International Journal of Digital Multimedia Broadcasting

Network-Aware Peer-to-Peer (P2P) and Internet Video

Guest Editors: John F. Buford, Ken Kerpez, Yuanqiu Luo, Dave Marples, and Stan Moyer

Network-Aware Peer-to-Peer (P2P) and Internet Video

International Journal of Digital Multimedia Broadcasting

Network-Aware Peer-to-Peer (P2P) and Internet Video

Guest Editors: John F. Buford, Ken Kerpez, Yuanqiu Luo, Dave Marples, and Stan Moyer

Contents

Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 283068, 2 pages
doi:10.1155/2010/283068

Editorial
Network-Aware Peer-to-Peer (P2P) and Internet Video

John F. Buford,[1] Ken Kerpez,[2] Yuanqiu Luo,[3] Dave Marples,[4] and Stan Moyer[2]

[1] *Avaya Labs Research, 233 Mount Airy Road, Room 2A11, Basking Ridge, NJ 07920-2311, USA*
[2] *Telcordia Technologies, 1 Telcordia Drive, Piscataway, NJ 08854, USA*
[3] *Huawei Technologies USA, 400 Somerset Corporate Boulevard, Suite 602, Bridgewater, NJ 08807, USA*
[4] *Technolution B.V., P. O. Box 9202, Mansfield Notts NG18 9DY, UK*

Correspondence should be addressed to Ken Kerpez, kkerpez@telcordia.com

Received 10 March 2010; Accepted 10 March 2010

Video and peer-to-peer content are both rapidly increasing Internet bandwidth demands. The recent reports predict an "exaflood" from advances in video over the Internet, rich media content, and user-generated content (UGC). Another trend is the continued popularity and growth of peer-to-peer (P2P) content delivery. Many estimates say that P2P accounts for *most* of the current Internet traffic, and that video accounts for *most* of the growth in Internet traffic. New systems and studies to optimize future P2P and video traffic can have a very high impact on the future of the Internet.

In many cases, P2P traffic traverses long distances, across core networks and multiple Internet Service Provider (ISP) networks, even though the content could have been retrieved from a much closer location. Internet video is similar; often delivered from distant servers via multiple, redundant, unicast streams. To address this, there has been a spate of recent effort on systems, information exchange, and control to enable efficient P2P and video content distribution. New systems and protocols are needed to enable Internet content to work in concert with the network to be delivered from the best source or over the least congested links. Localized P2P traffic may only traverse a few hops instead of ten or twenty, allowing a vast decrease in core network bandwidth. Real-time streaming via P2P is becoming competitive with traditional broadcast, and enhanced live P2P protocols are emerging. Related fields such as automotive telematics are embracing P2P techniques. Efficient measurement and analyses are necessary to understand new P2P protocols.

These topics and more are investigated in this special issue. Many thanks go to the paper authors and guest editors for providing the timely and high-quality analyses here. A summary of papers is as follows.

"RTSP-based mobile peer-to-peer streaming system," by Jani Antero Peltotalo et al., presents an effective real-time peer-to-peer streaming system for the mobile environment. The basis for the system is a scalable overlay network which groups peers into clusters according to their proximity using round-trip time (RTT) values between peers. Media is delivered using a number of "partial streams."

"Localized multistreams for P2P streaming," by Majed Alhaisoni et al., looks at the combination of two P2P streaming techniques, redundant streaming, and locality awareness, in the context of both live and video-on-demand streaming. Results show that redundancy affects network utilization only marginally if traffic is kept at the edges via localization techniques.

"Bandwidth reduction via localized peer-to-peer (P2P) video," by Ken Kerpez et al., presents recent research into P2P distribution of video that can be highly localized, preferably sharing content among users on the same access network and Central Office (CO). Results show that nearly all of the traffic volume of unicast video could be delivered via localized P2P.

""Q-Feed"—an effective solution for the free-riding problem in unstructured P2P networks," by S. M. Thampi and Chandra Sekaran K, presents a solution for reducing the ill effects of free riders in decentralised unstructured P2P networks. "Q-learning" is applied to a free-riding control mechanism, which is shown to effectively service what is received by free riders and also encourage the low-performing neighbors to improve performance with a higher quantity of popular files.

"A hybrid approach to assess the network awareness of P2P-TV applications," by D. Rossi and P. Veglia, develops a general methodology to assess the level of network awareness

and friendliness of P2P-TV applications that is based on a combination of active and passive measurement techniques. This methodology is applied to PPLive, where it is shown that PPLive generally does not show preference toward peer proximity.

"Providing adapted contextual information in an overlay vehicular network," by Jose Santa et al., discusses P2P in the context of automotive telematics. The vehicular network presented in this paper fills the current gap between solutions lacking in flexibility, mainly supported by an infrastructure deployment, and those highly local and distributed, such as sole- vehicular ad hoc network (VANET) approximations.

John F. Buford
Ken Kerpez
Yuanqiu Luo
Dave Marples
Stan Moyer

Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 470813, 15 pages
doi:10.1155/2010/470813

Research Article

RTSP-based Mobile Peer-to-Peer Streaming System

Jani Peltotalo,[1] Jarmo Harju,[1] Lassi Väätämöinen,[1] Imed Bouazizi,[2] and Igor D. D. Curcio[2]

[1] *Department of Communications Engineering, Tampere University of Technology, P.O. Box 553, 33101 Tampere, Finland*
[2] *Nokia Research Center, P.O. Box 1000, 33721 Tampere, Finland*

Correspondence should be addressed to Jani Peltotalo, jani.peltotalo@tut.fi

Received 1 June 2009; Revised 12 November 2009; Accepted 6 January 2010

Academic Editor: John Buford

Peer-to-peer is emerging as a potentially disruptive technology for content distribution in the mobile Internet. In addition to the already well-known peer-to-peer file sharing, real-time peer-to-peer streaming is gaining popularity. This paper presents an effective real-time peer-to-peer streaming system for the mobile environment. The basis for the system is a scalable overlay network which groups peer into clusters according to their proximity using RTT values between peers as a criteria for the cluster selection. The actual media delivery in the system is implemented using the partial RTP stream concept: the original RTP sessions related to a media delivery are split into a number of so-called partial streams according to a predefined set of parameters in such a way that it allows low-complexity reassembly of the original media session in real-time at the receiving end. Partial streams also help in utilizing the upload capacity with finer granularity than just per one original stream. This is beneficial in mobile environments where bandwidth can be scarce.

1. Introduction

Peer-to-Peer (P2P) streaming applications are gaining more and more users around the world. These applications allow end-users to broadcast content throughout the Internet in real-time without the need for any special infrastructure, since the user's device, together with all other peers, collectively forms the infrastructure. Furthermore, dedicated servers are no longer required since every peer can serve data to other peers. This is in contrast to a service like YouTube [1] which still requires content to be uploaded to a central server first. Some of the currently existing P2P streaming applications, such as Octoshape [2] and SopCast [3], are suitable to be used in a mobile environment but still there are many issues to be solved before an optimized solution for mobile devices can be realized [4].

With real-time P2P streaming there is no need to download the entire media file before playback can be started. Decoding can be started as soon as enough data is buffered in the peer. This avoids long startup times, and eliminates the need to store the entire content on the mobile device which still has a relatively small amount of internal memory compared to the increasing size of the actual media. In live streaming, video of an ongoing event, like a football match, is delivered as a stream in real-time. After an initial buffering period, the user starts to watch the stream from a certain location and all peers consume data in the same time window. With a Video-on-Demand (VoD) streaming the user searches a video from some catalogue, and after a certain amount of initial buffering the user starts to play the video from the beginning.

In order to increase the robustness and to accommodate the limited up- and download bandwidth between peers in the network, the original multimedia session needs to be split into smaller parts, which can be reassembled at the receiving peers into the original media representation. This paper presents an effective real-time P2P streaming system where original Real-time Transport Protocol (RTP) [5] sessions related to a media delivery are split into a number of so-called partial streams according to a predefined set of parameters. This approach allows low-complexity reassembly of the original media session in real-time at the receiving end.

The structure of the remainder of this paper is as follows. The related work is discussed in Section 2. Then, a short

overview of the system is given in Section 3. Detailed descriptions of the overlay network and the media delivery are given in Sections 4–8. After that, results from the performance experiments are presented in Section 9. Interesting areas for further work are discussed and highlighted in Section 10. Finally, Section 11 concludes this paper.

2. Related Work

Many P2P file sharing applications make use of multiple source distribution. A file is first partitioned into pieces or chunks, typically of equal size. A peer then connects to the seeder or leecher peers, and requests the missing pieces of the file in a random order. The difference between a seeder and a leecher peer is that the former has a complete copy of the file while the latter has only a partial copy. For example, with BitTorrent [6] the complete multimedia file can be partitioned into blocks of 256 KB which are then selected by the interested peers and requested according to a rarest-first piece selection algorithm. This approach is not at all suitable for streaming applications as it does not consider the delay problem. Users may experience long download delays of possibly several days. It also assumes that the full content is known and available at the source peers, which does not necessarily apply to streaming applications as the stream may be live.

A P2P multimedia streaming solution based on the BitTorrent protocol is proposed in [7]. The rarest-first chunk downloading policy is replaced by a policy where peers first download chunks that will be consumed in the near future. The tit-for-tat peer selection policy is also modified to allow free tries to a larger number of peers to let peers participate sooner in the multimedia distribution. Another P2P streaming system based on a P2P file sharing implementation was proposed already in [8]. However, the data partitioning based on fixed byte ranges is not suitable for streaming a continuous media, which is of variable bit rate nature.

In P2P content distribution, an overlay network is created at the application layer in order to transfer the actual content among peers in the network. A random mesh-based overlay architecture, like in [9, 10], provides flexibility for handling peer departures, but good general connectivity between peers is not usually achieved. There have been many studies about how to organize peers in an efficient and scalable way. In [11] receivers are organized into a hierarchy of bounded-size clusters and the multicast tree is built based on that. In [12] peers are organized into a directed acyclic graph to enable peers to obtain locality awareness in a distributed fashion. To improve the file sharing performance of the BitTorrent protocol, an overlay network where peers are grouped into clusters according to their proximity is proposed in [13]. Even though some solutions have proven their functionality with wired connections, those might not be suitable for the mobile environment.

Preliminary results of the mobile P2P system described in this paper have been published in [14]. In the following sections more information about the system is given by explaining in detail the extended Real Time Streaming Protocol (RTSP) [15] messages used for signalling and providing more results from the experiments.

3. General System Overview

The architecture of the system is designed to be scalable and efficient for real-time streaming services in the mobile environment. The system supports both live and VoD streaming services. Location awareness in terms of peer proximity has been exploited to reduce delay, and thus, to improve the scalability of the system.

Peers are grouped into clusters according to their proximity in order to efficiently exchange data between peers. For VoD streaming services, the clusters could be constructed for example, based on the interest level for certain pieces of data, so that the peers watching the same part of a video at the same time belong to the same cluster. In live streaming services a cluster can be formed only based on the proximity of peers, because all peers are interested in the same data pieces within the same time window. Clusters will also help with scalability issues of peer maintenance. Peers inside a cluster are considered to be close to each other and thus communication between peers can be done more efficiently.

All overlay network operations in the system are implemented using extended RTSP messages. All RTSP methods are extended to include an additional `RTP2P-v1` tag in the `Require` header field. This tag makes it possible for the receiving peer to detect that support for the real-time P2P extensions is needed. Additionally, all RTSP messages will include a `Peer-Id` header field to indicate the source of a message. The most important new header field is called `overlay` and it is used widely in the overlay network operations. The usage of the `Overlay` header field and other additional header fields depending on the message type are explained in Sections 4 and 7. The syntaxes of the `Peer-Id` and `Overlay` header fields in Augmented Backus-Naur Form (ABNF) [16] are given below:

```
Peer-Id = "Peer-Id:" SP id CRLF id = 1*DIGIT

Overlay = "Overlay:" SP operation CRLF
operation = "backup" | "create" | "join_bcl" |
            "join_neighbor" |"join_peer" | "leave" |
            "new_peer_id" | "remove" | "split" |
            "update"
```

The RTSP Uniform Resource Locator (URL) is formatted according to the ABNF syntax shown below. The `host` and `port` parts are defined in [17, Section 3.2]. The `service-id` specifies the service and the `stream-id` specifies the RTP session. Like in [15], it is also possible to use the asterisk character instead of the URL meaning that the request does not apply to any particular resource:

```
rtsp_URL = "rtsp://" host [":" port] ["/" [ service-id
           ["/" stream-id]]]
service-id = 1*DIGIT
stream-id = 1*DIGIT
```

Peers exchange actual media data between each other using RTP. The system is using time-based chunking, which

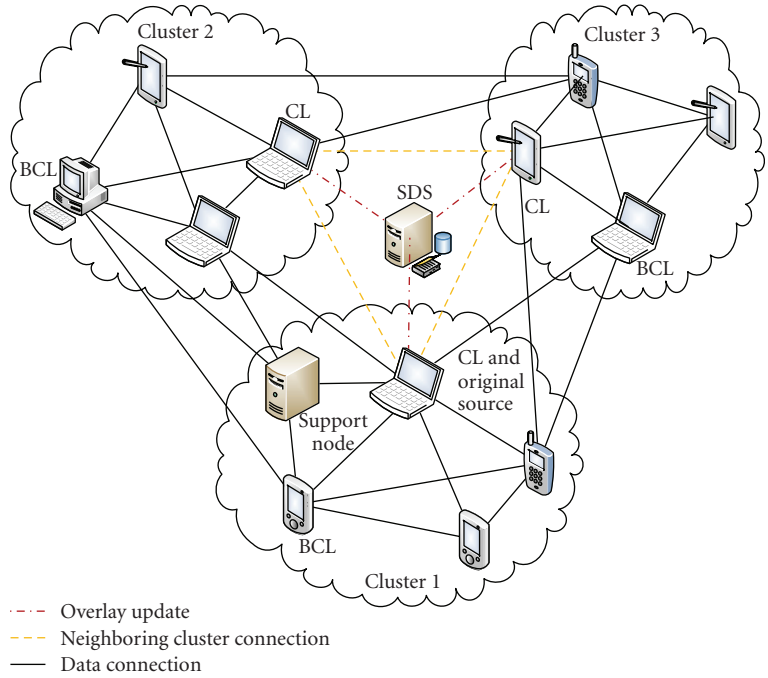


Figure 1: Overlay architecture.

creates multiple partial streams from an original RTP session. This implements multisource streaming in a way that each sender sends bursts of data from a different partial stream. Multisourcing will help to cope with the dynamics of mobile peers and distributes bandwidth usage in the system more flexibly and evenly. The original data stream source generates also RTP timestamps and sequence numbers for the RTP delivery. Timestamps and sequence numbers are delivered unchanged within the streaming service. This is done for allowing RTP packets from multiple partial streams being reassembled in the correct sequence order at each peer for local playback. The RTP time line is known system-wide, so the timestamps can be used to uniquely identify individual packets within a streaming service.

4. Overlay Network Architecture

The architecture of the overlay network with three clusters sharing a certain streaming service, such as a live stream channel or a VoD movie, is presented in Figure 1. It should be noted that for every different streaming service such an overlay network is maintained separately. The Service Discovery Server (SDS) is a central nonmobile server containing information about cluster hierarchy and the available streaming services in the system.

When a peer wants to join the P2P overlay network, a peer identifier (ID) is first requested from the SDS using an `RTSP OPTIONS` message with a `new_peer_id` tag in the `Overlay` header field. Because the peer does not have a peer ID yet, it must set the value to minus one in the `OPTIONS` message. A unique peer ID is then returned by the SDS using a `200 OK` message with a `New-Peer-Id` header field. The syntax of the `New-Peer-Id` header field in ABNF is given below:

```
New-Peer-Id = "New-Peer-Id:" SP id CRLF
id = 1*DIGIT
```

The cluster concept is implemented with the help of Cluster Leaders (CLs). There is one CL assigned to each cluster with the possibility for one or more Backup Cluster Leaders (BCLs). CLs are used to manage peers inside the cluster and to connect new arriving peers. Each ordinary peer must perform periodical keep alive messaging to inform its existence to the CL and all other peers from which it has received RTP packets. The latter is done to avoid unnecessary data transmission because RTP uses User Datagram Protocol (UDP) and the sending peer does not otherwise know that the receiving peer is still in the network. A new arriving peer can select a suitable cluster according to its best knowledge of locality using Round Trip Time (RTT) values between CLs and itself.

In addition to RTT measurements, location awareness could be also based on, for example, IP level hop count, geographic location or some combination of these three mentioned metrics. IP level hop count is not alone suitable for proximity metric, since with Virtual Private Networks (VPNs) or other tunneling techniques one hop might actually consist of a large number of hops and the distance could be quite long. Nor does small IP level hop count guarantee small delay, because it does not take connection speed into account. Geographic location is also little problematic in IP level point of view. Even if peers are geographically close to each other, the IP level routing path could circulate through distant router. Hence, only RTT values are used in our system for proximity checks.

CLs are nodes with suitable capabilities, such as a high throughput access network connection, enough memory and CPU power, and long-expected battery lifetime. One cluster should contain only a limited number of peers in order to sustain system scalability. The CL collects statistical data of the peers participating in a cluster. This statistical data contains information about service join time, reception buffer position, missing RTP packets, and upstream and downstream connections, and can be used to make the decision of the best peer from which to start downloading data. Statistical information can be used to, for example, filter out candidate source peers which already have many upstream connections or lots of missing RTP packets. Service join time can be used to estimate the behavior of the peer. If the peer has joined to the service very long time ago, it is most likely a stable peer which will provide data in the future also. On the other hand, without extra information about the expected battery lifetime with mobile devices, long service joining time can also mean short-expected battery lifetime.

The CL is a functional entity in the network and may also participate as an ordinary peer at the same time, by receiving and sending media data. Thus, the CL can be seen as a functional extension of an ordinary peer. The CL will inform the SDS currently at ten seconds intervals about changes in the cluster by sending an `OPTIONS` message with an `update` tag in the `Overlay` header field in order to maintain an up-to-date cluster list at the SDS. The updated cluster information will be expressed using an Extensible Markup Language (XML) [18]. To decrease the amount of data delivered in the network, all XML fragments are compressed using `deflate` compression mechanism from the zlib data compression library [19].

While joining the selected cluster, a peer receives an initial list of peers from which the actual media data can be acquired. Naturally, the corresponding CL inserts joined peers into its peer list, and if the amount of peers is very large, the CL can return only a subset of peers. Proximity testing in the peer selection is optional since the cluster selection procedure guarantees that peers are reasonably close to each other. Anyway, a peer which finally selects its sources for the stream, needs to test a certain amount of peers until suitable ones are found. The peer can later receive updates of the peer list while performing periodical keep alive messaging to the CL, which ensures that the peer list can be kept up-to-date during the streaming service.

