), we had seen that the main difference at inter-ISP traffic rates. So, by increasing the PID size we could expect to getting closer to ALTO-free scenario in terms of cross-ISP traffic rates with high charging rates. However, since we assing a high default cost value for the peers that are belong to different ISPs, in any PID size case, we again put these peers at the bottom of the peer list. Therefore, increasing PID size does not affect the inter-ISP traffic rates since PID size cannot affect the default cost value, as can be seen in Figure 36 and 37, for both of the cost calculation methods, respectively.

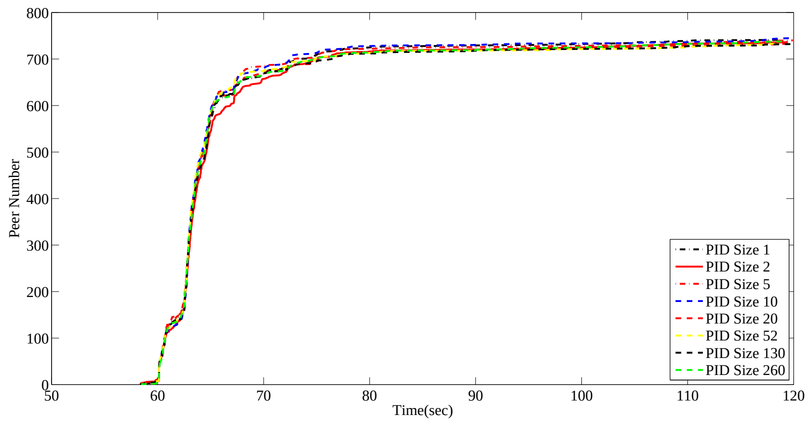


Figure 34: CDF of Video Completion Times of the Peers with Different PID Sizes by Using MDB Peer Ranking Alg.

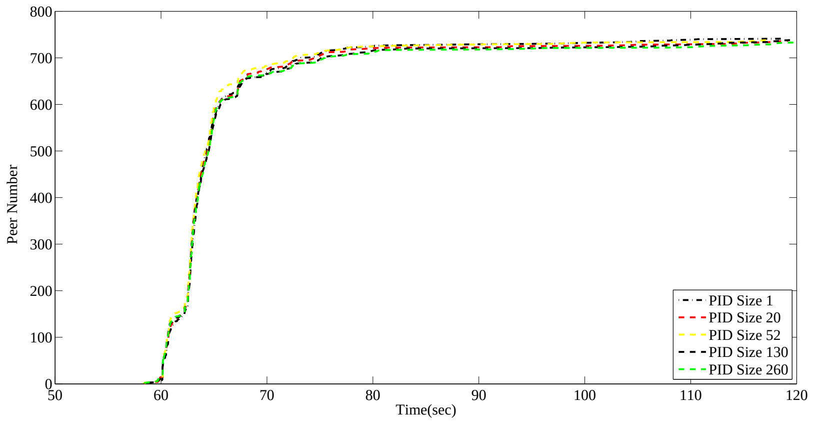


Figure 35: CDF of Video Completion Times of the Peers with Different PID Sizes by Using DB Peer Ranking Alg.

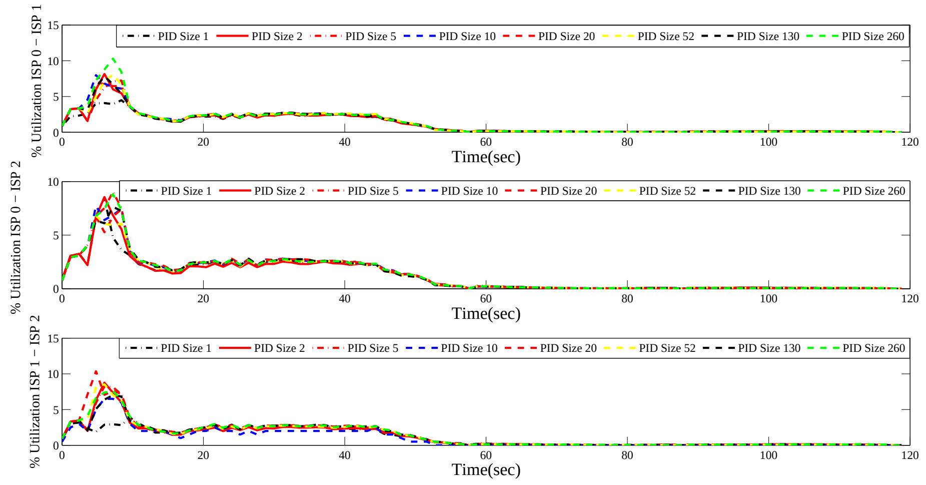


Figure 36: Average Inter-ISP Traffic Rates with Different PID Sizes by Using MDB Peer Ranking Alg.