The peer's contact information, that is, all information needed for contacting the peer, could include also a cluster ID, so that peers can prioritize connections within their own cluster. However, there should always be data connections between peers that are located in different clusters. This ensures that clusters do not become separate islands having only one incoming connection from other clusters, which would form a single point of failure that could cause problems later on when that peer leaves the streaming service.

4.1. Service Creation and Initial Cluster. The message exchange during service creation is presented in Figure 2. When a peer wants to create a service, an `ANNOUNCE` message will be sent to the SDS. A `Client-Port` header field indicates the port number to be used in the overlay communication. The service is described using the Session Description Protocol (SDP) [20]. Two new SDP attributes, `service-type` and `stream-info` are used to signal the service information. The `service-type` attribute defines the type for the service, and the `stream-info` attribute defines the identifier for the RTP session and parameters to be used in the RTP session partitioning explained in Section 7. The syntaxes for the `service-type` and `stream-info` attributes and the `Client-Port` header field in ABNF are given below:

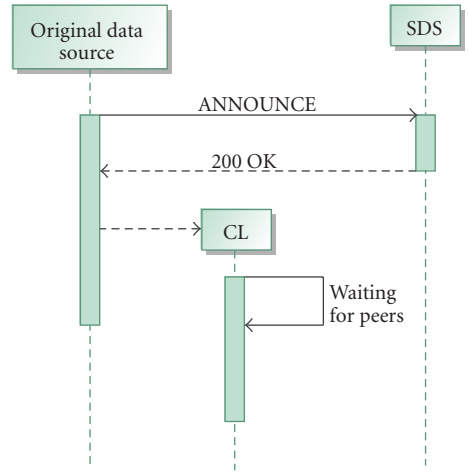


Figure 2: Creation of the service and the initial cluster.

```
service-type-line = "a=service-type:" type CRLF
type = "live" | "vod"

stream-info-line = "a=stream-info:" id";" piece-size";"
                   nb-of-partials ";" CRLF
id = "id=" 1*DIGIT
piece-size = "piece-size=" 1*DIGIT
nb-of-partials = "nb-of-partials=" 1*DIGIT

Client-Port = "Client-Port:" SP port CRLF
port = 1*DIGIT
```

As a response to the successful session creation, a `200 OK` message is sent by the SDS. The message contains the `Cluster-Id` and `Service-Id` header fields to describe the IDs for the initial cluster and the newly created service, respectively. A `301 Moved Permanently` message can also be sent if the SDS has been moved to another location. In a redirection case the `Location` header field must be present informing the new location of the SDS. Any other message type must be interpreted as a failed session creation. The syntaxes of the `Cluster-Id` and `Service-Id` header fields in ABNF are given below:

```
Cluster-Id = "Cluster-Id:" SP id CRLF
id = 1*DIGIT

Service-Id = "Service-Id:" SP id CRLF
id = 1*DIGIT
```

There are two possibilities for creating the initial cluster and selecting a CL for it: (a) the first peer joining the service

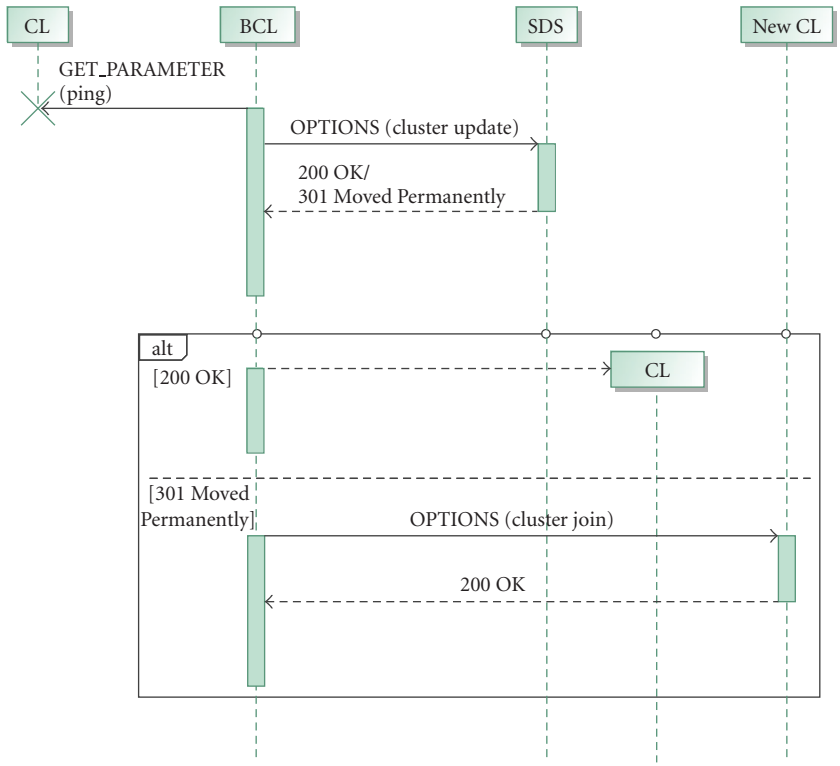


Figure 3: Uncontrolled CL departure.

will be assigned as a CL by the SDS, and (b) the original data source will be the first CL in the service. The latter possibility uses more resources from the original data source and therefore the original data source should be released from the CL responsibilities when possible. However, the latter possibility also guarantees that the initial cluster remains operational because the first joining peer might depart from the service quite quickly. Hence, the alternative (b) is used in our system.

When the service is successfully created, the original data source becomes CL for the initial cluster, which is illustrated by the dashed line without message type in Figure 2, and starts to wait for other peers to join the service. When new peers are joining the service, BCLs are assigned by the CL by using an `OPTIONS` message with a `backup` tag in the `Overlay` header field. If a peer accepts the BCL assignment, it sends a `200 OK` message, and if not, it will send a `403 Forbidden` message.

The service is updated and removed by using an `ANNOUNCE` message. If some part of the information have changed, the SDS updates the information in the database. To remove a service, the stop time in the SDP `t-line` should be set smaller than the prevailing system time, which means that the service has been stopped and the SDS can remove the service from the database. To a successful service update or removal the SDS will respond with a `200 OK` message, otherwise the SDS will return `400 Bad Request` or `404 Not Found` messages.

4.2. Cluster Leader Departure. When the CL leaves the network it needs to be replaced by one of the BCLs. If a cluster does not have an active CL, new peers cannot be accepted into the network. However, this does not affect the data streaming connections between existing peers because the streaming and overlay connections are independent. New peers cannot be discovered by normal peers during the cluster leader change, but this should not be an issue because peers should have knowledge about more peers than they are using.

The message exchange in the event of an uncontrolled CL departure is presented in Figure 3. When the BCL does not get a response to its periodical `GET_PARAMETER` message, it concludes that the CL has left from the cluster and contacts the SDS using an `OPTIONS` message with an `update` tag in the `Overlay` header field to replace the old CL. The source of the first received `OPTIONS` message will be assigned as a new CL, illustrated by the dashed line without message type in the figure, and the new arriving peers can normally start using the new CL. All other BCLs will receive a `301 Moved Permanently` message with the information about the new CL and will send an `OPTIONS` message with the `join_bcl` tag in the `Overlay` header field to the new CL and will continue in the BCL role. If the original CL has not left the cluster but has had connectivity issues, it is redirected to the new CL by the SDS. In this case the old CL becomes a BCL.

When a peer notices that the CL is not available, it tries to connect to the known BCLs. If the BCL has replaced the

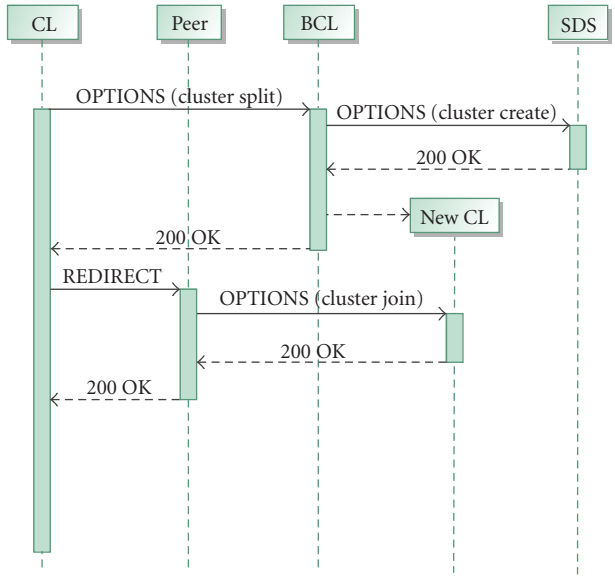


Figure 4: Successful cluster splitting procedure.

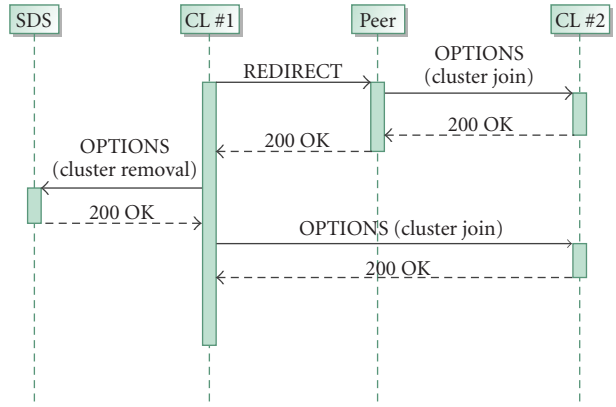


Figure 5: Successful cluster merging procedure.

old CL, it accepts the connection with a `200 OK` message, otherwise it sends a `301 Moved Permanently` message with the `Location` header field indicating the location of the last known CL. It should be noticed that the CL departure is not an atomic operation and takes some time, and therefore there can be short times when any one of the BCLs does not know the correct CL of the cluster. If none of the BCLs in the list respond, the peer sends a query to the SDS and asks for a new cluster which it can join.

4.3. Cluster Splitting and Merging. When the cluster grows too large to be handled by a single CL, the cluster should be split into two separate clusters. The existing CL assigns one of its BCLs to become a new CL for the new cluster, and redirects a number of existing peers to the new cluster. The message exchange in the successful cluster splitting procedure is presented in Figure 4.

Cluster splitting is performed by using an `OPTIONS` message with the `split` tag in the `Overlay` header field. After receiving this message, the BCL will inform the SDS that wants to become the CL of a new cluster by sending an `OPTIONS` message with a `create` tag in the `Overlay` header field. To a successful cluster creation, the SDS will respond with a `200 OK` message, which contains a `Cluster-Id` header field to describe the ID for the new cluster. Otherwise, the SDS will return a `400 Bad Request` message if the request message format is not valid, or a `404 Not Found` message if the service is not available anymore. After a successful cluster creation, the BCL will become the CL of the new cluster, and replies to the splitting CL by sending a `200 OK` message, otherwise it must send a `400 Bad Request` message to the splitting CL and wait for further instructions.

The splitting CL then sends a `REDIRECT` message, with the location of the new CL in the `Location` header field to those peers that should change the cluster. The redirected peers will then join the new cluster by sending an `OPTIONS` message to the new CL. After a successful cluster join, that is, the peer received a `200 OK` message from the new CL, the peer will send a `200 OK` message to the splitting CL. Otherwise, the peer will send a `400 Bad Request` message to inform that it is not possible to join to the new cluster.

The overlay connections between the CLs are created after a successful splitting by sending an `OPTIONS` message with a `join_neighbor` tag in the `Overlay` header field and a `200 OK` message. This connection is subsequently used to exchange cluster information expressed using XML fragments between neighboring clusters.

Merging of two clusters must be done when a cluster becomes too small. If the amount of peers is too small, a new joining peer will get a very small list of data sources which makes the functionality less reliable when one of these peers leaves the service. The message exchange in the successful cluster merging procedure is presented in Figure 5.

The merging is started by the CL, when the amount of peers in the cluster drops below some predefined threshold, by sending a `REDIRECT` message to all peers in the cluster. Peers will then join the new cluster, selected by the merging CL from its neighbor clusters, by sending an `OPTIONS` message to the new CL. After a successful cluster join, the peer will send a `200 OK` message to the merging CL. If the redirected peer does not receive any response from the new CL or it receives a `400 Bad Request` message, it must send a `400 Bad Request` message to the merging CL to inform that it is not possible to join to the new cluster and wait for further instructions.

After all peers in the cluster have confirmed the cluster change, the merging CL will remove the cluster by sending an `OPTIONS` message with a `remove` tag in the `Overlay` header field to the SDS. This message is optional, because the cluster is removed also automatically if a keepalive message has not been received during a certain interval. To a successful cluster removal, the SDS will respond with a `200 OK` message. Otherwise, the SDS will return a `400 Bad Request` message if the request message format is not valid, or a `404 Not Found` message if the cluster is not available anymore. The merging CL itself then must send a cluster join message, that is, an `OPTIONS` message with a `join_bcl` tag in

the `Overlay` header field, to a known neighbor and join as a BCL.

All overlay network connections are maintained by sending `GET_PARAMETER` and `200 OK` messages between peers as keep alive messages. Keep alive messages between neighboring CLs are exchanged at 20 seconds intervals and are used to exchange information about neighboring clusters. Keep alive messages between the CL and the BCL are used to deliver cluster information to the BCL and these messages are sent at 30 seconds intervals.

5. Partial RTP Streams

In order to have unique sending slots for each of the sources, a partial RTP stream concept is introduced in this paper. First, every RTP session, such as video, audio or subtitle streams of the entire multimedia session is split into smaller pieces along the time axis. Each of the pieces has a fixed duration T_P which is expressed in time. The start time is aligned with the start time of the RTP time base, that is, the start of the first piece is located at the origin of the RTP time line. One of the benefits of taking time as a unit is that all packets can remain intact at the RTP layer. Segmentation at the RTP packet level is not required for creating the partial streams. This significantly reduces the complexity of the implementation.

In the second step, RTP packets belonging to each of the RTP sessions are assigned to N partial streams according to (1), where i denotes the index of the partial stream ($0 \leq i < N$) and t_{RTP} denotes the RTP timestamp as carried in the RTP data packet. The algorithm allows assigning every RTP packet in the session to a partial RTP stream without having to maintain the state in the peer itself. Only by examining the RTP timestamp in combination with the constant parameters is sufficient to identify the partial stream the RTP packet is assigned to. Assuming a timeline of a single RTP session, this process is illustrated in Figure 6, where $T_C = N * T_P$ is used to denote the cycle time:

$$i = \text{floor}\left(\frac{t_{\text{RTP}}}{T_P}\right) \bmod N. \tag{1}$$

Note that the piece size T_P should be selected in such a way that it is large enough to contain at least one RTP packet on average. If it is chosen too small, not every piece will have data, which may in the extreme case lead to an empty partial stream. On the other hand, larger cycle times lead to longer startup times, since a complete cycle needs to be buffered before seamless playback can be guaranteed. In one particular type of partitioning, every piece would start with an intracoded picture. This would facilitate independent decoding of partial streams in the presence of packet loss due to the fact that a partial stream is not being received. This could for instance easily be achieved by aligning the pieces with group-of-picture (GOP) boundaries. Enhanced robustness will also be achieved by assigning key (RTP) packets to multiple partials. Key packets could for instance be Instantaneous Decoding Refresh (IDR) picture data or other data that would help error concealment. Duplicate RTP packets would simply be removed upon reception and would therefore not affect the basic algorithm or its complexity.

The number of partial streams, N, can vary per RTP session. For instance it may not be very useful to partition an audio stream into lower bit rate partial streams if the bit rate of the entire RTP audio session is already in the order of magnitude of a single partial RTP video stream. This also has the additional advantage that the audio is received either in its entirety or not at all, thereby reducing annoying audible artifacts in the case of partial stream loss.

The number of partial streams does not necessarily need to be constant throughout the P2P network within a particular streaming service. As a matter of fact it is possible to vary N at every forwarding peer in the network. However, choosing the same N throughout the network simplifies the design of the partitioning functionality.

6. Media Delivery Mechanism

All peers in the streaming network are forming a non-hierarchical mesh structure. Peers are connected to several other peers and are receiving data from and sending data to multiple other peers. An example mesh layout can be seen in Figure 1.

A peer may request the delivery of one or more partial streams from another peer. A partial stream is the smallest granularity for media streaming, that is, a peer may not stream a fraction of a partial stream. The number of partial streams can be tuned to achieve the target bit rate of a partial stream. Each peer in the network should have enough uplink bandwidth to be able to stream at least a single partial stream.

In this kind of streaming network, where most of the peers are from the same cluster, some kind of intelligence to avoid loops is needed. Such loops occur when a sender starts receiving its own data via a number of intermediate peers in the mesh network. To avoid loops, an algorithm based on streaming path in the form of list of ancestors is used. The path for the data stream in the application level containing peer IDs which have forwarded the stream is delivered using the contributing source (CSRC) list in the RTP packets. This list is then used to avoid accepting connections from peers who are already in the list and for dropping connections if a peer notices that it is in the list.

Figure 7 illustrates media delivery among four peers. Arrows between each peer denote an active RTP session, and the direction defines the data flow direction. A sourcing peer can send multiple partial streams to a particular receiving peer. This allows for a smaller granularity of rate adaptation between two individual peers in the network. These multiple partial streams could either be streamed in a single RTP session or separate RTP sessions. Peers in the figure are numbered and colored. The smaller the number is, the earlier the peer has joined the network; in this case peer number one is the original source. Different colors in the peers buffers show the origin of the received data. For simplicity, the value four is used for the number of partials for each peer.

In order to receive a complete stream, a peer must receive all partial streams; in this example, partials 0, 1, 2, and 3.

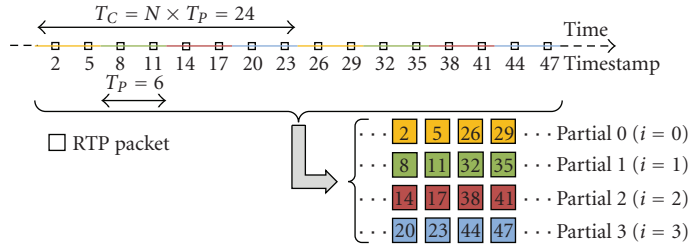


Figure 6: RTP stream partitioning.

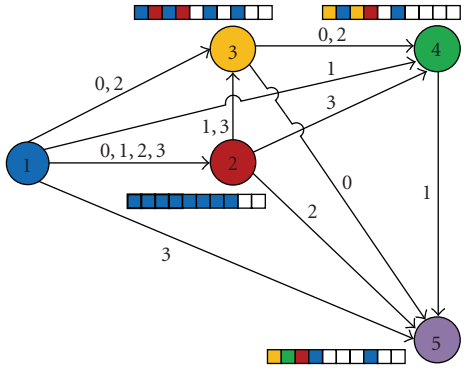


Figure 7: Partial RTP stream delivery.

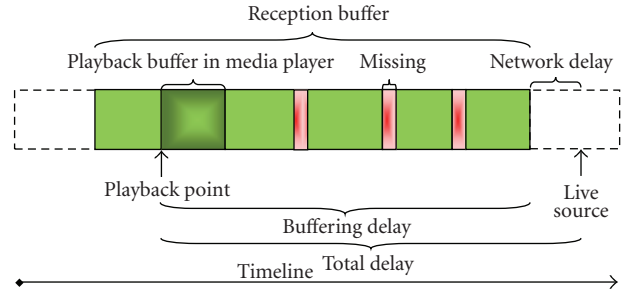


Figure 8: Data buffering.

Peer number two is receiving all partials from the original data source, that is, peer number one, and is forwarding partials to peers number three, four, and five. Peer number four is also receiving partials from peer number three, and peer number five from peers number three and four. All incoming packets for all of the partial streams that constitute a particular RTP session are added to a separate buffer pool. This buffer pool maintains the information about the destination peer to which the incoming packets need to be forwarded. Every incoming packet is examined and assigned to one of the outgoing queues. In case the peer is playing back the received streams locally, the local player is considered to be a destination as well.

Packets destined for a particular receiving peer are transferred from the buffer to the outgoing RTP queue as soon as they have been received. In case of local playback, that is, the receiving peer is decoding and rendering the received RTP session, the packets are also forwarded internally to the media player. Buffering is needed to recover from peer departures and consequential missing data. The peer can ask a replacement peer to send the missing data from its buffer. There is also a possibility that a peer does not have certain part of the buffer available because of bad network conditions, for example, insufficient bandwidth. These parts might be later downloaded if needed. Because of the missing data, the playback is interrupted during this period and it depends on the used codec how missing data will affect on Quality of Experience (QoE). Data will most probably not come in order or in time from all sources as the delay could vary very much between different sources, so the buffer is also needed to collect all the data from different partial streams and arrange the data in order before passing it to the local media player.

When all partial streams are received almost at the same time, the buffering delay can be reduced and the data can be passed earlier to the media player. However, the situation may change during time, and a constant buffer delay is currently used in the implementation. Data buffering is presented in Figure 8. The playback buffer is the buffer located in the media player. When peers consume media, the reception buffer is simultaneously shared with other peers. The total delay from content generation to receiver playback is the sum of the network delay, buffer delay, and media player playback delay.

In addition to the reception buffer, data storage via caching should be used in a VoD streaming service. When using caches, the VoD data can be distributed away from the original data source. This helps relieving the network load from the original data source, as new peers joining the network are able to download VoD content from multiple sources instead of relying on the original data source. Caching could be implemented in many ways, in the simplest model all peers store all data they have consumed. If a peer does not have enough storage capacity, the data belonging to a specific partial stream could be cached instead. Alternatively, a peer can limit the amount of cached data by applying a sliding time window. In the former case, an algorithm based on peer IDs could be used to determine which partial stream should be stored. This kind of partial caching requires that the amount of peers in a streaming service is reasonably high. To ensure data availability from multiple peers in every situation, support nodes, which

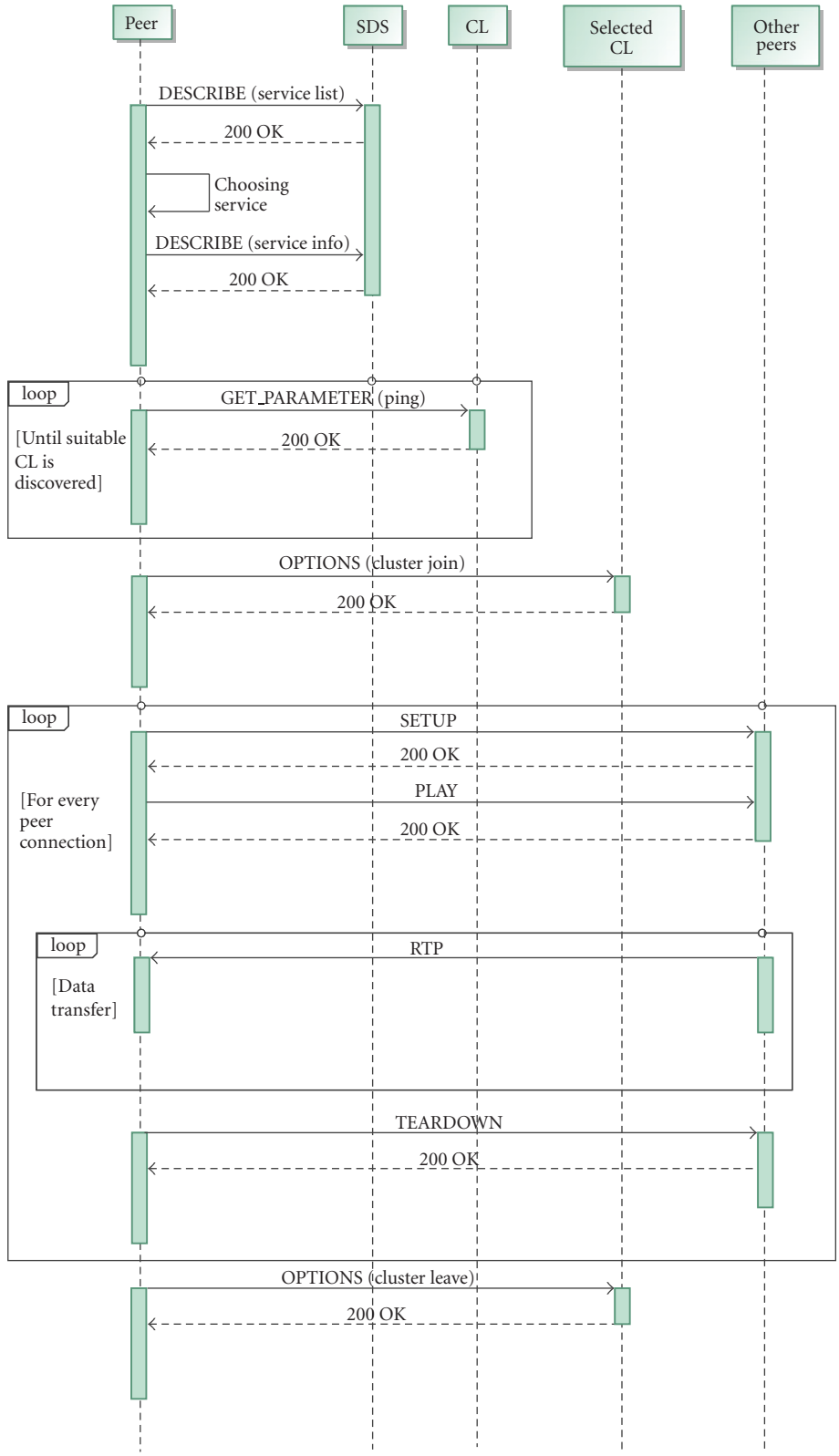


Figure 9: Peer in a service.

act as stream relay nodes, could be used to store all the data. Such relay nodes simply store the data and pass it to other network participants. It should be noted that a support node also consumes upload bandwidth from other peers, but with a high throughput network connection it provides more bandwidth to others than it consumes. The currently existing simple VoD implementation uses the simplest caching model, which is of course not the most optimal one, but it enabled a fast proof-of-concept implementation.

7. Peer Operation in a Service

Figure 9 shows the message exchange in case a peer is participating in a particular service. For getting a list of all available services a `DESCRIBE` message is sent to the SDS. The service list information obtained in the `200 OK` message contains only general information of the services to decrease the message size and the information is expressed as XML fragments. If a user wishes to search services with a wildcard string, a search element could be used to deliver the wildcard string to the SDS. If there are not available services, the SDS will respond with a `404 Not Found` message.

To be able to join to a particular service, more detailed service information must be retrieved from the SDS using a `DESCRIBE` message with an RTSP URL specifying the service. A `200 OK` message contains only a partial list of the available clusters, in case a large number of clusters has been created. The response uses multipart MIME [21], because it must deliver both the SDP of the service and the initial cluster list in XML format. If the service is not available anymore, a `204 No Content` message will be sent by the SDS.

After obtaining the CL list from the SDS, the peer makes contact with several CLs until a suitable CL is discovered. For this purpose, the peer sends a `GET_PARAMETER` message to the CL and starts the RTT counter. The peer stops the RTT counter when it receives a `200 OK` message and compares the RTT value to some predefined upper limit. The first CL with an RTT value below the limit is then selected as the CL for the service.

The peer joins the selected cluster by sending an `OPTIONS` message with a `join_peer` tag in the `Overlay` header field to the CL of the cluster. In the `200 OK` message, an initial peer list is received in XML format if some peers have already joined the cluster. Otherwise, a `Cluster-Id` header field is used to describe the ID for the cluster. The initial peer list is a random subset of the total peer set, if there are lots of peers in the cluster.

The peer tries to request data from other peers in the list order using a `SETUP` message. This message handles the configuration of the UDP port numbers for the RTP reception using the `Transport` header field. If there are fewer peers than the target number of partial streams, the peer continues requesting again from the beginning of the list, so that it will receive more than one partial stream from a single peer. If a certain peer is not responding, it will be removed from the internal *known peer list*, so that the peer does not try to reconnect again. The peer which is receiving the requested stream, that is, audio or video stream, will respond with a `200 OK` message to indicate that it might be possible to get the stream data from the peer.

Setting up of the partial RTP streams is done by sending a `PLAY` message to the peer which is receiving the requested stream. Splitting of the original RTP session into partial streams is explained in Section 5, and these `partial stream` parameters are signaled using a `Partial-Stream` header field. The format of a `Partial-Stream` header field in ABNF is given below:

```
Partial-Stream = "Partial-Stream:" SP partial-stream-
                 info
                 CRLF

partial-stream-info = id ";" piece-size ";"
                      nb-of-partials ";"
id = "id=" 1*DIGIT
piece-size = "piece-size=" 1*DIGIT
nb-of-partials = "nb-of-partials=" 1*DIGIT
```

If the peer is able to send the requested partials, it will respond with a `200 OK` message. If the peer noticed a loop, it will response with a `400 Bad Request` message, and if the peer cannot send the requested partials, it will respond with a `404 Not Found` message. After the `200 OK` message, data delivery from the requested peer using RTP is started. If the interval between two consecutive RTP packets is more than the predefined maximum allowed delay, the receiving peer should conclude that the sending peer is not capable of delivering the data in time and it should change the sender for the partial in question.

A peer can depart from the network in two ways. In a controlled departure, the peer informs its neighbors and the CL that it is leaving the network. The peer sends an `OPTIONS` message with a `leave`, tag in the `Overlay` header field to the CL, and a `TEARDOWN` message to all of its data delivery neighbors. Neighbors, which were sending data know that they can terminate the RTP session. Also peers that were receiving data know that they will not be able to receive more data from that peer, and can search for a replacement. The `TEARDOWN` message will also be sent when a peer notices that there is a loop in the data delivery for some partial stream.

An uncontrolled peer departure is noticed both by the CL and a peer which sends data to the departed peer after connection keep alive messages, that is, `GET_PARAMETER` messages, have not been received within some time interval X. Currently, keep alive messages are sent at 30 seconds intervals towards the CL and at 15 seconds intervals towards the peer which sends the data. If the sending peer does not receive keep alive message within 30 seconds interval it concludes that the receiving peer has departed and terminates the RTP session. Similarly, if the CL does not receive any message from the peer within 45 seconds interval it concludes that the receiving peer has departed and removes the peer from the cluster because of inactivity. A peer which is receiving data from the departed peer will notice uncontrolled departure after it has not received any RTP packets since Y seconds. The value Y should be defined so that it is possible to get data from a replacement peer within the reception buffer duration in order to avoid interruption. This is basically the same situation when the sending peer

is not capable of delivering the data in time and currently Y is calculated according to (2). This value consists of time between two pieces belonging to the same partial RTP stream, the normal network delay T_N, that can be calculated from RTSP request-response pairs, and a small extra time T_E given to peers to patch packets that might still be forwarded in the network:

$$Y = T_P * (N - 1) + T_N + T_E. \quad (2)$$

To request data from a replacement peer from a certain starting point, a `Packet-Range` header field can be included into a `PLAY` message to signal the play-after value using RTP sequence numbers. The `Packet-Range` header field can also be used to signal the current playback position when the peers are seeking a new playback position in a VoD service. The format of a `Packet-Range` header field in ABNF is given below. The two different use cases can be distinguished by the minus sign used in the former case:

```
Packet-Range = "Packet-Range:" SP range-specifier CRLF
range-specifier = 1*DIGIT ["-"]
```

In the VoD service, two other additional header fields are required for the seeking operation. The desired seek time in milliseconds will be signaled using a `Seek` header field. A `Fast-Send` header field will be used to inform the sender to send a specified amount of data (in milliseconds) as fast as possible to be able to fill up the reception buffer and start playback with as small delay as possible. The formats of the `Seek` and a `Fast-Send` header fields in ABNF are given below:

```
Seek = "Seek:" SP seek-time CRLF
seek-time = 1*DIGIT

Fast-Send = "Fast-Send:" SP fast-send-time CRLF
fast-send-time = 1*DIGIT
```

8. Implementation

The architecture of the real-time P2P streaming (RTP2P) application is presented in Figure 10. The RTP2P application is implemented using C++ and it consist of ten different software components. The principle in the figure is that the higher layer uses all components that are immediately below it, so, for example, the Graphical User Interface (GUI) uses Service, Common, and Media Player components.

The GUI is implemented using the gtkmm framework [22] for the Linux desktop environment and the maemomm framework [23] for the Nokia N800 Internet Tablet. Currently three different media players, VLC [24], MPlayer [25], and GStreamer [26], are needed in the application. This is necessary because any single player cannot offer all the features required for our application. VLC is used to stream RTP packets locally to the RTP2P application in the case of the original data source. The original data source only listens to the local socket, and receives RTP packets generated by the VLC and can forward those further using the multisource streaming concept explained in Sections 5 and 6. MPlayer is used for the media playback on the client applications. It is also possible to use VLC for the media playback, but with MPlayer it is possible to achieve a better synchronization with multiple elementary streams by using RTCP sender reports. GStreamer is used to create an RTP stream from the camera of the N800 device. The needed RTP operations are provided by the GNU ccRTP library [27]. In addition, the Boost libraries [28] are utilized for threading time and file system operations.

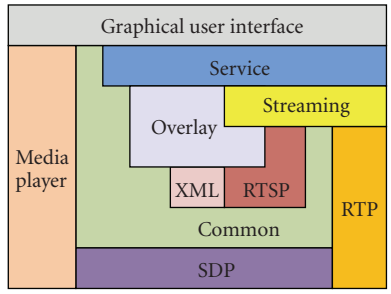


Figure 10: Architecture of the RTP2P streaming application

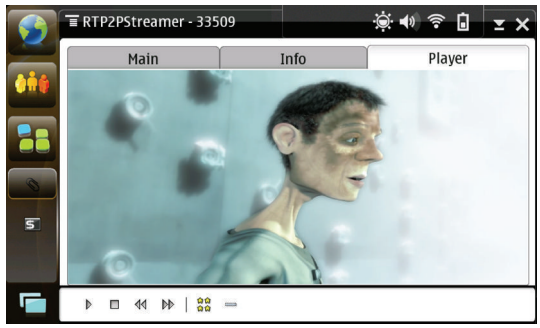


Figure 11: Graphical user interface.

Other proprietary software components, that is, Service, Overlay, Streaming, XML, RTSP, Common, and SDP, form the basis for the peer operation. The SDP component is based on the GNU oSIP library [29] and is used to parse the streaming service description expressed in the SDP format. The Common component contains definitions and functionalities which are widely used by other components including, for example, threading and socket operations. The XML component is based on the Expat XML parser [30] and it is utilized for parsing cluster and service information expressed in the XML format. The RTSP component contains RTSP message creation and parsing and also the base functionality for RTSP operations, which are enhanced and utilized by Streaming and Overlay components. The Streaming component includes sender and receiver functionalities to handle P2P RTSP communications and RTP reception and sending operations for the streaming service. The functionality of the cluster leader and all RTSP overlay communication are included in the Overlay component. The Service component contains the functionality needed to join, create, and manage streaming services.

Figure 11 presents the GUI in a Nokia N800 device. The GUI consist of three parts: (a) the main application view

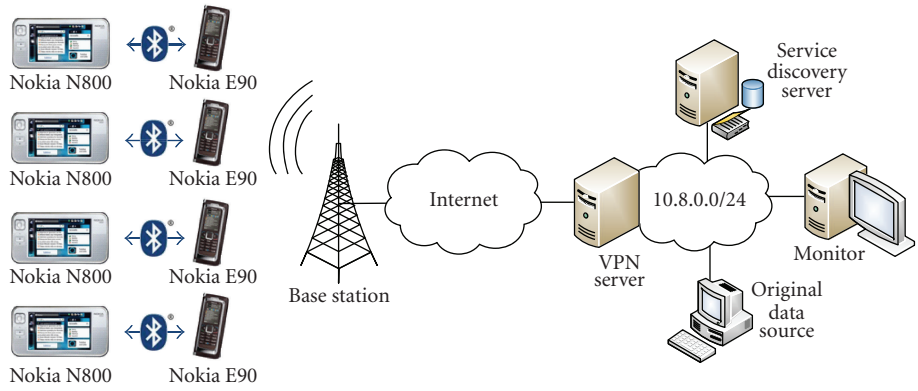


Figure 12: Test setup for evaluating the operation of the system in the mobile environment.

with Main, Info and Player tabs, (b) Create, Connect, List and Preferences dialogs launched from the drop down menu, and (c) Toolbar buttons at the bottom to start and stop the selected service at the receiving end, and the service in the original data source. The List dialog is used to obtain the service list from the SDS which is then shown in the Main tab. Information about the sourced service is also shown in the Main tab. The Info tab is used to show detailed service information about the sourced service and the service that is currently received by the peer. The Player tab shows the sourced stream in the original data source and the currently received stream by the peer. The Connect dialog is used to enter an IP address or a domain name for connecting to the SDS. The Create dialog is used for creating new services. Service name, type and description are given by the user, as well as the file to be sourced in case of stored content. The content can also be captured directly from the camera. The Preferences dialog can be used to adjust some attributes, for example, port numbers for the network traffic and the level of debug messaging.