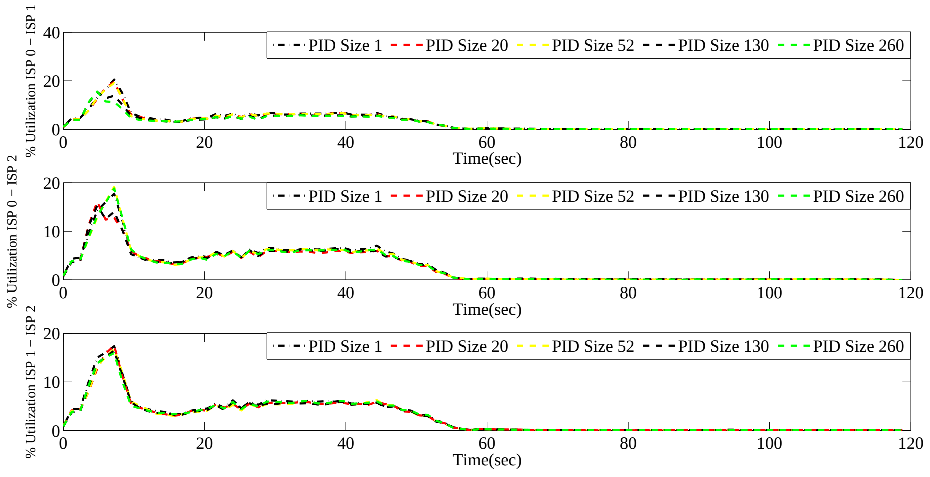


Figure 37: Average Inter-ISP Traffic Rates with Different PID Sizes by Using DB Peer Ranking Alg.

CHAPTER VI

ENABLING ALTO SERVICE ON SOFTWARE DEFINED NETWORK FOR A CDN APPLICATION

In this chapter, we implement a CDN based real-time video streaming application, run on a simple Software Defined Network (SDN) and develop an ALTO Service application on the control plane of the SDN. In our demo, we try to show the functionality of ALTO service, how easy these type of services can be enabled and its benefits for these type of applications.

6.1 SDN in a Nutshell

Network management is a challenging problem of wide impact with many enterprises suffering significant monetary losses, that can be of millions per hour, due to network issues, as downtime cost [46]. Also researchers cannot implement their new and innovative ideas at any scale on real network settings since todays current network architecture is mostly distributed, in terms of controlling and routing each packet in a network. In other words in todays Internet each device has a partial view of the network and forwards the packets by looking at these partial network topology informations. Moreover, every vendor has its own operating system and different applications run on its own device(*e.g.* switch, router). Therefore, service providers cannot obtain new enhancements and *Quality of Service* (QoS) requirements and they cannot meet their customers' requirements, since an overall optimization is not possible to reach. In Figure 38 below, we see the summary of this situation.

SDN approach is a new paradigm that enables the management of networks with low cost and complexity. It allows network operators to manage networking elements