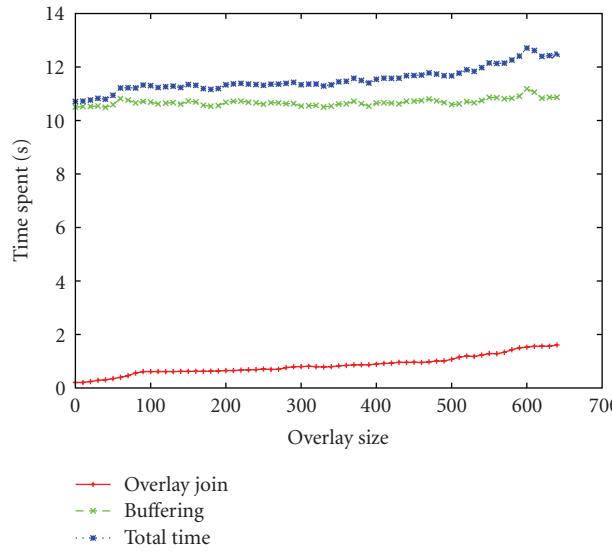


Figure 13: Overlay joining time and an initial buffering time as a function of overlay size; maximum cluster size 30 peers.

9. Performance Evaluation

The test setup for evaluating the operation of the system in the mobile environment is presented in Figure 12. Four Nokia N800 Internet Tablets with HSDPA network connection provided by the Nokia E90 were used together with a PC acting as an SDS. In some test cases, also another PC was acting as an original data source. During the tests a special application was used to monitor streaming connections between peers. As most normal consumer connections, also mobile connections are suffering from Network Address Translation (NAT) based connection limitations because Internet Service Providers (ISPs) do not want that users use the relatively small mobile bandwidth for hosting services or using P2P technologies. To avoid connection limitations, a workaround with VPN solution was utilized. Every entity with a restricted network connection first creates a connection to the VPN server and gets the desired free connectivity through the VPN tunnel. A more permanent solution could be implemented using NAT traversal to overcome these restrictions.

Tests have shown that the system performs well over mobile connections with ten seconds initial buffering time. The buffer size in our application is also ten seconds due to the RTP usage and is relatively small compared to the buffer sizes of existing solutions reported in [4].

Because the amount of available mobile devices was limited, a laboratory network environment with 17 desktop PCs (Intel Core2Duo E6550, 4 GB DDR2, running CentOS Linux, kernel 2.6.18 PAE) has been utilized to test the system functionality with a larger amount of peers. 16 hosts were used to run 40 peers in each host together with one host acting both as an SDS and as an original data source. The connectivity between all devices was provided by a 1 Gbps switch. The length for one live streaming service was roughly one hour and all forthcoming figures present average values from five different live streaming services. The maximum cluster size was set either to 30 or to 70 peers, and peers were started in 40 cycles with a 5 s starting interval: first one peer was started at each host, then a second one and so on.

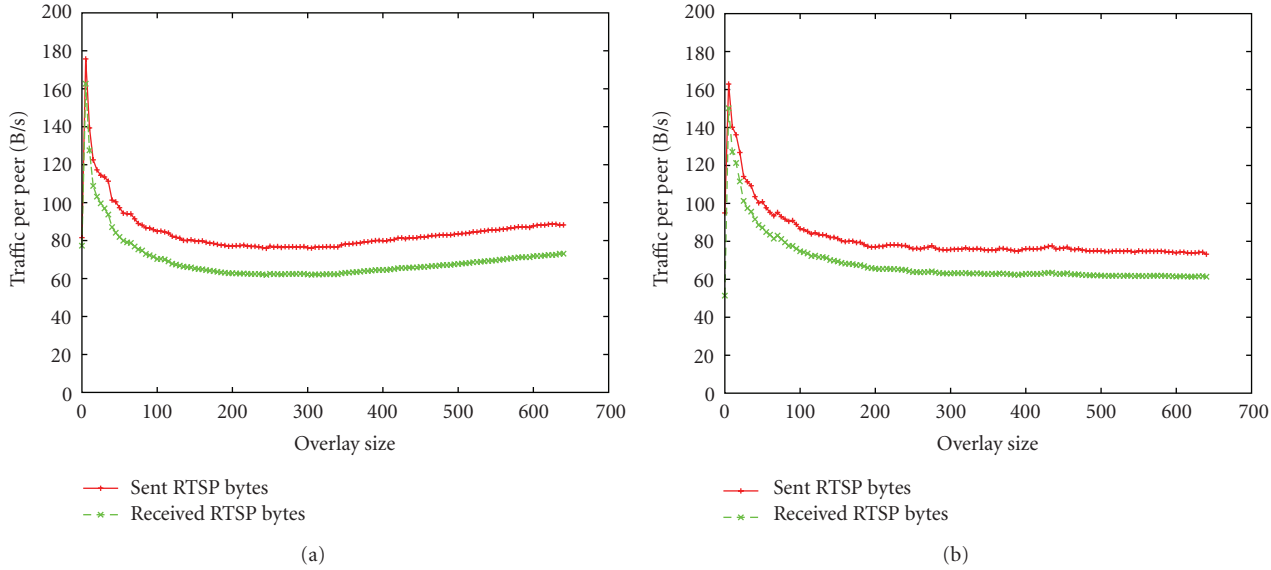


Figure 14: The amount of sent and received RTSP data, (a) maximum cluster size 30 peers, and (b) maximum cluster size 70 peers.

9.1. Steady Network. In this test scenario all peers stayed steadily in the service from the joining time to the end of the service. The streaming service joining time, that is, the overlay joining time plus the initial buffering time, as a function of the overlay size is presented in Figure 13. From the figure we can see that the initial buffering time remains almost constant regardless of the number of peers in the network. The increase in overlay joining time could be minimized by improving the current cluster selection algorithm.

The amount of sent and received RTSP data in bytes per peer as a function of overlay size is shown in Figure 14. The combined bit rate of the original RTP sessions is about 112 kbps, encoded using FFmpeg's [31] H.263+ video, and AAC audio codecs. Hence, the RTSP signalling overhead is quite minimal compared to the actual media data. The reason for the larger amount of RSTP data sent is caused by the relatively large cluster status information messages transmitted to the SDS by the CLs. The main difference between Figure 14(a) and Figure 14(b) is the slight increase in the RTSP signalling data after 300 peers with the maximum cluster size of 30 peers. This is caused by the larger (and increasing) amount of clusters and cluster status information messages. Similar effect might take place also with the maximum cluster size of 70 peers when the amount of peers is increased above the currently used maximum value.

9.2. Network with Leaving and Rejoining Peers. To simulate even a more realistic situation and the churning caused by the real mobile nodes, a timer functionality that is able to randomly shut down and restart nodes was used. After a peer had joined to the service, it stayed randomly from 30 s to 10 min in the service and then left the service and joined back after 10 s. This allowed us to test the network in more realistic situations where peers are leaving and data connections are failing.

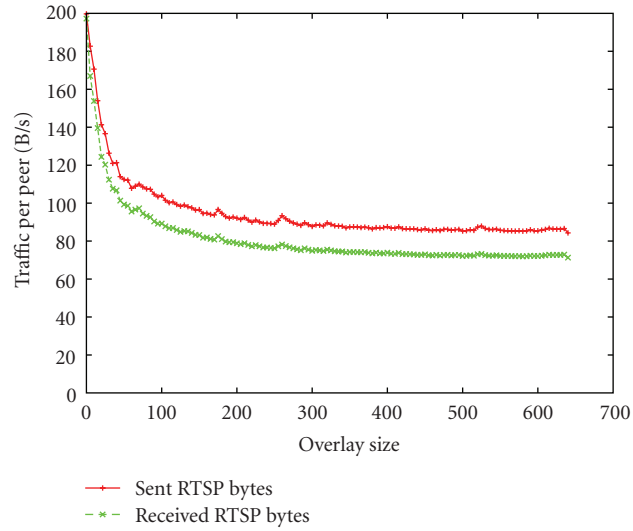


Figure 15: The amount of sent and received RTSP data with leaving and rejoining peers; maximum cluster size 70 peers

The amount of sent and received RTSP data in bytes per peer as a function of the overlay size is presented in Figure 15. If we compare these values to the corresponding values in the steady state scenario, we can see that the values follow the same trend, but the replacement peer searching and peer rejoining messages cause small overall increase to the signalling data.

10. Future Developments

In the current overlay implementation, a cluster change is controlled by the CL. Peer-controlled cluster change, where a peer changes the cluster after acquiring knowledge about a new cluster which would serve its data requirements better,

would make the system more scalable and will increase the overall performance.

Clusters are currently loosely connected together between the neighboring CLs to share some peer information. A clearer cluster group structure with a Cluster Group Leader (CGL) is worth studying. The CGL could collect the information about all clusters within the cluster group and send cluster update messages to the SDS, instead of separate update messages from individual CLs. This organization into an n-level hierarchical structure could increase network scalability and reduce the overlay joining time since the cluster search time would be reduced from $O(n)$ to $O(\log(n))$. As a drawback, the complexity of the system and the overlay maintenance will be highly increased.

More advanced implementation level support for VoD streaming such as better caching mechanism and support for other Video Cassette Recording (VCR) functionalities like fast-forward and rewind, in addition to the currently existing seek functionality will definitely pose different requirements compared to the live streaming case. Mechanisms for handling packet losses is an important research area in peer-to-peer streaming. Different error robustness techniques, such as simple retransmission, Forward Error Correction (FEC) and network coding, need to be studied to find out the benefits and drawbacks of those techniques, when used in addition to the current mechanism which is based on peer replacement before the reception buffer underflows.

One interesting research area is the usage of Multiple Description Coding (MDC) [32] or Scalable Video Coding (SVC) [33] in the real-time P2P streaming as is proposed also in [10]. A single stream is divided into several descriptions and each of the descriptions is then forwarded separately to the network. With this approach, the current partial stream could be replaced by one description without affecting the clustered overlay network architecture. However, our partial stream concept has much lower complexity (than MDC or SVC), which has enabled a fast proof-of-concept implementation. Our design allows an easy replacement of the partial stream concept with MDC or SVC as soon as their implementations become publicly available.

11. Conclusions

The effective real-time P2P streaming system for the mobile environment presented in this paper is an alternative solution to traditional client-server-based streaming applications. A scalable overlay network which groups peers into clusters according to their proximity is created and maintained using extended RTSP messages by the cluster leaders with the help of a service discovery server. Furthermore, the actual media delivery is implemented using a partial RTP stream concept. RTP sessions are split into a number of partial streams in such a way that it allows reassembling the original media session in real-time at the receiving end.

The first laboratory tests together with the tests in the mobile environment have shown that the current implementation performs well and offers very low initial buffering times. More advanced laboratory tests with different latencies and throughputs between peers are still needed to highlight system bottlenecks and usability issues.

Acknowledgments

The authors would like to thank Joep van Gassel, Alex Jantunen and Marko Saukko for their valuable work as part of the development team. This work was partially supported by TEKES as part of the Future Internet program of Finnish Strategic Centre for Science, Technology and Innovation in the field of ICT (TIVIT).

References

[1] "YouTube—Broadcast Yourself," May 2009, http://www.youtube.com/.

[2] "Octoshape," May 2009, http://www.octoshape.com/.

[3] "SopCast," May 2009, http://www.sopcast.org/.

[4] J. Peltotalo, J. Harju, A. Jantunen, et al., "Peer-to-peer streaming technology survey," in *Proceedings of the 7th International Conference on Networking (ICN '08)*, pp. 342–350, April 2008.

[5] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," Internet Engineering Task Force, RFC 3550, July 2003, http://www.rfc-editor.org/rfc/rfc3550.txt.

[6] B. Cohen, "Incentives build robustness in BitTorrent," in *Proceedings of the Workshop on Economics of Peer-to-Peer Systems (P2PECON '03)*, pp. 116–121, June 2003.

[7] P. Shah and J.-F. Paris, "Peer-to-peer multimedia streaming using BitTorrent," in *Proceedings of the 26th IEEE International Performance, Computing, and Communications Conference (IPCC '07)*, pp. 340–347, April 2007.

[8] X. Jiang, Y. Dong, D. Xu, and B. Bhargava, "GnuStream: a P2P media streaming system prototype," in *Proceedings of the International Conference on Multimedia and Expo (ICME '03)*, pp. 325–328, July 2003.

[9] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming," in *Proceeding of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)*, vol. 3, pp. 2102–2111, March 2005.

[10] N. Magharei and R. Rejaie, "PRIME: peer-to-peer receiver-drIven MEsh-based streaming," *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM '07)*, pp. 1415–1423, May 2007.

[11] D. A. Tran, K. A. Hua, and T. Do, "ZIGZAG: an efficient peer-to-peer scheme for media streaming," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '03)*, vol. 2, pp. 1283–1292, March 2003.

[12] J. Liang and K. Nahrstedt, "DagStream: locality aware and failure resilient peer-to-peer streaming," in *Proceedings of the 13th Annual Multimedia Computing and Networking Conference (MMCN '06)*, pp. 224–238, January 2006.

[13] J. Yu and M. Li, "CBT: a proximity-aware peer clustering system in large-scale BitTorrent-like peer-to-peer networks," *Computer Communications*, vol. 31, no. 3, pp. 591–602, 2008.

[14] J. Peltotalo, J. Harju, M. Saukko, et al., "A real-time peer-to-peer streaming system for mobile networking environment," in *Proceedings of the INFOCOM and Workshop on Mobile Video Delivery (MoVID '09)*, April 2009.

[15] H. Schulzrinne, A. Rao, and R. Lanphier, "Real Time Streaming Protocol (RTSP)," Internet Engineering Task Force, RFC 2326, April 1998, http://www.rfc-editor.org/rfc/rfc2326.txt.
[16] D. Crocker and P. Overell, "Augmented BNF for Syntax Specifications: ABNF," Internet Engineering Task Force, RFC 2234, November 1997, http://www.rfc-editor.org/rfc/rfc2234.txt.
[17] T. Berners-Lee, R. Fielding, and L. Masinter, ""Uniform Resource Identifier (URI): Generic Syntax," Internet Engineering Task Force, RFC 3986, January 2005, http://www.rfc-editor.org/rfc/rfc3986.txt.
[18] W3C, *Extensible Markup Language (XML) 1.0*, World Wide Web Consortium (W3C), 4th edition, 2006.
[19] "zlib," May 2009, http://zlib.net/.
[20] M. Handley and V. Jacobson, "SDP: Session Description Protocol," Internet Engineering Task Force, RFC 2327, April 1998, http://www.rfc-editor.org/rfc/rfc2327.txt.
[21] K. Moore, "MIME (Multipurpose Internet Mail Extensions) Part Two: Message Header Extensions for Non-ASCII Text," Internet Engineering Task Force, RFC 1522, Sepember 1993, http://www.rfc-editor.org/rfc/rfc1522.txt.
[22] "gtkmm—C++ Interfaces for GTK+ and GNOME," May 2009, http://www.gtkmm.org/.
[23] "maemomm—C++ bindings for the Maemo API," May 2009, http://maemomm.garage.maemo.org/docs/index.html.
[24] "VLC Media Player," May 2009, http://www.videolan.org/vlc/.
[25] "MPlayer—The Movie Player," May 2009, http://www.mplayerhq.hu/.
[26] "GStreamer: open source multimedia framework," May 2009, http://www.gstreamer.net/.
[27] "GNU ccRTP—GNU Telephony," May 2009, http://www.gnu.org/software/ccrtp/.
[28] "Boost C++ Libraries," May 2009, http://www.boost.org/.
[29] "The GNU oSIP Library," May 2009, http://www.gnu.org/software/osip/osip.html.
[30] "The Expat XML Parser," May 2009, http://expat.sourceforge.net/.
[31] "FFmpeg," May 2009, http://www.ffmpeg.org/.
[32] V. K. Goyal, "Multiple description coding: compression meets the network," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 74–93, 2001.
[33] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, 2007.

Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 843574, 12 pages
doi:10.1155/2010/843574

Research Article

Localized Multistreams for P2P Streaming

Majed Alhaisoni,[1] Mohammed Ghanbari,[1] and Antonio Liotta[2]

[1] *School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK*
[2] *Electrical Engineering Department and Mathematics and Computer Science Department, Technische Universiteit Eindhoven, P.O. Box 513, PT 11.29 5600 MB Eindhoven, The Netherlands*

Correspondence should be addressed to Majed Alhaisoni, malhai@essex.ac.uk

Received 7 June 2009; Revised 26 November 2009; Accepted 3 February 2010

Academic Editor: Stan Moyer

Streaming video over the Internet, including cellular networks, has now become a commonplace. Network operators typically use multicasting or variants of multiple unicasting to deliver streams to the user terminal in a controlled fashion. P2P streaming is an emerging alternative, which is theoretically more scalable but suffers from other issues arising from the dynamic nature of the system. Users' terminals become streaming nodes but they are not constantly connected. Another issue is that they are based on logical overlays, which are not optimized for the physical underlay infrastructure. An important proposition is to find effective ways to increase the resilience of the overlay whilst at the same time not conflicting with the network. In this article we look at the combination of two techniques, redundant streaming and locality awareness, in the context of both live and video-on-demand streaming. We introduce a new technique and assess it via a comparative, simulation-based study. We find that redundancy affects network utilization only marginally if traffic is kept at the edges via localization techniques.

1. Introduction

Overlays and P2P (peer-to-peer) systems, that have originally been developed as alternatives to IP multicast and to provide file sharing services, now have moved beyond that functionality. These technologies have deeply improved the distribution of information on the Internet by enabling resourceful cooperation among end consumers. With the growing bandwidth capacity provided by the Internet, they are also proving to be key technologies for the delivery of real-time video and of video-on-demand content.

By the cooperation among peers in helping each other in the network, P2P technology overcomes various limitations of the more conventional client-server paradigm to attain user and bandwidth scalabilities. In a P2P streaming application, multimedia contents are delivered to a large group of distributed users with low delay, high quality and high robustness [1]. P2P-based versions of IPTV, Video on Demand (VoD), and conferencing are thus becoming popular.

Many hosts can be supported by a P2P multimedia system, possibly in excess of hundreds or even millions, with miscellaneous heterogeneity in bandwidth, capability, storage, network, and mobility. Another aim is to maintain the stream even under dynamic user churn, frequent host failures, unpredictable user behaviors, network traffic, and congestion. To accomplish these goals, it is imperative to address various challenges to achieve effective content delivery mechanisms, including routing and transport support.

In this article, we are primarily concerned with finding effective ways of increasing the resilience and scalability of the overlay whilst at the same time minimizing the impact on the physical network (or underlay). We find that P2P frameworks mostly fail to address the latter issue and, in doing so, they tend to cause severe network operational and management issues. In turn, this limits P2P scalability when traffic streams traverse, and thus congest large portions of the network. Another issue is that existing P2P streaming systems are intrinsically best-effort. This fact, combined with their network unfriendly behavior, often leads the operator to impair P2P traffic, with detrimental consequences for the resulting quality of service.

The key question we are addressing herein is whether and how it would be possible to increase the user quality of experience (QoE) in P2P streaming. A common technique is to increase redundancy, that is, send multiple streams to the same user in order to reduce packet loss. The downside is that

redundancy increases traffic load, thus increasing congestion and hence reducing network utilization.

In order to retain the benefits of redundancy (QoE) and reduce its detrimental effects on the network, we study the combination of two techniques, multistreaming and network locality. We find that by keeping traffic local among the peers and mainly at the edges of the network, the benefits of multistreaming outweigh their shortcomings. We carry out a comparative evaluation using a popular P2P TV system, Joost, as the benchmark. Initial results indicate that QoE is significantly improved at a little cost for the network.

2. Related Work

As we are dealing with network QoS, QoE, P2P locality awareness, and stream redundancy in this paper, we give an overview of different studies that have looked at these topics individually.

Ways to pursue efficiency between overlay and underlay have started to be investigated only recently. Authors in [2] propose a technique, where the peers on the overlay are chosen based on their mutual physical proximity, in order to keep traffic as localized as possible. A similar approach is described in [3], where they measure the latency distance between the nodes and appropriate Internet servers called landmarks. A rough estimation of awareness among the nodes is obtained to cluster them altogether, as in [4, 5].

On the other hand, another study in [6] proposes different techniques where the video stream is divided into different flows that are transmitted separately to increase parallelism and, hence, reduce transmission latency. The authors use the PSQA technique that gives an estimate of the quality perceived by the user. This study was concerned on how to influence and improve on quality (as measured by PSQA). They introduce three cases: sending a single stream between nodes, sending two duplicate streams via different paths, and sending two disjoint substreams whose union recreates the original one. In our work we look at the case of multiple redundant streams, looking at the effects that redundant streams have on both the network load and the user QoE. Also we emphasize on techniques to choose intercommunicating peers based on their mutual proximity, to keep traffic local and minimize the impact on the network load.

Overlay locality is also studied by [7], where the authors make use of network-layer information (e.g., low latency, low number of hops, and high bandwidth). We use though a different distance metric, based on RTT (round trip time) estimations, to prioritize overlay transmissions. Additionally, we use a cluster management algorithm whereby intercommunicating peers are forced to periodically handover, in order to distribute computational load as well as network efficiency (as explained in [8, 9]).

Hefeeda et al. [10] have proposed a mechanism for P2P media streaming using Collectcast. Their work was based on downloading from different peers. They compare topology-aware- and end-to-end selection based-approaches.

The latter approach is also the subject of [11], which employs a simpler version of our RTT approach based on continuous pinging of peers. Similarly, we adopt clustering to limit the signaling overheads associated with this process and prevent bottlenecks.

Other studies such as [12] propose relevant methods to serve multiple clients based on utility functions or clustering. A dynamic overlay capable of operating over large physical networks is presented in [13, 14]. In particular, they show how to maximize the throughput in divisible load applications.

Moreover,Locher et al. [15] proposed a distributed hash table which is suitable for high dynamic environment. Their work was designed to maintain fast lookup in terms of low delay and number of routing hops. In their work, number of hops was the main metric which used to determine locality-awareness. According to their work, neighboring nodes are grouped together to form a clique. Nodes share the same ID in a clique; moreover, the data will be replicated on all the nodes on the clique to avoid data loss.

Additionally, a clique has an upper and lower bound in terms of the number of nodes, such that cliques are forced to merge or split. Another aspect of their work is to assume that all the nodes are distributed uniformly in a two-dimensional Euclidean space. However, this may not work in a large network such as the internet. In addition, the link structure is updated periodically in order to establish a structured network. On the other hand, their proposal is based on pining nodes to join the closet clique which will drastically introduce extra signaling overhead.

Another study similar to [15] was conducted by Asaduzzaman et al. [16]; their proposal was built on top of [15], with some modifications by introducing stable nodes (supernode) and replicating the data among the stable nodes only. However, their proposal elects one or more stable nodes of highest available bandwidth in each cluster and assigns special relaying role to them.

Their work is based on a combination of tree and mesh architectures where the nodes on the clique form a mesh and the stable nodes are connected in a tree structure.

For each channel, a tree based is formed between the stable nodes including only one stable node in each clique. However, stable nodes are elected based on their live session. So, in this case a clique may have more than a stable node. The downside to this approach is that the relaying nodes (super nodes) are forming a tree; so reconstructing them in case of failures and peers churn will be costly and can introduce some latency.

By contrast to the abovementioned two works, our proposal aims not only to retain the benefits of redundancy (QoE) but also to reduce its detrimental side effects on the network. We study the combination of two techniques, multistreaming and network locality, while in [15, 16] they are mainly concerned with network locality. We prioritize the choice of sources based on their mutual distance from the destinations. In essence we adopt a previously published hierarchical RTT monitoring approach [17] to maintain a list of sources $\{S_i\}$, ranking their order based on their distances from the recipient (R). Periodically, a new set of sources $\{S_{inew}\}$ is chosen from this pot and handover is forced from $\{S_i\}$ sources to $\{S_{inew}\}$ sources.

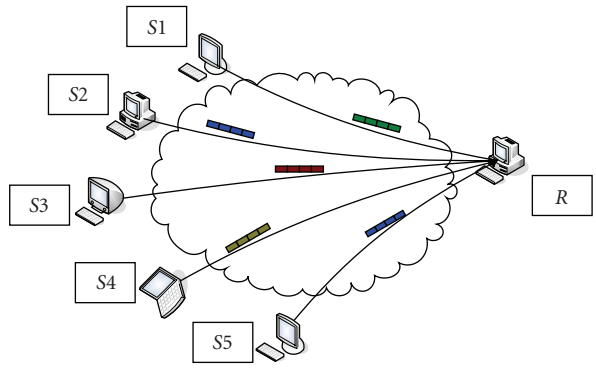


Figure 1: Proposed Architecture.

The hypothesis here is that this forced handover strategy does not impact network congestion if traffic is kept away from the core network. We aim to establish, however, until which point we can increase redundancy without triggering network congestion. Moreover, our work not only introduces locality-awareness and multiple-streams but also loads balances computing and network resources. Additionally, we introduce a new way of calculating the packet loss ratio, end-to-end delay, network utilization with upper and lower bonds and receiver utilization with the stream redundancy.

Furthermore, QoS and QoE are tested out by transmitting a video to examine the scalability and resilience of this proposition. In an earlier publication [9], we have examined the locality-awareness and computational efficiency and compared them to some popular P2P streaming applications. On the other hand, from the locality-awareness viewpoint, our proposal is similar to [15, 16] but with a different aim and approach.

Looking at previous studies, we can say that our main contributions are:

(1) to study a new combination of existing techniques (cross-layer optimization, localization, forced handovers, and multistreaming),

(2) to take the perspective of the network operator, in trying to harmonize overlay and underlay networks,

(3) to look for trade-offs between redundancy levels (to increase QoE) and network efficiency.

3. Proposed Approach

3.1. Target Architecture. In our study the number of redundant (multi)streams varies from 1 to 5 as shown in Figure 1. Sources $\{S1 \cdots S5\}$ are chosen based on locality and are also continuously (periodically) forced to handover, choosing new sources from a pot of available sources. These are prioritized based on mutual interpeer distances to ensure traffic is kept as local as possible. On the other hand, forced handovers ensure that the important feature of computational load-balancing is maintained—we have discussed this particular issue in previous publications [8, 9].

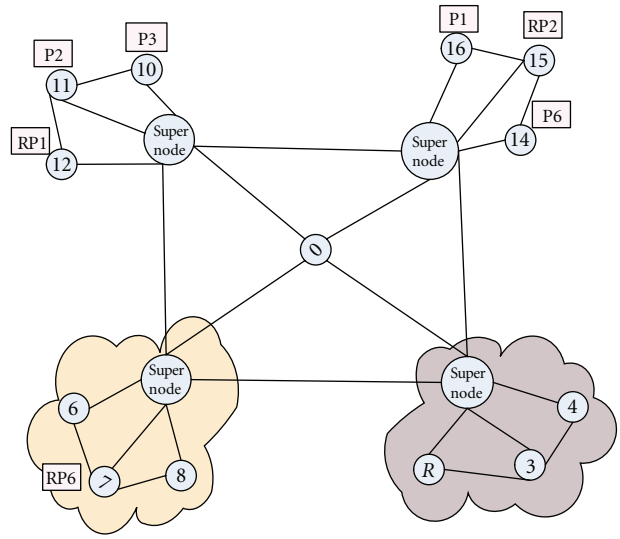
Instead, herein we are mainly interested in understanding whether location-aware P2P techniques can actually reduce the detrimental side effects of P2P redundant streaming. Under architectures other than P2P (unicast, multiple unicasts, and multicast), redundant streams reduce packet loss but have the side effect of increasing network congestion. Interestingly in P2P, redundant streams act as backup of each other and although more redundant streams increase congestion, they actually reduce packet loss. Thus an interesting question is what is the optimum redundancy level that can lead to maximum QoE improvement? By contrast if we adopt a P2P approach that succeeds in keeping traffic away from the core network, we have a better chance that redundancy does not directly result into network congestion. Our aim is to verify this hypothesis and better exploit its implications. However, before dealing with the redundancy issue, another important factor in this proposition is the locality-awareness; so next section is given a wider view of how this is introduced in the proposed study.

3.2. Locality-Awareness. Network efficiency (locality) is the ability to keep traffic as local as possible, which can be achieved by connecting to those peers which are nearby and changing the sources among the participants. Therefore, in the proposed method, a decision is made among the participant peers based on the measured RTT values by the monitoring system. Peers are prioritized on the order of lower RTT values, and the connections are setup based on these values. Consequently, this will not only maintain the network locality among the intercommunicating nodes but it will also improve the QoS and, hence, the user's quality of experience (QoE).

However, offering network locality only without changing the sources among the peers would be drastically impairing load balancing or, in other words, the load distribution between the network and the computing sources. Therefore, different techniques are embedded to the proposed method. The main aim of these techniques is to distribute the load among the participants and at the same time having the network locality not impaired. This can be shown in the next section.

3.3. Computing and Network Resources Load. In order to maintain the load balancing among the contributing peers, different handover techniques have been embedded into the proposed approach. Two conditions trigger the handover among the interconnected peers.

Switching Over. Since the network may experience various constraints such as congestion, bottleneck, and link failures, the RTT values will be severely affected and may not be reliable. Additionally, these stochastic conditions will drastically affect the network locality and degrade the quality of service (QoS) parameters such as throughput, packet loss, and end-to-end delay. There is also another important requirement arising directly from the adoption of P2P: peers are not reliable entities and cannot be assumed to be always connected. Nodes may leave and join at unpredictable times. So, we must adopt a mechanism which allows the receiving peers (in client mode) to maintain a continuing reception of



P: Packet
RP: Redundant packet
R: Receiver
CBR: Source
Sink: Destination

Figure 2: Simulation Topology.

video, although the streaming peers (in server mode) are not constantly available.

One solution to this requirement is that any intending client should regularly update the neighbor's list and reorder them based on the lower RTT values. In our implementation, we keep a ranking list of the peers based on their RTT values. Each peer streams from $\{S1 \cdots S5\}$ other peers. At the switching over time, a new set of the top peers are chosen (those with lower RTT values to the peer under consideration).

Enforced Handover. Another favorable property in the proposed method is its computational efficiency. This can be achieved when the load is periodically distributed among the peers. Under normal network conditions, peers with lower RTT are selected; but when link latency changes, switch over is applied and new peers having lower RTT values are selected.

Some peers may not experience any constraints such as congestion, bottleneck, and link failures. The RTT values will not be affected and may not be changed; so those peers may become the best in every periodical check. Therefore, selecting them regularly would impair computational load balancing among the peers. To avoid this condition, enforced handover is applied.

Furthermore, to avoid pure randomness on the enforced handover process, network locality is applied into clusters of peers, named superpeers, similar to the one adopted in KaZaA [18]. Thus, peers are grouped and they are managed by a special peer, or a super node (Figure 2 depicts a sample of the used topology). Our experiments have confirmed that peers on the same cluster share nearly the same RTT values.

3.4. Redundancy Principle. In order to study the effect of redundancy in relation to both QoS and QoE, we first measure relevant parameters (as detailed in Section 4) for a Joost-like system [19], which in practice chooses sources randomly and is used here as a benchmark. Redundancy is increased from 0 (1 source per destination) to 4 (5 sources per destination). We then compare this with our proposed approach, in which sources are forced to handover continuously (as in Joost)—to ensure computational load balancing—but are not chosen randomly.

We prioritize the choice of sources based on their mutual distance from the destination. In essence we adopt a previously published hierarchical RTT monitoring approach [17] to maintain a list of sources $\{\mathbf{S_i}\}$, ranked based on their round trip delay distances from the recipient ($\mathbf{R}$). Periodically, a new set of sources $\{\mathbf{S_{inew}}\}$ are chosen from this pot and handover is forced from $\{\mathbf{S_i}\}$ sources to $\{\mathbf{S_{inew}}\}$ sources. Our hypothesis is that this forced handover strategy does not have ill effects on network congestion if traffic is kept away from the core network based on locality. We aim to determine, however, until which point we can increase redundancy without triggering network congestion.

4. Assessment Method

We study the effects of redundancy on QoS and QoE, for each of the two cases under scrutiny: (1) randomized scenario (new sources are chosen randomly) and (2) localized scenario (new sources are chosen based on minimal mutual distances (locality) from the recipient). We add background traffic into the simulated network in order to simulate network performance under congestion. Simulation and design parameters are described below.

4.1. Simulation Setup. The proposed approach was implemented and tested on the ns-2 network simulator (http://isi.edu/nsnam/ns/). A sample of the used topology is shown in Figure 2. Although the used topology is fixed in the simulation, the two scenarios are treated in a different way. So, for the randomized scenario, senders and receivers are chosen randomly for every run. On the localized (proposed) scenario, the senders are chosen based on locality and the receivers are selected randomly for every run. This gives the advantage of testing the localized scenario under different conditions over the used topology.

Moreover, various parameters were set on the used topology. First of all, each link has a bandwidth of 2 Mbps with equal length (delay). However, the actual delay will be according to the nodes distance of each other; so, all the participants' peers have the same characteristics. IP as the network protocol and UDP as the transport protocol have been chosen. For simulation of video traffic, the "Paris" video clip of CIF resolution with 4:2:0 format was H.264/AVC coded and the video packets were sent from one and multiple peers to the receiver.

Secondly, in order to overload the network, it was imperative to set the CBR background traffic to vary the network load and enable us to study the localized multistream approach under different loading conditions. The CBR traffic was setup from different sources to different destinations, with a 512 byte packet size. This background traffic operates during the whole duration of the simulations. This was set to 1 Mbps for the moderately congested network and 1.7 Mbps for the heavily congested network; this is added to the stream video on the running simulation.

The localized and randomized approaches were simulated independently and repeated 10 times. The presented results correspond to the average values of these simulations.

4.2. Evaluation Metrics

Packet Loss Ratio. Packet loss ratio is usually defined as the ratio of the dropped over the transmitted data packets. It gives an account of efficiency and of the ability of the network to discover routes. However, in P2P communication, a new way of calculating the packet loss ratio needs to be defined to study the particular issues relating to redundancy. In our case a packet is actually transmitted by several sources and is considered lost only if it is never received through any of the streams. This is formalized as follows:

$$\begin{aligned} &P_i: \quad \text{generic packet } (i) \text{ sent by all source nodes,} \\ &d_j: \quad \text{sending or source nodes,} \\ &X_{ij} = \begin{cases} 1, & \text{if } P_i \text{ sent by } d_j \text{ is lost,} \\ 0, & P_i \text{ is received.} \end{cases} \end{aligned} \tag{1}$$

The decision as to whether a packet is lost or not will be according to the following Cartesian product:

$$\mathrm{PL}_i = \prod_{j=1}^{d} x_{ij}. \tag{2}$$

Therefore, if P is the total number of packets required to reconstruct a given stream, the packet loss ratio will be

$$\mathrm{PL} = \frac{\sum_{i=1}^{p} \mathrm{PL}_i}{P}. \tag{3}$$

Receiver Usage. Receiver usage is defined as the ratio of received useful packets over the number of packets required (P) to reconstruct the video as

$$R_U = \frac{P - \mathrm{PL}}{P}, \tag{4}$$

where PL is the packet loss ratio, thus, this parameter increases with reduction on the packet loss ratio, and it can be an efficient tool of P2P performance analysis.

4.2.1. Network Utilization. In point-to-point communication (*i.e., Client-Server*), it is common that network utilization is defined as the ratio of received packets over the sent packets, but in P2P paradigm, when some of the sent or received packets are not useful, this definition is not helpful. Also, since lost packets from any source may not have any ill effect on the received quality as long as these packets can be received from other peers, thus, in the multistream, one can define an upper and lower bound to the network utilization, where the actual utilization stands somewhere in between.

Upper Bound. Upper bound is defined as the ratio of received useful packets over the total number of received packets from all the sending peers. That is,

$$U_b = \frac{P - \mathrm{PL}}{\sum_{i=0}^{n} R_i}, \tag{5}$$

where U_b represents the upper bound utilization, P is the required number of packets to reconstruct the video, and n is the number of sending peers. R_i is the total number of received packets from the ith peers and PL is the lost packets

Lower Bound. Lower bound is defined as the ratio of received useful packets over the total number of sent packets from all the peers. That is:

$$L_b = \frac{P - \mathrm{PL}}{\sum_{i=0}^{n} S_i}, \tag{6}$$

where L_b represents the lower bound utilization, P is the number of required packets to reconstruct the video, n is the number of sending peers, and S_i is total sent packets by the ith peer and PL is the lost packets.

The difference between the upper and lower bounds is in fact due to some stray packets of the senders that are lost in the network. Although these stray packets may not be useful to the receiver, as the duplicate packets from the other peers will replace them, nevertheless their presence in the network can congest the network. Thus the difference between the lower and upper utilization is an indication of the side effect that multistream may have on the network. The lower the

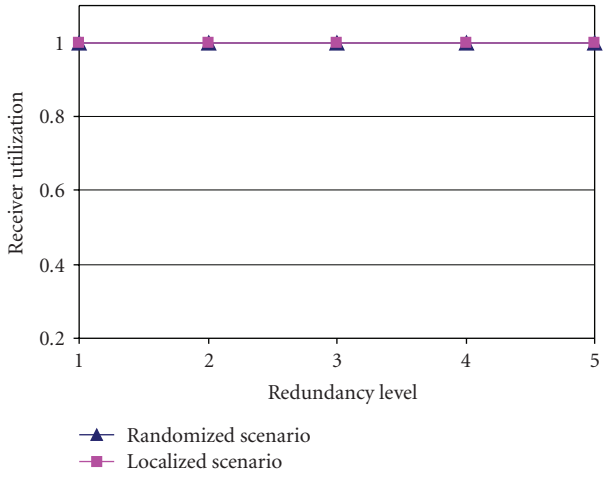


Figure 3: Receiver Usage—Moderately Loaded.