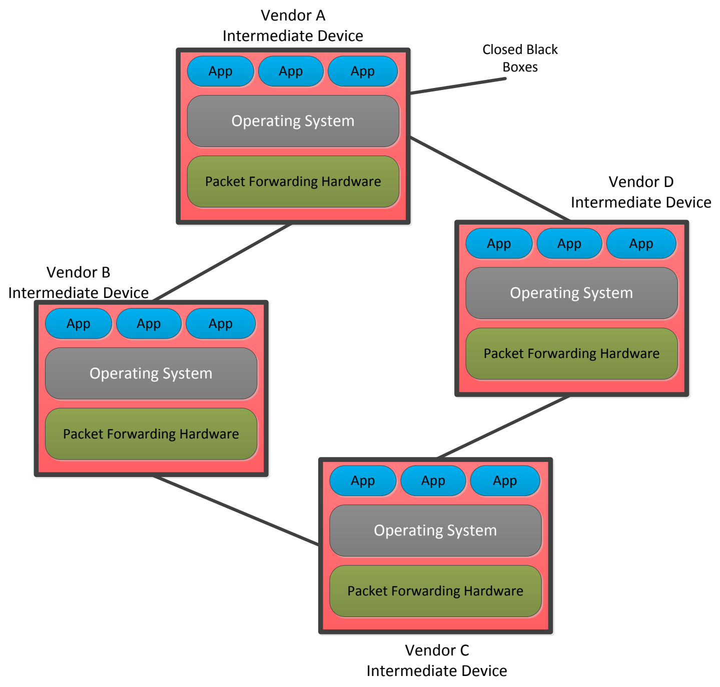


Figure 38: Today's Network's Intermediate Devices as Black Boxes

using software running on an external server. It is a way of separation of control plane and forwarding plane in the network devices (*e.g.* switches and routers) on the existing Internet. In this architecture, control plane is implemented with a open source software (*e.g.* Beacon). The logically centralized control plane is realized using a network operating system that constructs and presents a logical map of the entire network to services or control applications implemented on top of it. Therefore, with SDN, service or network providers can introduce and enable several new capabilities by using the topology information which is obtained by the OpenFlow Controller. Then, the rest is taken care of by the network operating system [47]. In Figure 39 below, we see the general architecture defined with SDN.

As we mentioned above, the most important entity comes with SDN is the central server which is called *controller*. Controller is a server which programs the intermediate devices (*e.g* switches) for the packet flows in the network. Therefore, by

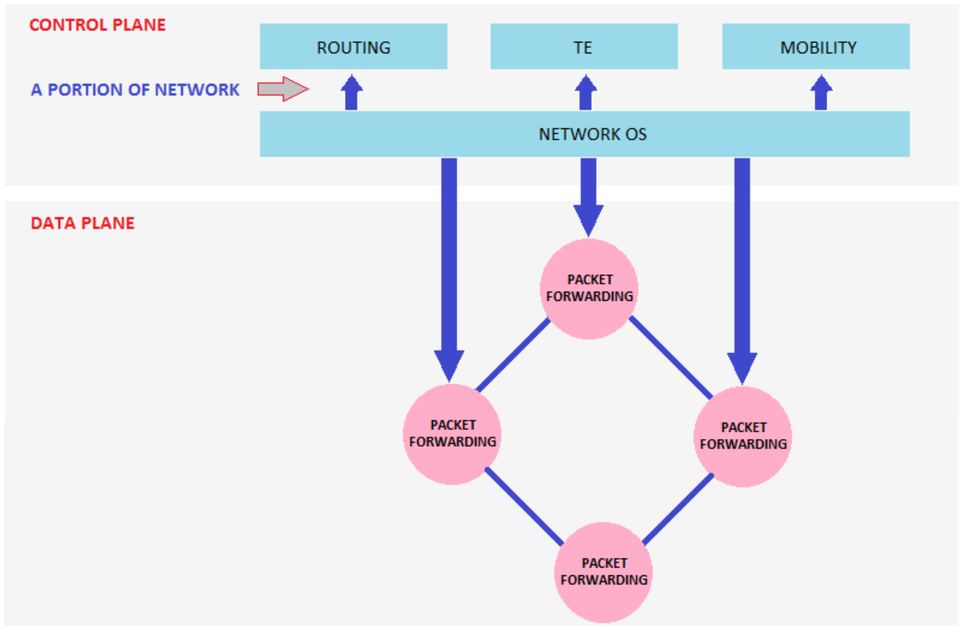


Figure 39: SDN Architecture

using these informations obtained by the controller, switches only forward the corresponding packet or flow by just looking its flow table. In order to standardize the communication between controller and switches a new protocol is defined. This protocol is called OpenFlow [48]. OpenFlow is a protocol that defines the design issues of the controller and message types used between OpenFlow switches and controller. There are several and similar open source OpenFlow controllers available on Internet [49][50][51]. At this stage let us describe how a single packet is sent on a simple SDN architecture shown in figure, below.