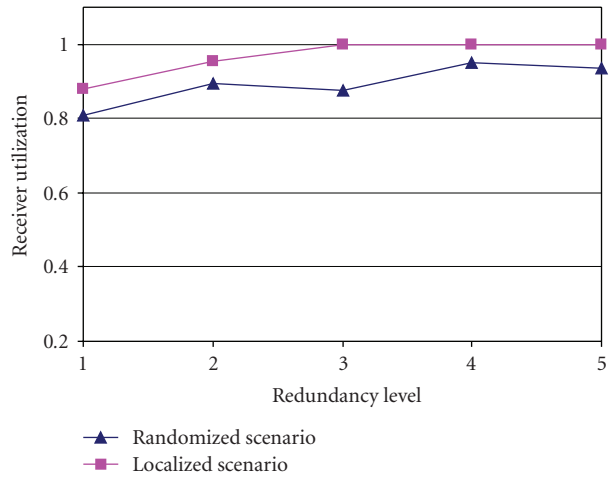


Figure 4: Receiver Usage—Heavily Loaded.

difference is, the better is the performance of the P2P stream. This difference, Nu is thus

$$Nu = \frac{P - \mathrm{PL}}{\sum_{i=0}^{n} R_i} - \frac{P - \mathrm{PL}}{\sum_{i=0}^{n} S_i}. \tag{7}$$

Average End-to-End Delay. Average end-to-end delay is average time span between transmission and arrival of data packets. In multistreams P2P, delay of each received packet is the minimum delay among all the packets of the same type sent by all senders. This includes all possible delays introduced by the intermediate nodes for processing and querying of data. End-to-end delay has a detrimental effect on real-time IPTV.

Peak Signal to Noise Ratio. PSNR is an objective quality measure of the received video, taken as the user QoE. It is defined as the logarithm of the peak signal power over the difference between the original and the received captured video:

$$\mathrm{PSNR} = 10 \log \frac{225^2}{E^2}. \tag{8}$$

5. Simulation Results

In order to assess the proposed scheme, different network parameters should be identified, tested, and evaluated. Therefore, the following parameters have been considered.

5.1. Receiver Utilization. Figures 3 and 4 show the receiver utilization, as defined in (4). For moderate network load our approach is almost similar to the benchmarking scheme which is based upon a random scenario. At higher network load shown in Figure 4, the relative superiority of localized over random connection is evident. The figure shows that at all redundancy levels, the localized connection has a better utilization. Moreover, for more than 3 levels of redundancy (receiving packets from more than 3 peers), the localized approach reaches 100% utilization. Beyond this value, increasing the number of sending peers does not increase receiver utilization. This figure also implies that 3 sending peers are sufficient enough to bring receiver utilization defined in (4) to 100% saturation level.

On the other hand, the randomized connection can never achieve the 100% utilization. Thus in this case there will always be packet losses, irrespective of how many sending peers are chosen, although increasing the number of sending peers improves receiver utilization and hence reduces the packet loss rate.

5.2. Network Utilization. For the network utilization, upper and lower bounds are defined and measured for both scenarios under moderately and heavily congested network loads.

5.2.1. Upper Bound. For the moderately loaded network which is not included here, the two scenarios almost behave similarly since the packet loss is almost negligible (see Figure 8). On the other hand, in a heavily loaded network, the upper network utilization shown in Figure 5, the localized method shows lower utilization than the randomized method. As mentioned before this parameter on its own may not be a good quality indicator, but its deviation from the lower bound is a better indicator.

5.2.2. Lower Bound. Like the upper bound, the lower bound for the moderately loaded network does not show any noticeable difference between the localized and randomized methods. However, for heavily load network shown in Figure 6, the localized method shows a better utilization than the randomized method. The lower bound can show actual network utilization better than the upper bound, since the lower bound indicates how much of the received packets are useful.

Perhaps the best indication of network utilization is the departure of its higher and lower bounds, as shown in Figure 7. This is because the larger difference between them can indicate that some of the sent packets are lost in the network, congesting it at the cost of receiving packets

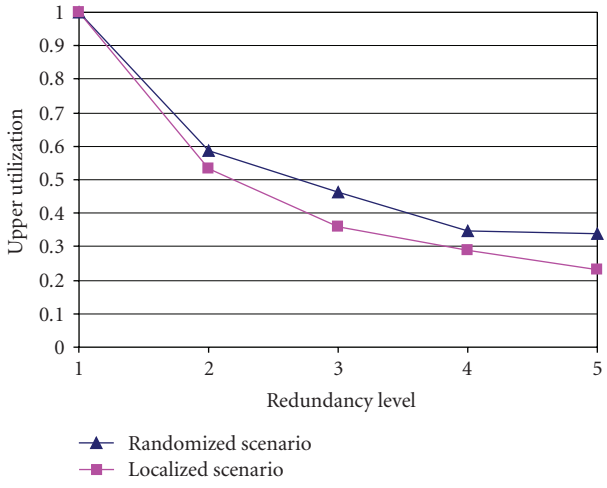


Figure 5: Upper Utilization—Heavily Loaded network.

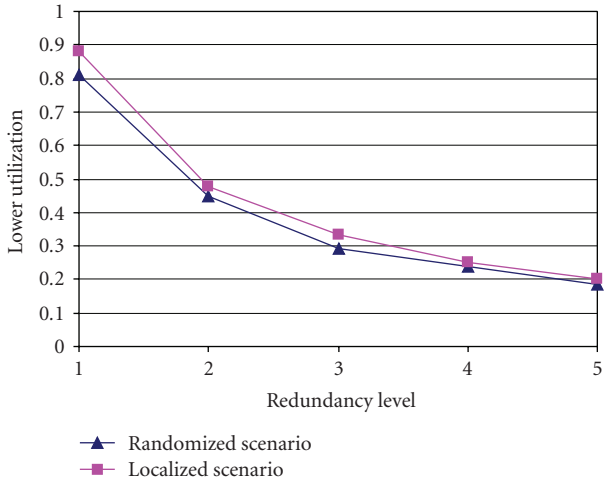


Figure 6: Lower Utilization—Heavily Loaded Network.

from peers to improve video quality. Hence, the smaller the difference between upper and lower bounds of network utilization, the better is the performance, which is in favor of localized method, as shown in Figure 7.

5.3. Quality of Service. Quality of Service was considered to determine whether the proposed approach is also actually affecting the quality at the application (or user) level. In P2P networking, Quality of service is inherited to different metrics on the network. These metrics are intrinsic to each other. For this reason, to quantify and test the localized multistream approach, different effective parameters have been presented and used here which reflect the performance of this proposition.

5.3.1. Packets Loss. Figures 8 and 9 show our findings of packet loss, for the cases of moderately and heavily loaded networks, respectively. At moderate network load shown in Figure 8 both methods do not lead to any packet loss up to a redundancy level equal to 4. This means that for randomized redundant streams, up to 4 sending peers, enough redundant

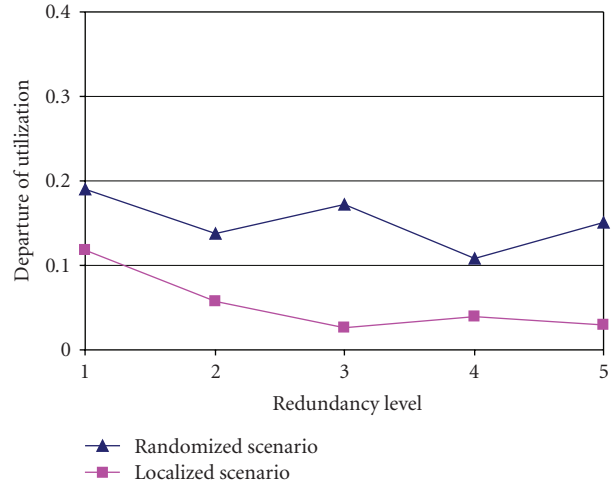


Figure 7: Departure of Utilization—Heavily Loaded.

packets can be received to compensate for any losses. However beyond 4, the added traffic creates congestion that leads to more losses, such that backed up redundant packets may also be lost. With the localized method, the figure shows that even up to 5 redundant streams does cause any packet loss. This is due to the fact that no matter how the network is congested, there is always enough number of redundant packets to be used at the receiver.

A network friendly behavior of the locality aware approach is even more apparent at higher network load, as shown in Figure 9. In this case, the network is brought close to congestion by the background traffic (not by the streams under scrutiny). The network is severely congested; then even one or two senders in action can lead to packet loss. By increasing the redundancy level (e.g., more senders), the packet loss is reduced in both methods. However, with the localized method, there is almost no packet loss after receiving from 3 senders. This is due to the fact that multiple copies of the same packets are now sent to the recipient.

By disparity, the randomized connection of Joost-like approach cannot bring packet loss down to zero. In this case, even increasing the number of senders (more redundancy), the senders themselves create additional congestion such that, beyond 4 senders, congestion increases as shown in Figure 9.

5.3.2. Average End-to-End Delay. Delay is another important quality of experience parameter. Figures 10 and 11 show the average end-to-end network delay, for the moderately and heavily congested scenarios, respectively. It is important to note that at heavy network load, the end-to-end delay under localized connection has the least value at the redundancy level of 3-4 senders. Considering that packet loss rate is almost eradicated with just about 3 redundant senders (Figure 9), it appears that the optimal redundancy level is comprised between 3 and 4.

5.4. Quality of Experience. Finally, the objective and subjective qualities of the decoded video under both loading

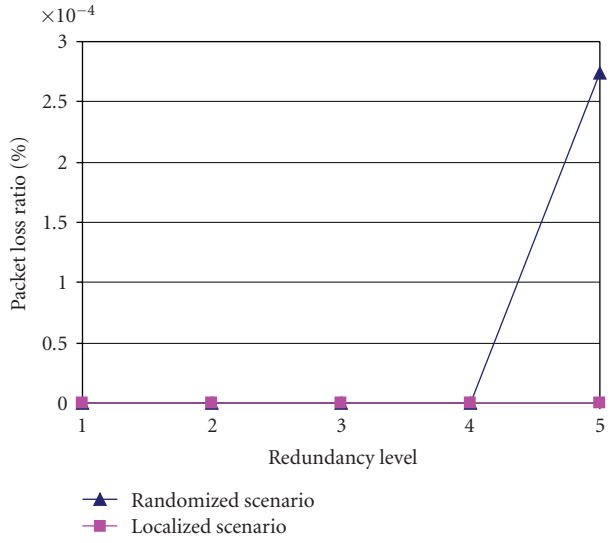


Figure 8: Packets loss ratio—Moderately loaded.

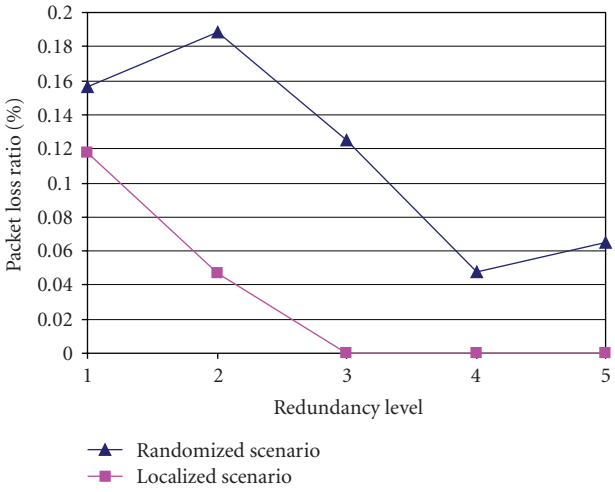


Figure 9: Packets Loss ratio—Heavily loaded.

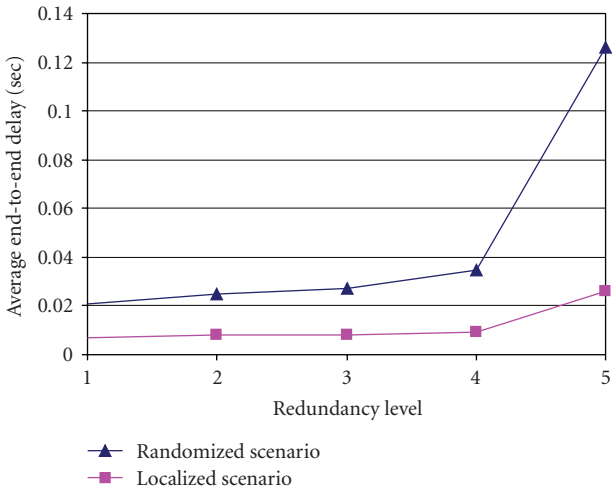


Figure 10: Avg. E2E Delay—Moderately loaded.

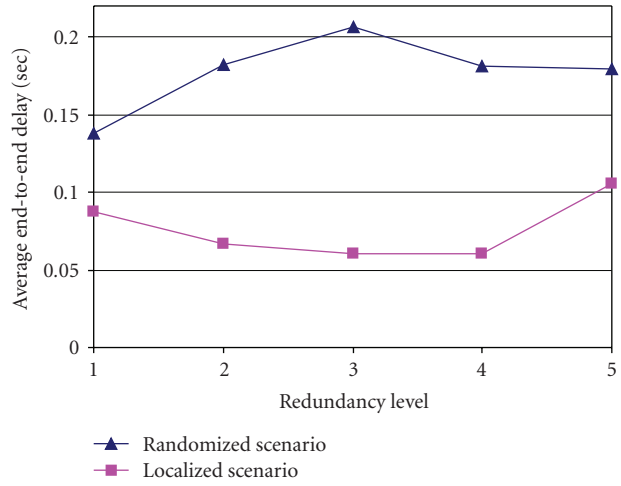


Figure 11: Avg. E2E Delay—Heavily loaded.

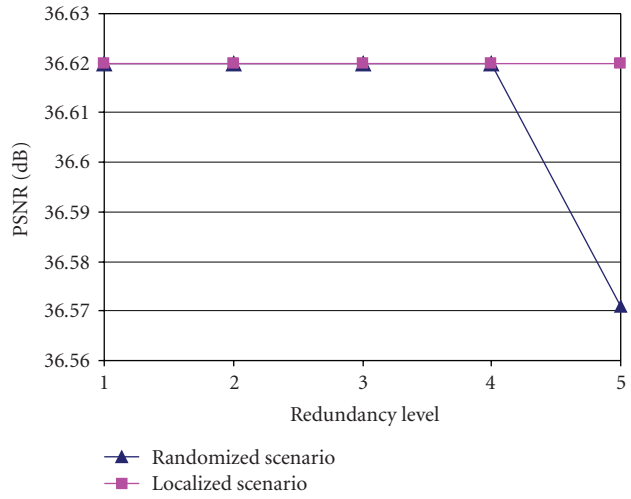


Figure 12: PSNR—Moderately loaded.

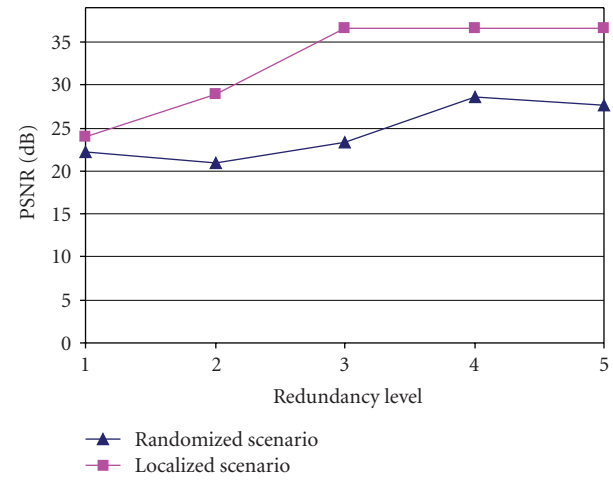
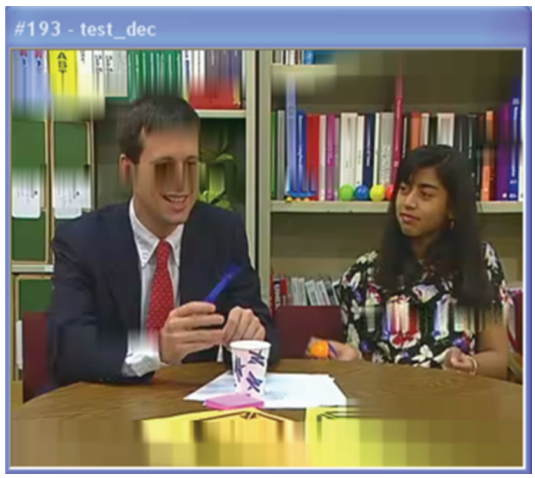


Figure 13: PSNR—Heavily loaded.

(a) 1 Sender (Randomized)

(b) 1 Sender (Localized)



(c) 2 Senders (Randomized)

(d) 2 Senders (Localized)

(e) 3 Senders (Randomized)

(f) 3 Senders (Localized)

Figure 14: Continued.

(g) 4 Senders (Randomized)

(h) 4 Senders (Localized)

(i) 5 Senders (Randomized)

(j) 5 Senders (Localized)

Figure 14: Subjective quality of the localized method against the randomized method for each redundancy level (S1...S5).

conditions are compared. The streams were coded and decoded with an H.264/AVC encoder/decoder of type JM15. Figures 12 and 13 show the objective video quality as measured by PSNR for moderately and heavily congested network perceptively. As expected they exhibit a behavior similar to the packet losses findings of Figures 8 and 9. When the network is moderately loaded, there is hardly any packet lossup to a redundancy of 4, at which point the randomized scenario generates congestions and, thus, a drop in PSNR. The localized approach does not show any packet loss and maintains a constant level of QoE even with 5 injected redundant streams. Figure 13 is also consistent with this rationale. Noticeably, the localized approach improves PSNR steadily up to a redundancy level of 3, where the quality reaches its maximum theoretically achievable value (packet loss is zero at that point).

5.4.1. Subjectivity Quality. For the assessment of the subjective quality perceived by the end-users, in this section snapshots of the two methods are shown. For each case of the five redundant streams, a picture of the proposed localized multistream is compared against the randomized approach (Joost-like). Figures 14(a), 14(b), 14(c), 14(d), 14(e), 14(f), 14(g), 14(h), 14(i), and 14(j) show the subjective quality of the localized method against the randomized methods. Quality of pictures on the table, with the randomized method, specially cup, pen, and papers are distributed in all the redundancy levels, but in the localized method, picture quality is as good as can be compared to the original source.

6. Discussion

In this paper, a localized multi-source is proposed to offer a redundancy of the same content over the network with high quality and low end-to-end delay. This proposal has been tested and run under different scenarios and network conditions. In order to quantify its robustness, it has been run under a point-to-point connection where there is only

one sender and one receiver. Furthermore, multiconnections were introduced where the receiver can connect from 2 and up to 5 senders (peers) simultaneously. Redundant streams are used in combination with locality-awareness to assess our initial hypothesis.

Varieties of connections have been run in two extreme congestion levels, moderately and heavily congested. Consequently, different effective parameters on the network have been measured to show and validate how robust and practical is the proposed localized scheme with the offered redundancy by the chosen sending peers.

However, in order to adjudicate on the goodness of the localized multistream, a benchmark of a popular VoD application [20] is compared against this proposal to show how the localized multistream behaves in contrast to the randomized scheme.

For that reason, one of the valuable and insightful network parameter to be considered is the network utilization which gives the impact of this proposition over the network. Our results (Figure 7) have shown that locality-aware combined with stream redundancy has positive impact on the network utilization. Moreover, another vital network parameter is packets loss; to gauge this factor, a new way of measuring packet loss has been defined mainly to quantify the performance of the proposed localized multistream as shown earlier in (3). The presented results show that the localized scheme is performing better across all the quality of service and QoE parameters. Consequently, by looking at the packets loss ratio, it is apparent from Figures 8 and 9 that the localized approach is better and mainly in case of the heavily congestion network. This was mainly due to the offered awareness of sending peers complemented by stream redundancy.

Moreover, end-to-end delay is almost consistent on both congestion levels and particularly from 2 to 4 senders as shown in Figures 10 and 11; this was taken as the minimum delay among all the received packets. On the other hand, QoE is maintained appropriately on the localized multistream. So, Figures 12 and 13 are an extensive evidence of the perceived video quality to the end-consumers. However, the most divergence can be derived between the two compared schemes on Figure 13 where the network is heavily congested, which is the case of the internet traffic nowadays.

Finally, another interesting point that has resulted from the localized multistream is the finding of the required optimum number of peers to serve a client. According to the presented results by the localized multistream, it is so clear from all the figures that 3 to 4 peers are good enough to provide high quality to the end-users within the current configuration. In contrast to that, with the randomized approach, it is difficult to give an indication for that as the interconnections among peers are chaotic and randomly.

7. Conclusion

A large variety of popular applications, including VoD, live TV, and video conferencing, make use of P2P streaming frameworks. These have emerged from the fundamental principles of insulation and abstraction between the network and the application layers. With this regard, several studies published recently (e.g., [21]), including also some by the authors of this article (e.g., [9, 22, 23]), have identified that when the P2P overlay is designed in isolation from the underlay physical network, the P2P stream has detrimental effects on the network itself. To aim for scalability and user QoE, P2P solutions adopt redundancy, caching, statistical handovers, and other similar techniques, which generate substantial network management and control problems to the network operator.

This problem motivates our initial work aimed at studying ways to maintain the QoE and scalability of the overlay, whilst reducing its detrimental effects onto the underlay. This article represents our initial attempt to pursue network friendly P2P streaming. Our initial hypothesis that the combination of network locality and multistreaming can lead to significant improvements on QoE is reinforced by the initial findings presented herein.

References

[1] Y. Liu, Y. Guo, and C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-Peer Networking and Applications*, vol. 1, no. 1, pp. 18–28, 2008.

[2] Y. Liu, X. Liu, L. Xiao, L. M. Ni, and X. Zhang, "Location-aware topology matching in P2P systems," in *Proceedings of the 32rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 4, pp. 2220–2230, Hong Kong, March 2004.

[3] Z. Xu, C. Tang, and Z. Zhang, "Building topology-aware overlays using global soft-state," in *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS '03)*, p. 500, May 2003.

[4] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," *Computer Communication Review*, vol. 32, no. 4, pp. 205–217, 2002.

[5] B. Zhao, A. Joseph, and J. Kubiatowicz, "Locality aware mechanisms for large-scale networks," in *Proceedings of Workshop on Future Directions in Distributed Computing (FuDiCo '02)*, Bertinoro, Italy, June 2002.

[6] P. Rodríguez-Bocca, *Quality-centric design of peer-to-peer systems for live-video broadcasting*, Ph.D. thesis, 2008.

[7] T. P. Q. Nguyen and A. Zakhor, "Distributed video streaming over Internet," in *Multimedia Computing and Networking*, vol. 4673 of *Proceedings of SPIE*, pp. 186–195, San Jose, Calif, USA, January 2002.

[8] M. Alhaisoni, A. Liotta, and M. Ghanbari, "An assessment of self-managed P2P streaming," in *Proceedings of the 5th International Conference on Autonomic and Autonomous Systems (ICAS '09)*, pp. 34–39, Valencia, Spain, April 2009.

[9] M. Alhaisoni, A. Liotta, and M. Ghanbari, "Resource-awareness and trade-off optimization in P2P video streaming," *Advances in Differential Equations*, vol. 4, no. 1, pp. 59–77, 2010.

[10] M. M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: peer-to-peer media streaming using CollectCast," in *Proceedings of the 11th ACM International Conference on Multimedia (MM'03)*, pp. 45–54, Berkely, Calif, USA, November 2003.

[11] T. Nguyen and A. Zakhor, "Distributed video streaming with forward error correction," in *Proceedings of the 12th*

International Packet Video Workshop, pp. 1–12, Pittsburgh, Pa, USA, April 2002.

[12] M. Mushtaq and T. Ahmed, "Adaptive packet video streaming over P2P networks using active measurements," in *Proceeding of the 11th IEEE Symposium on Computers and Communications (ISCC '06)*, pp. 423–428, IEEE Computer Society, Los Alamitos, Calif, USA, June 2006.

[13] M. M. Hefeeda and B. K. Bhargava, "On-demand media streaming over the internet," in *Proceedings of the 9th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS'03)*, p. 279, 2003.

[14] K. Ponnavaikko and D. Janakiram, "Overlay network management for scheduling tasks on the grid," in *Proceedings of the 4th International Conference on Distributed Computing and Internet Technology (ICDCIT '07)*, pp. 166–171, Springer, Bangalore, India, December 2007.

[15] T. Locher, S. Schmid, and R. Wattenhofer, "eQuus: a provably robust and locality-aware peer-to-peer system," in *Proceedings of the 6th IEEE International Conference on Peer-to-Peer Computing*, pp. 3–11, Cambridge, UK, September 2006.

[16] S. Asaduzzaman, Y. Qiao, and G. Bochmann, "CliqueStream: an efficient and fault-resilient live streaming network on a clustered peer-to-peer overlay," in *Proceedings of the 8th IEEE International Conference on Peer-to-Peer Computing*, pp. 269–278, Aachen, Germany, September 2008.

[17] C. Ragusa, A. Liotta, and G. Pavlou, "An adaptive clustering approach for the management of dynamic systems," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 12, pp. 2223–2235, 2005.

[18] http://www.kazaa.com/.

[19] http://isi.edu/nsnam/ns/.

[20] http://www.joost.com/.

[21] H. J. Kolbe, O. Kettig, and E. Golic, "Monitoring the impact of P2P users on a broadband operator's network," in *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (IM '09)*, pp. 351–358, IEEE Press, New York, NY, USA, June 2009.

[22] N. N. Qadri and A. Liotta, "Effective video streaming using mesh P2P with MDC over MANETS," *Journal of Mobile Multimedia*, vol. 5, no. 4, pp. 301–316, 2009.

[23] A. Liotta and L. Lin, "The operator's response to P2P service demand," *IEEE Communications Magazine*, vol. 45, no. 7, pp. 76–83, 2007.

Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 562832, 10 pages
doi:10.1155/2010/562832

Research Article

Bandwidth Reduction via Localized Peer-to-Peer (P2P) Video

Ken Kerpez,[1] Yuanqiu Luo,[2] and Frank J. Effenberger[2]

[1] *Telcordia Technologies, One Telcordia Drive, Piscataway, NJ 08854-4151, USA*
[2] *Huawei, USA*

Correspondence should be addressed to Ken Kerpez, kkerpez@telcordia.com

Received 1 June 2009; Accepted 14 October 2009

Academic Editor: Stan Moyer

This paper presents recent research into P2P distribution of video that can be highly localized, preferably sharing content among users on the same access network and Central Office (CO). Models of video demand and localized P2P serving areas are presented. Detailed simulations of passive optical networks (PON) are run, and these generate statistics of P2P video localization. Next-Generation PON (NG-PON) is shown to fully enable P2P video localization, but the lower rates of Gigabit-PON (GPON) restrict performance. Results here show that nearly *all* of the traffic volume of unicast video could be delivered via localized P2P. Strong growth in video delivery via localized P2P could lower overall future aggregation and core network bandwidth of IP video traffic by 58.2%, and total consumer Internet traffic by 43.5%. This assumes aggressive adoption of technologies and business practices that enable highly localized P2P video.

1. Introduction

IPTV, IP video on demand (VOD), over-the-top streaming, video downloads, and user-generated content are all becoming increasingly popular with the proliferation of broadband. With the exception of multicast IPTV, these video streams are sent unicast (with a dedicated stream for each source-destination pair) through core and aggregation networks, and together they can consume tremendous amounts of bandwidth. Worse, video streams cannot benefit from statistical multiplexing concentration the way that most Internet data does, because of the need to support simultaneous constant streams at the prime-time peak.

Another trend is the continued popularity of peer-to-peer (P2P). In many cases, P2P traffic traverses long distances, across core networks and multiple Internet Service Provider (ISP) networks, even though the content could have been retrieved from a much closer location. There has been a spate of recent effort to localize P2P traffic, as discussed in Section 2.2 here. Localized P2P traffic may only traverse a few hops instead of ten or twenty, allowing for a vast decrease in core network bandwidth.

This paper expands and extends previous work examining bandwidth implications of localized P2P [1]. This paper focuses on the maximum localization that is generally possible to localize traffic within access networks and central-office (CO) serving areas.

High P2P video traffic volumes can be accommodated across optical access networks; and it is shown here that localized P2P video is readily enabled by the Next-Generation Passive Optical Network (NG-PON) as envisioned by the Full-Services Access Networks (FSAN) Group and others.

Results here show that nearly *all* of the traffic volume of unicast video could be delivered via localized P2P. Strong growth in video delivery via localized P2P could lower overall future aggregation and core network bandwidth of IP video traffic by 58.2% and total consumer Internet traffic by 43.5%. This assumes aggressive adoption of technologies and business practices that enable highly localized P2P video.

2. Driving Trends and Enabling Technologies

2.1. Traffic Growth: P2P and Video. Video and peer-to-peer contents are increasing Internet bandwidth demands. Reports from the Distributed Computing Industry Association (DCIA) [2] predict an "exaflood" from advances

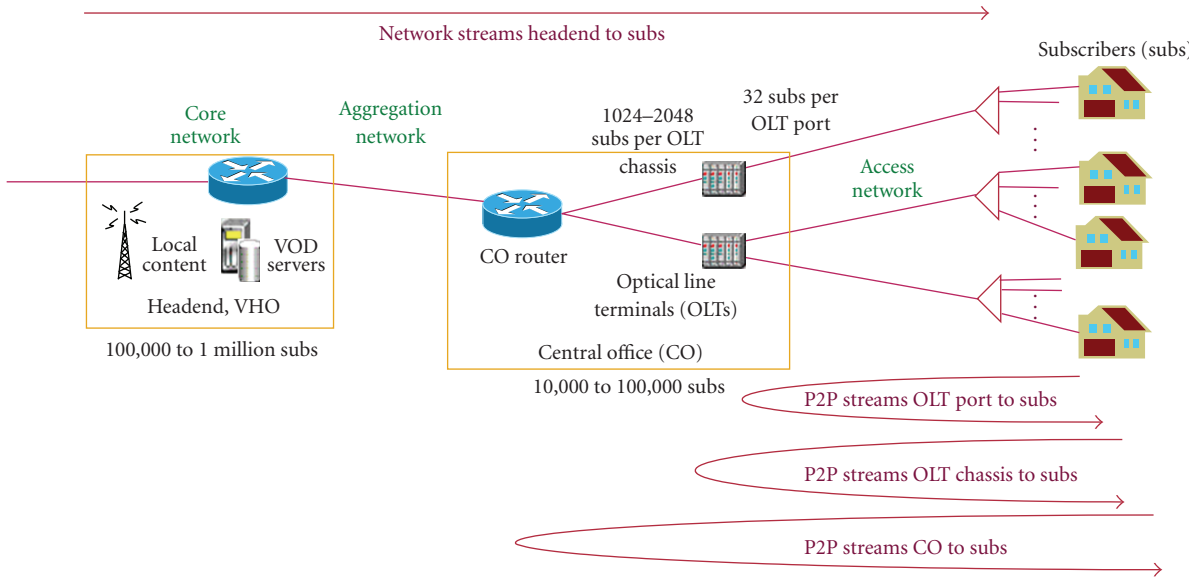


Figure 1: Localized P2P.

in rich media content delivery, particularly via peer-to-peer (P2P) technologies, along with user-generated content (UGC) and video. Recent data [3] shows both global and US Internet bandwidth growing at approximately 40% to 60% per year.

P2P traffic occupies much Internet bandwidth. It was found that 62% of the total volume of Internet traffic was used by P2P in Japan [4–6]. A small segment of users dictate the overall behavior; 4% of heavy-hitters account for 75% of the inbound traffic volume.

Ellacoya Networks studied European Internet traffic in 2005 [7], and found that usage by the top most active 5% of subscribers represented approximately 56% of total bandwidth, while the top 20% of active subscribers consumed more than 97% of total bandwidth. P2P was by far the largest consumer of bandwidth with 65.5% of traffic on the network being P2P applications.

CacheLogic conducted direct packet monitoring of Internet backbones and ISPs data streams via Layer 7 packet analysis [8] in 2005. This study found 61.4% of peer-to-peer traffic to be video, 11.4% audio, and 27.2% other traffic.

Ipoque [9] analyzed the Internet traffic in five regions of the world between August and September 2007. The results for these different regions vary considerably. P2P produced, on average, between 49% and 83% of all Internet traffic. From 47% to 79% of the exchanged P2P traffic was video. BitTorrent and eDonkey were by far the most commonly used P2P systems. Multimedia Intelligence [10] predicts P2P traffic will grow almost 400% over the next 5 years.

Some recent reports suggest that the percentage of Internet traffic which is P2P may have dropped somewhat. DSL Prime [11] reported that P2P dropped from about 40% of Internet traffic in 2007 to between 20% and 26% in early 2008. Streaming, largely YouTube, appears to have taken up the slack. Overall Internet traffic growth was reported at 30–40% percent per year.

On the other hand, some reports show P2P traffic share increasing. Sandvine [12] presented recent statistics of Internet traffic in North America in 2008. In aggregate, consumer broadband traffic was 44% P2P, a slight increase from 2007 when 40.5% of traffic was P2P. This increase could perhaps be attributed to improved DPI recognition of P2P. In aggregate, web browsing was 27% and streaming was 15% of North American consumer broadband traffic. P2P, web browsing, and streaming constituted much of the downstream traffic, while P2P dominated upstream with 75% of the upstream traffic.

Cisco [13] has presented a detailed forecast with numbers similar to the other studies but more comprehensive. Cisco projects 49% annual growth from 2007 to 2012 in overall Internet traffic, with only 31% annual growth for P2P but with 68% annual growth in Internet video plus IPTV traffic. Cisco estimated that 60% to 70% of P2P traffic is actually video. As seen in Figure 2, Internet video alone is expected to account for 50% of all consumer Internet traffic in 2012, in addition to P2P video and IPTV.

Forecasts show strong growth in Internet video. In-Stat [14] has recently forecasted that 160 billion user-generated served videos will be served in 2012. User-generated video revenue is projected at $1.2 billion by 2012. IPTV growth projections are also robust; Gartner projects that in one year (2008) there will be a 64.1% increase in global IPTV subscribers and 93.5% increase in IPTV revenue to $4.5 billion. Recent forecasts from Strategy Analytics predict that IPTV service revenues in the USA will grow from $694 million in 2007 to nearly $14 billion in 2012 [15].

The underlying trend is continued growth in P2P Internet traffic, and strongly increasing growth in video traffic. Much video is now being streamed from servers, from YouTube, or from content owners such as CBS, ABC, NBC, Netflix, and so forth. If more video shifts to P2P, then we can expect very large increases in P2P traffic.

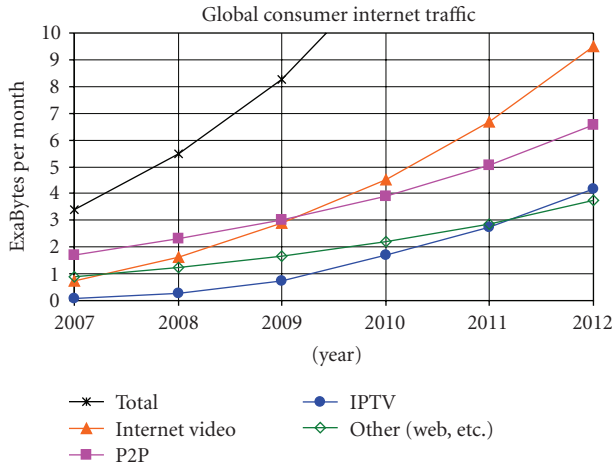


FIGURE 2: Cisco [13] forecasts of Global Consumer Internet Growth. One ExaByte = 10^{18} Bytes.

2.2. Localizing and Legitimizing P2P. Localized P2P can save network bandwidth by delivering titles from a source that is closer than currently used peers, and perhaps even closer than the nearest network-owned video server [1]. P2P systems that prefer to get content from the most local sources should have traffic that traverses fewer links. An article [16] showed that 99.5% of current P2P traffic (using "eDonkey" in France) traversed national or international networks. It further showed that 41% to 42% of this long distance traffic could be made local if a preference for local content was built into the protocol.

Methods for localizing P2P have been discussed. It is not uncommon to use the IP hop count or TTL value to localize somewhat when choosing peering sources. Current routing schemes in P2P networks such as Chord [17] work by correcting a certain number of bits at each routing step. Reference [18] used the IP number to localize, and found that the first octet in the IP number provided localization to roughly a national level, an improvement over global delivery. Reference [19] accesses hosts in the same domain of the DNS namespace, for example, in the same country, which are often also geographically closer than hosts in other domains or countries. Oversi has a product called NetEnhancer that automatically routes P2P traffic to local peers, and away from interconnect peers and overloaded access networks.

P2P may work in concert with traditional content network delivery. References [20, 21] both focus on methods that allow partial control of P2P traffic by network providers, with [20] focusing on P2P. Pando Networks Inc. and Kontiki are companies now offering peer-assisted content delivery. A recent popular method of network-provider control for P2P is called "Proactive network Provider Participation for P2P (P4P)." P4P [21, 22] provides network information (called "itrackers") to distributed P2P systems which then optimize their traffic through the network. Pando Networks claims that P4P tests at Verizon, Comcast, and AT&T sped delivery and increased the percentage of data routed internally. The percentage of data delivered within each ISP increased from fourteen percent (14%) with normal P2P delivery to as much as eighty-nine percent (89%) with P4P delivery.