Let us assume that Client A wants to send a packet to the Client D which is connected to a different OpenFlow switch. At this stage it is important to note that, in the current topology flow tables of both of the OpenFlow switches are empty. In other words, OpenFlow 1 switch does not know from which port it should send the packet coming from Client A whose destination is Client D. Under these conditions when switch 1 have received the packet, it asks for the flow entry to the OpenFlow controller in order to push the packet to Client D. Once OpenFlow controller received

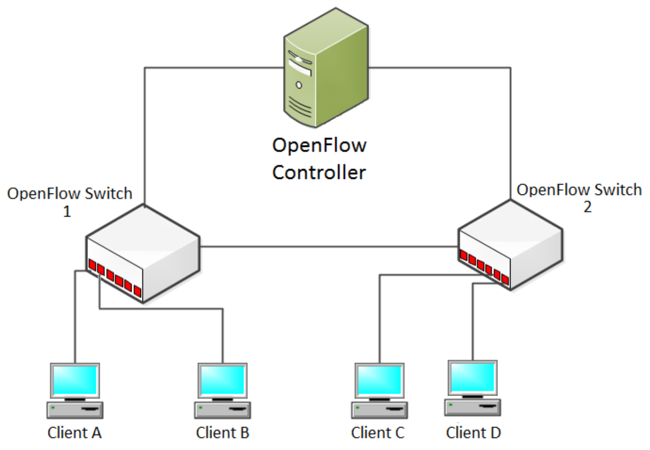


Figure 40: A Sample SDN Topology

this request, it calculates the convenient route and sends this information to the switch 1 and switch 2. Having obtained the flow entry, switch 1 sends the packet to its correct port, that has a connection to switch 2. Then, OpefFlow switch 2 sends the corresponding packet to Client D by just looking at its flow table, without no need any flow entry since switch 2 has updated its flow table before.

6.2 Real Time Video Streaming with ALTO Service on SDN

After defining what SDN and OpenFlow is, in this section we explain our CDN based real time video streaming and ALTO service embedded OpenFlow controller setup. As discussed before, in CDN applications the content is generally available more than one server or location in order to reduce the traffic and obtain scalability. Therefore, in our setup we activate two real time video streaming servers (SS) which are streaming the same content in a synchronous way. It is also important to note that, every application run on a CDN structure, has a unique application server as well. This application server does nothing but redirects a user or client to the convenient or mostly available server where the content is available for receiving. To summarize, in our CDN setup there are 3 servers where two of them are SS and one of them is application server, all connected to a same hardware OpenFlow switch. As an

OpenFlow controller, we use Beacon due to its strong documentation and ease of use. Beacon is fully written in Java by David Erickson from Stanford University and it is an open-source OpenFlow controller as well. Besides, OpenFlow protocol serves an API on which several applications can be implemented. By using this opportunity, we implemented an emmbedded ALTO service on top of OpenFlow protocol, as can be seen in Figure 41.

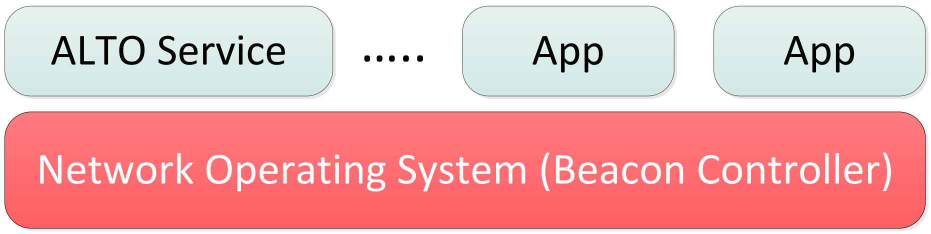


Figure 41: ALTO Service on top of Beacon Controller

The role of ALTO service in our demo is as follows; when a peer wants to join streaming via application server, ALTO service calculates and share the corresponding costs of suitable streaming servers which can be streaming server 1 or streaming server 2. Today's current CDN applications, application themselves make this desicion by using their own algorithms and intuitively it can be said that there is no need for an ALTO service. However with SDN technology, since all the topology is obtained by OpenFlow controller or controllers, an ALTO service and cost metrics may play a crucial role for these type of applications. Also, in this study we aim to show how easy to develop an application run on top of the controller as well. The overall topology, that we have setupped and partially explained above, is shown in Figure 42.

In this topology, when a peer wants to join streaming, it first sends a request to application server. Then, once this packet has arrived to the OpenFlow switch, it request for a suitable flow entry from the OpenFlow Controller. At that stage, our OpenFlow Contoller looks at the destination address of the flow either for the

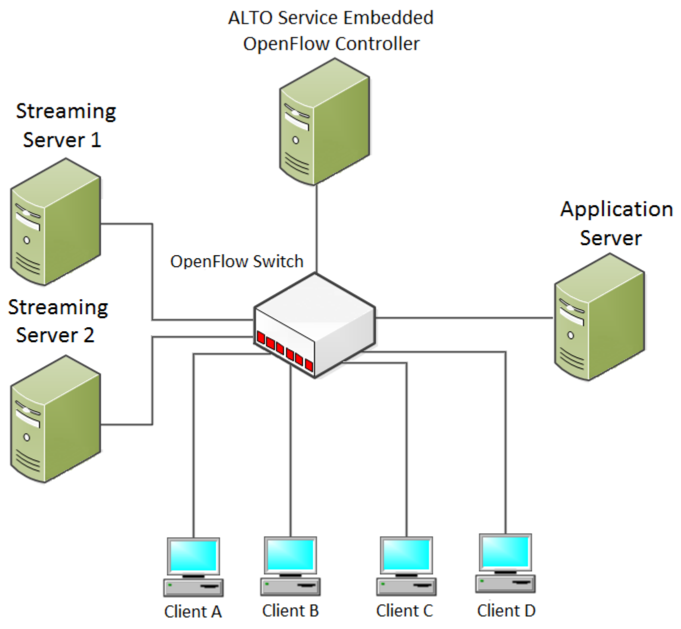


Figure 42: Installed Test-Bed Topology

calculation of the flow and for determining if the destination IP of the packet is belong to the CDN application server or not. If so, then ALTO Server implemented on top of the Controller determines the most convenient SS by just looking at the port traffics of the servers, and it chooses the one whose port traffic is lower. Once determining the most available SS, ALTO Service sends a packet to application server which includes the determined streaming server's IP address. Finally, after receiving the streaming request and ALTO information from the client and ALTO service respectively, application server sends a response packet to the client that contain the IP address of the corresponding SS's IP address and redirects the peer to that SS. Then, client starts a connection with the SS in order to start streaming.

In our demo, we see that when no client active (does not streaming) in the topology, ALTO service returns with a IP address of one of the SS since no user active. After that, when a second user joins to the streaming ALTO service returns the IP address of the streaming server different from the first one, in order to obtain load balancing issue. According to our observations the assigning process of the servers to the users has followed this structure.

$SS2 - SS1 - SS1 - SS2 - SS1$

As we see above, ALTO service distribute the users to the streaming servers according to the load balancing issue, and the scalability is doubled with that way. On the other hand, by devolving the server selection problem to central controller, application may have a chance to optimize its traffic in its infrastructures just by using ALTO service informations as well.

CHAPTER VII

CONCLUSION

In this thesis, we develop a generic ALTO Server and we simulate it with BitTorrent-Like file sharing application and P2P real time scalable video streaming application, respectively. Also in the last chapter, we also try to develop an ALTO service as an application on top of OpenFlow Controller, in order to provide source selection guiding mechanism for CDN structures. One of the aim, that we want to reach by emplyoying this service is, first to reduce cross-domain traffic rates as well as improve the performance results of the applications at the same time. Apart from these goals, ALTO service may also help CDN architectures to optimize latency, by providing the cost informations as well, especially in SDN structures. Our simulation and demo results show that, ALTO protocol may play a crucial role both for applications and service providers, by reducing Inter-ISP traffic rates and improving the performance parameters of the applications.

In our ALTO Server, we develop a cost calculation algorithm, based on minimizing the delay between source and destination. While calculating the corresponding delay, we use the current traffic rates on any link at a given time t and the corresponding bandwidth values between source and destination. Also a different cost metric calculation method, proposed in literature before, is explained and developed in order to compare the performance results of different approaches. Results demonstrate that, our proposed algorithm promises a bit more better performance than the other method from literature. Also as an important note, deploying ALTO Server (independent from which cost calculation algorithm we use) dramatically improves the performance of the simulations in terms of service providers as well.

Although the proposed method promises a high performance, it still needs some additional improvements or maybe modifications, since a real time link traffic determination is not a simple (actually a challenging) issue especially for the service providers that host millions of users and links.

Also, a crucial problem or in other words issue in P2P networks is, the dynamic behaviours of the peers. In these networks, peers typically join and leave from the overlay often and with an undetermined way, so this situation triggers partner and/or child changing scenarios especially in real time streaming applications. Therefore and unfortunately, this scenario can cause pause situations for the peers. In our simulations we assume that the peers are fully active until the end of the simulation. Therefore in order to get a more realistic and practically applicable results we may consider with that issue as well, as a future work.

Bibliography

[1] T. Bates, P. Smith, and G. Huston, "CIDR report," tech. rep., http://cidr-report.org/as2.0, Oct. 2011.

[2] "Internet study 2008/2009," tech. rep., http://www.ipoque.com.

[3] D. Srinivas, G. Abhiram, K. P. Rohith, and R. Sandeep, "Traffic types on Internet," tech. rep., Indian Institute of Technology Madras, Chennai, India, Sept. 2011.

[4] S. Ihm and V. S. Pai, "Towards understanding modern web traffic," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, IMC '11, (New York, NY, USA), pp. 295–312, ACM, 2011.

[5] M. Hofmann and L. Beaumont, *Content Networking: Architecture, Protocols and Practice*. San Francisco, CA: Elsevier, 2005.

[6] L. Braun, A. Klein, G. Carle, H. Reiser, and J. Eisl, "Analyzing caching benefits for youtube traffic in edge networks - a measurement-based evaluation," in *Network Operations and Management Symposium (NOMS), 2012 IEEE*, pp. 311 –318, april 2012.

[7] J. F. Buford, H. Yu, and E. K. Lua, *P2P Networking and Applications*. Burlington, MA: Elsevier, 2009.

[8] X. Shen, H. Yu, J. Buford, and M. Akon, *Handbook of Peer-to-Peer Networking*. New York, NY: Springer, 2010.

[9] J. F. Buford, "Management of peer-to-peer overlays," *Int. J. Internet Protoc. Technol.*, vol. 3, pp. 2–12, July 2008.

[10] G. Held, *A Practical Guide to Content Delivery Networks*. Boca Raton, FL: CRC Press, 2011.

[11] C. Wang, N. Wang, M. Howarth, and G. Pavlou, "An empirical study on the interactions between ALTO-assisted P2P overlays and ISP networks," in *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pp. 719 –726, oct. 2011.

[12] S. Guha, N. Daswani, and R. Jain, "An experimental study of the skype peer-to-peer voip system," in *In Proc of IPTPS*, 2006.

[13] H. Xie and Y. R. Yang, "A measurement-based study of the skype peer-to-peer voip performance," in *In Proc of IPTPS*, 2007.

[14] M. Shibuya, Y. Hei, and T. Ogishi, "ISP-friendly peer selection mechanism with ALTO-like server," in *Network Operations and Management Symposium (AP-NOMS), 2011 13th Asia-Pacific*, pp. 1 –8, sept. 2011.

[15] F. D'Elia, G. Di Stasi, S. Avallone, and R. Canonico, "Bittorrent traffic optimization in wireless mesh networks with ALTO service," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, pp. 1 –6, june 2011.

[16] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, "Should internet service providers fear peer-assisted content distribution?," in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, IMC '05, (Berkeley, CA, USA), pp. 6–6, USENIX Association, 2005.

[17] C. Wang, N. Wang, M. Howarth, and G. Pavlou, "A dynamic peer-to-peer traffic limiting policy for isp networks," in *Network Operations and Management Symposium (NOMS), 2010 IEEE*, pp. 317 –324, april 2010.

[18] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, modeling, and analysis of a peer-to-peer file-sharing workload," in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, (New York, NY, USA), pp. 314–329, ACM, 2003.

[19] N. Leibowitz, A. Bergman, R. Ben-shaul, and A. Shavit, "Are file swapping networks cacheable? characterizing p2p traffic," in *In Proc. of the 7th Int. WWW Caching Workshop*, 2002.

[20] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, "An analysis of internet content delivery systems," *SIGOPS Oper. Syst. Rev.*, vol. 36, pp. 315–327, Dec. 2002.

[21] D. R. Choffnes and F. E. Bustamante, "Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems," in *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, SIGCOMM '08, (New York, NY, USA), pp. 363–374, ACM, 2008.

[22] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante, "Drafting behind akamai (travelocity-based detouring)," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '06, (New York, NY, USA), pp. 435–446, ACM, 2006.

[23] G. Salton and M. Mcgill, *Introduction to Modern Information Retrieval.* New York, NY: McGraw-Hill, 1986.

[24] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4p: provider portal for applications," in *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, SIGCOMM '08, (New York, NY, USA), pp. 351–362, ACM, 2008.

[25] J. Seedorf and E. Burger, "Application-layer traffic optimization (ALTO) Problem Statement," tech. rep., Internet Engineering Task Force, Oct. 2009.

[26] R. Alimi, R. Penno, and Y. Yang, "Application-layer traffic optimization (ALTO) Protocol," tech. rep., Internet Engineering Task Force, 2012.

[27] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach (5th Edition)*. Boston, MA: Addison-Wesley, 2009.

[28] Y. Fukushima, K. Inada, Y. Tao, Y. Fujiwara, and T. Yokohira, "Performance evaluation of as-friendly peer selection algorithms for p2p live streaming," in *Communications, 2009. APCC 2009. 15th Asia-Pacific Conference on*, pp. 866 –870, oct. 2009.

[29] J. Qi, H. Zhang, Z. Ji, and L. Yun, "Analyzing bittorrent traffic across large network," in *Cyberworlds, 2008 International Conference on*, pp. 759 –764, sept. 2008.

[30] R. Xia and J. Muppala, "A survey of bittorrent performance," *Communications Surveys Tutorials, IEEE*, vol. 12, pp. 140 –158, quarter 2010.

[31] *www.bittorrent.com.*

[32] E. Setton and B. Girod, *Peer-to-Peer Video Streaming.* Reading, Massachusetts: Springer, 1984.

[33] S. Xie, B. Li, G. Keung, and X. Zhang, "Coolstreaming: Design, theory, and practice," *Multimedia, IEEE Transactions on*, vol. 9, pp. 1661 –1671, dec. 2007.

[34] V. Venkataraman, K. Yoshida, and P. Francis, "Chunkyspread: Heterogeneous unstructured tree-based peer-to-peer multicast," in *Network Protocols, 2006. ICNP '06. Proceedings of the 2006 14th IEEE International Conference on*, pp. 2 –11, nov. 2006.

[35] *www.pplive.com.*

[36] X. Zhang, J. Liu, B. Li, and Y.-S. Yum, "Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3, pp. 2102 – 2111 vol. 3, march 2005.

[37] U. Kozat, O. Harmanci, S. Kanumuri, M. Demircin, and M. Civanlar, "Peer assisted video streaming with supply-demand-based cache optimization," *Multimedia, IEEE Transactions on*, vol. 11, pp. 494 –508, april 2009.

[38] J. Liu, S. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer internet video broadcast," *Proceedings of the IEEE*, vol. 96, pp. 11 –24, jan. 2008.

[39] V. Padmanabhan, H. Wang, and P. Chou, "Resilient peer-to-peer streaming," in *Network Protocols, 2003. Proceedings. 11th IEEE International Conference on*, pp. 16 – 27, nov. 2003.

[40] P.-C. Liu, C.-W. Yi, Y.-T. Chuang, H.-H. Lu, J.-S. Leu, and W.-K. Shih, "On the capacity of tree-based p2p streaming systems," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, pp. 117 –122, 29 2010-april 2 2010.

[41] C. Liang, Y. Liu, and K. Ross, "Topology optimization in multi-tree based p2p streaming system," in *Tools with Artificial Intelligence, 2009. ICTAI '09. 21st International Conference on*, pp. 806 –813, nov. 2009.

[42] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live p2p streaming approaches," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 1424 –1432, may 2007.

[43] *www.opnet.com.*

[44] "The state of the internet: 1st quarter, 2012 report," tech. rep., http://www.akamai.com/stateoftheinternet/.

[45] *https://github.com/kierank/jsvm.*

[46] R. Bennesby, P. Fonseca, E. Mota, and A. Passito, "An inter-as routing component for software-defined networks," in *Network Operations and Management Symposium (NOMS), 2012 IEEE*, pp. 138 –145, april 2012.

[47] *http://opennetsummit.org/why.html.*

[48] *www.openflow.org/.*

[49] *openflow.stanford.edu/display/Beacon/Home.*

[50] *www.noxrepo.org/.*

[51] *floodlight.openflowhub.org/.*

VITA

Koray Kökten received his undergraduate education from the Department of Electronics Engineering at Istanbul Technical University in 2010. His undergraduate thesis topic was Applications of Tunable Current Mirrors and his advisor was Prof. Dr. Ali Toker. He is currently a graduate student and research assistant at Ozyegin University, Istanbul. He is working with Prof. M. Oğuz Sunay on multimedia transmission over P4P networks.