The DCIA and Pando Networks have formed the P4P Working Group (P4PWG) to help specify P4P. The IETF P2PSIP Working Group is working to standardize SIP signaling for P2P and user and resource location data in an SIP environment with no or minimal centralized servers. The IEEE Next Generation Service Overlay Network (NGSON) Working Group may help enable carrier-grade services delivery over P2P. NGSON considers a control-plane "network overlay" for controlling P2P systems. These P2P overlays can be broadly categorized as three types: Structured (based on network connectivity and number of hops), Hierarchical (with multiple autonomous groups), and Semantic (using hashed-based loop-up routing).

Recent reports say that P2P will be integrated into traditional content distribution [23]. P2P Content Distribution Network (CDN) leader Akamai has recently bought the company Redswoosh and now has their P2P distribution system. Abacast is a hybrid CDN, combining the advantages of P2P, both for live streaming and on-demand downloads.

Digital rights management (DRM), licensing, and copy control systems are emerging for P2P content [24, 25]. New PCs have powerful processors that can perform strong encryption and help enable distributed content management. Reference [26] defines a distributed P2P content delivery capability with authentication, personalization, and payment functions that can increase carrier and content owner revenues.

It's predicted that the bulk of P2P will become content that is legitimate and licensed [2]. P2P is predicted to become a legitimate mainstream distribution method, and licensed P2P content is expected to grow 10 times as fast as unauthorized P2P traffic [10]. Some content filtering engines are said to be able to detect and interdict illegal content distribution at any point in a network. MediaDefender and Cloudshield can be used for content protection on P2P. Brilliant Digital Entertainment has developed an application for Internet service provider (ISP) networks that is said to automatically and rapidly identify illegal files.

2.3. P2P Video. Much current P2P traffic is video, usually file downloads. In the future, P2P may take a more central role in delivering Internet video and IPTV [27]. P2P can be employed to enable non-real-time download of video on demand (VOD). Personal Video Recorders (PVRs) in subscribers' set-top boxes that are deployed by network or service providers could host P2P video in order to offload traffic from network video servers. A number of control, oversight, and copy protection issues are being addressed to make this practical [28]. Other work has enabled fault tolerant and cost-effective P2P Video on Demand (VOD) [29, 30]. Studies have explored the bandwidth and performance of time-shift TV (TSTV), which allows users to pause, fast forward, or rewind a title that was viewed elsewhere [31].

Streaming video with P2P poses unique delay and bandwidth challenges. New work is enabling P2P to stream linear broadcast TV by using application-layer multicast [32].

Router-level multicast infrastructure is used by network providers to limit core bandwidth, and multicast can be combined with P2P in interesting ways, for example allowing "VCR" controls for P2P linear broadcast service [33, 34]. Most successful P2P streaming application-layer multicast systems use fairly simple tree structures, but more recent work is looking at using multiple trees, meshes [35, 36], data chunks, and swarms. Recent measurements of a large-scale live streaming video P2P multicast reported that most traffic could be offloaded by P2P, with about 15% of traffic still streamed from servers [37].

Popular P2P protocols, as exemplified by BitTorrent, download many different chunks of a file from many different peers, lowering the impact of individual upstream bottlenecks. Users form a "swarm" to share content, and use a "torrent file" that identifies pieces of data to retrieve in the swarm. BitTorrent's non-contiguous download methods do not appear to be suited to real-time streaming, however, recent work has modified BitTorrent to "LiveSwarms" [38] which enables P2P streaming. The "Swarm Player" is being developed as an open source project for P2P streaming.

There are now a number of companies and platforms that provide P2P streaming including Joost, Veoh, Nextshare, Vudu, PiCast, Vatata, End System Multicast (ESM), Gridmedia, PPStream, PPLive, Zattoo, Octoshape, Sopcast, Tvkoo, Roxbeam, Tribler, Ustream, RedSwoosh, Mediamelon, RawFlow, Selfcast.com, Velocix, NeoKast, BitTorrent, uTorrent, Brightcove, and so forth. [2]. The Vudu Internet set-top box is said to predownload the first few minutes of the most popular movies to give the viewer a "head start" using a sort of proprietary P2P technology to retrieve content from other Vudu boxes.

The European P2P-Next consortium has created a set-top box (STB) prototype that allows P2P streaming, which is called "NextshareTV" and was developed at the Pioneer Digital Design Center in London. NextshareTV is being developed for PCs as well as STBs, and it makes use of some of the open-source P2P streaming technology of the Swarm Player. Octoshape has P2P video streaming technology called the "grid delivery system" that is said to enable HDTV (at about 2.5 Mbps).

There is clearly much current interest and many new developments in localized P2P, network-controlled P2P, P2P video, and P2P streaming.

3. Bandwidth Evaluation Methodology

The remainder of this report shows calculations of video bandwidth usage and the impact of localizing P2P video. Models are created representing typical usage.

3.1. Video Demand Model. The most popular video titles are viewed far more than the least popular. This is particularly true for TV shows, first-run movies, and recently released DVD rentals. TV, movie, and video rental rating statistics [39–41] were examined, and an overall model for the demand of video titles was created by matching these statistics. The model first rank orders all video titles, from most popular to least popular. The title number increases with decreasing popularity. A probability density model is assigned to the title numbers. A series of curve fits were made to several datasets of number of viewers of different types of video content, with the video model here constructed as a composite of these curve fits to represent all overall video usage.

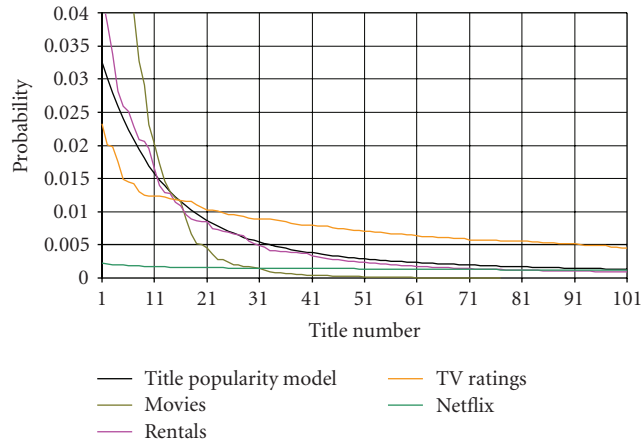


Figure 3: Video demand statistics and model. Statistics and probabilities are normalized so that the probability density of title #1 = 1.

Long-tailed distributions often follow a "Zipf-law" probability distribution, also known as a power-law or hyperbolic probability. The video demand model here uses this power-law to fit the long tail, formally called the hyperbolic probability density. A recent paper [42] also showed that the power law is a good fit to Netflix and YouTube video demand, which are both very long tailed. The YouTube data matched a power-law with correlation 0.8. Statistics for higher title numbers, out in the very long tail, are represented by Netflix [42] and video rental data here. The long tail is somewhat supply-driven: as more titles are offered, someone will eventually view them.

On the other hand, some video content is not long tailed and is instead highly weighted toward the most popular titles. It was found that the popularity of linear broadcast TV channels and new movie releases are modeled closely by an exponential probability density function, with a few titles very popular and the less popular content rapidly dropping off with increasing title number. These include weekend movie gross.

A combined video demand model is used here; this is a mixture of exponential and hyperbolic probability densities. This model is a slightly modified version of a previous model [1]; there is more weight on the tail in the new model to include the Netflix data. The statistics and model are shown in Figure 3 for popular titles with low title numbers.

The model density is a combination of hyperbolic (Zipf or power-law) and exponential densities. The hyperbolic component models the long-tail Netflix data, while the exponential component models content such as new movie releases. The density is truncated to limit to a finite number

of available titles, title # n, such that $nmin \le n \le nmax$. The mathematical definition of the model is

$$\begin{aligned} &\text{Video demand probability model} \\ &= (\text{In})\text{hy}(n) + (1 - \text{In})\text{ex}(n), \end{aligned} \tag{1}$$

with

$$\begin{aligned} \text{hy}(n) &= \text{truncated hyperbolic density}, \\ \text{ex}(n) &= \text{truncated exponential density}, \end{aligned} \tag{2}$$

where In is an indicator for a Bernoulli random variable; $\Pr(\text{In} = 1) = p_1$, $\Pr(\text{In} = 0) = p_2$; and satisfying $p_1 + p_2 = 1$.

Further, the truncated hyperbolic density is

$$\text{hy}(n) = C1^* n^{-A}, \qquad nmin \le n \le nmax, \tag{3}$$

where A is a constant, and

$$C1 = \frac{1 - a}{nmax^{(1-A)} - nmin^{(1-A)}}. \tag{4}$$

The mean of the truncated hyperbolic probability density is

$$E(\text{hy}(n)) = \frac{C1^* \left(nmax^{(2-A)} - nmin^{(2-A)}\right)}{2 - A}. \tag{5}$$

The truncated exponential density is

$$\text{ex}(n) = C2^* e^{(-B^* x)}, \qquad nmin \le n \le nmax, \tag{6}$$

where B is a constant, and

$$C2 = \frac{B}{e^{-(B^* nmin)} - e^{-(B^* nmax)}}. \tag{7}$$

The mean of the truncated exponential density is

$$\begin{aligned} &E(\text{ex}(n)) \\ &= \left(\frac{C2}{B}\right)^* \left(nmin^* e^{(-B^* nmin)} - nmax^* e^{-(B^* nmax)}\right) + \left(\frac{1}{B}\right). \end{aligned} \tag{8}$$

The overall mean is

$$\begin{aligned} &E(\text{Video demand probability model}) \\ &= p_1^* E(\text{hy}(n)) + p_2^* E(\text{ex}(n)). \end{aligned} \tag{9}$$

The TV title demand model parameters used in numerical evaluations here are

$$A = 0.3, \quad B = 0.09, \quad p_1 = 0.8, \quad p_2 = 0.2. \tag{10}$$

For all results in this report, the top 20 most popular titles are broadcast/multicast, the rest (64%) is unicast VOD or P2P. The broadcast/multicast titles are not considered here other than that they reduce the overall demand for unicast video. The total available number of different video titles is 10 000 here.

Table 1: TV viewing hours per day in 2003, FCC data.

Men	Women	Teens	Children	Average
4:29	5:05	3:07	3:14	3:58

Table 2: Model used in analyses here of probability of number of video streams per IP video subscriber in the busy hour (prime time). Average = 1.8 streams per sub.

Number of titles	0	1	2	3	4	>4
Probability	0.1	0.35	0.3	0.15	0.1	0

3.2. Number of Titles Demanded by Each Subscriber. Another aspect is the number of video streams that are viewed by each subscriber. Statistics [40] show that the maximum busy hour (prime time peak) has about 66% of all households watching TV. Also, each home averages a little less than two simultaneous TV viewings per home in the busy hour.

According to FCC statistics in 2005, 90% of US homes had TVs, and 67.5% subscribed to cable TV. The average number of TV sets per household was 2.62. The average TV viewing hours per day are listed in Table 1.

IP video subscribers may watch a little more video than the average person. A simple discrete and independent model for the number of video viewings per home suffices here, an example of which is in Table 2. The average total number of streams to each subscriber is 1.8.

3.3. Serving Area Sizes. Previous work [41] showed that a Gamma probability can closely model statistics of telecom serving area sizes. The video demand model presented here uses a Gamma model to determine the number of subscribers per Central Office (CO).

The gamma probability density function (pdf), defined as

$$\begin{aligned} &\Pr(\text{radius} \le x) \\ &= \int_0^x \frac{1}{\Gamma(\alpha)\beta^\alpha} x^{\alpha-1} e^{-x/\beta}\, dx \text{ with } \Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx. \end{aligned} \tag{11}$$

Define the mean of the Gamma to be μ, and the standard deviation to be σ. Then

$$\mu = \alpha\beta; \quad \sigma^2 = \alpha(\beta^2); \quad \alpha = \frac{(\mu^2)}{(\sigma^2)}; \quad \beta = \frac{(\sigma^2)}{\mu}. \tag{12}$$

Here the gamma model determines serving area radii. Given the radius, r, of the serving area, a uniform subscriber density is assumed and the number of subscribers in the serving area is simply the average subscriber density multiplied by the serving area size, πr^2.

The gamma model parameters used for modeling CO-serving area size here are as in Table 3 with:

$$\alpha = 25, \quad \frac{\sigma}{\mu} = 0.2, \tag{13}$$

resulting in the probability shown in Figure 4, which is truncated to a maximum value due to computer memory limitations in the evaluations here.

Table 3: Simulated serving area sizes for P2P results.

67000	Average number of homes per CO (1983 loop survey [41])
35	Percent NGPON penetration rate (guesstimate)*
23450	Average number of subscribers per CO

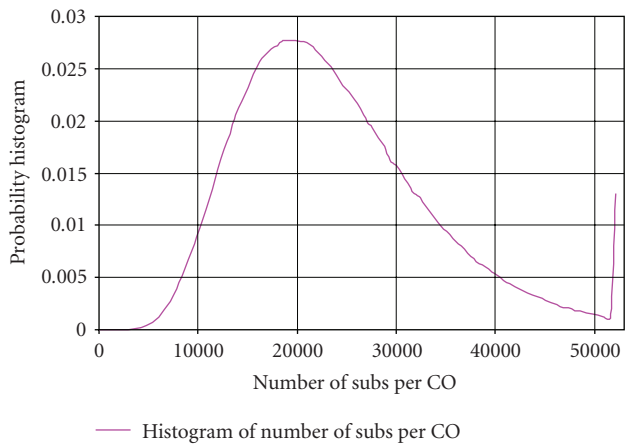


Figure 4: Model of CO-serving area sizes.

3.4. NG-PON Limits. P2P traffic could exhaust streaming and upstream capacity, and so realistic capacity limitations on P2P traffic are imposed in this study. Downstream capacity is not considered here because downstream bandwidth would be used anyway if the video was network-provided and not P2P. Each peer is assumed to be capable of simultaneously sourcing at most two P2P video streams. This work is forward-looking, so all video is assumed to be HDTV at 19.3 Megabits per second (Mbps) per stream.

P2P should not be allowed to exhaust all resources, and so up to at most 1/3 of any resource is allowed to be used by P2P.

Most results are for a next-generation passive optical network (NG-PON) access network, which is assumed to have the following capacity limitations:

(i) Each Optical Line Terminal (OLT) port can support 32 subscribers.

(ii) Each OLT port has upstream bandwidth of 2.488 Gbps, which can support 42 P2P video streams.

(iii) The OLT switching capacity is 400.0 Gigabits per second (Gbps), which can support 6908 P2P video streams.

(iv) The OLT uplink bandwidth = 40.0 Gbps, which can support 690 P2P video streams

(v) The CO-router throughput is one Terabyte.

In any cases where these limits would be exceeded, the demanded video streams are not served by P2P but instead by network servers. Much of the localized P2P traffic is served by, and goes to, subscribers on the same OLT, which would use no OLT uplink capacity, and so the OLT uplink capacity usually does not limit P2P here.

4. Bandwidth Impacts

Monte-Carlo simulations here repeatedly randomly generate CO-serving areas, video demand, P2P supply, and so forth, and collect statistics on P2P bandwidth usage. The stochastic models described in the previous section are used for serving area sizes, numbers of P2P titles, title selection, and storage. PON, OLT, and upstream streaming sizing are held constant to represent one fiber connection per home, with one fiber service provider.

A number of subscribers per CO is randomly generated by the model of Section 3.3, then these subscribers are assigned to OLT ports, and OLTs. An OLT supports a randomly-generated number of subscribers that is uniformly distributed between 1024 and 2048. OLT ports are filled with 32 subscribers. Each subscriber is randomly assigned some number of simultaneously demanded video streams according to the model of Section 3.2. A title is assigned to each demanded video stream using the model of Section 3.1.

A percentage of all subscribers are each assumed to store some number of P2P titles. The identity of these stored titles is determined by the same model as used for video demand in Section 3.1. Each of these stored titles can be delivered if demanded by other subscribers, from the closest source that stores the demanded title and has capacity for another stream. The simulation first searches to see if the demanded title can be served from a subscriber on the same OLT port (the PON); if there are none then all subscribers on the OLT are searched, if there are none then the all subscribers in the CO are searched (the CO-router area), and then if there are none the title is delivered through the core and aggregation network from the headend (*not* localized).

Statistics count up the number of video streams in each segment of the network. Simulations regenerate all serving area sizes, demanded video titles, stored P2P titles, and so forth, 250 times.

The most popular titles are highly likely to be demanded, *and* they are highly likely to be stored for P2P availability. This results in the localized P2P system here very often delivering content from nearby sources. If only very long-tailed contents were available for P2P, then less traffic would be localized.

4.1. Detailed Results. A P2P stream is switched or routed from upstream back to downstream at the "hairpin" location. Figure 5 shows results in terms of number of unicast video stream per CO in the busy hour (recall that the top 20 titles are broadcast/multicast and not included here). These figures vary the number of different video titles that are stored at each peer and are available for streaming to another peer along the x-axis. Recall that each peer can only transmit two P2P streams simultaneously. As more titles are stored, it is more likely that any demanded title can be transmitted from a local peer. As the number of stored titles available from each subscriber increases, the most local traffic at the same OLT port increases, and the most distant traffic from non-P2P unicast servers decreases. Traffic hairpin routed at the intermediate local points, the OLT chassis and the CO router, rises and falls.

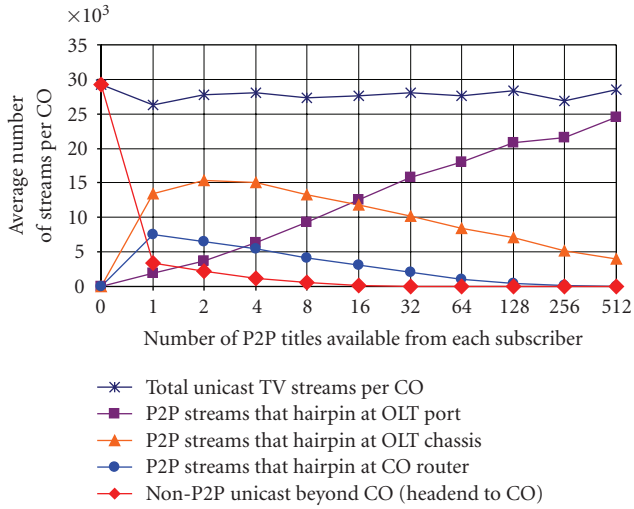


Figure 5: Detailed results: average numbers of streams. NG-PON.

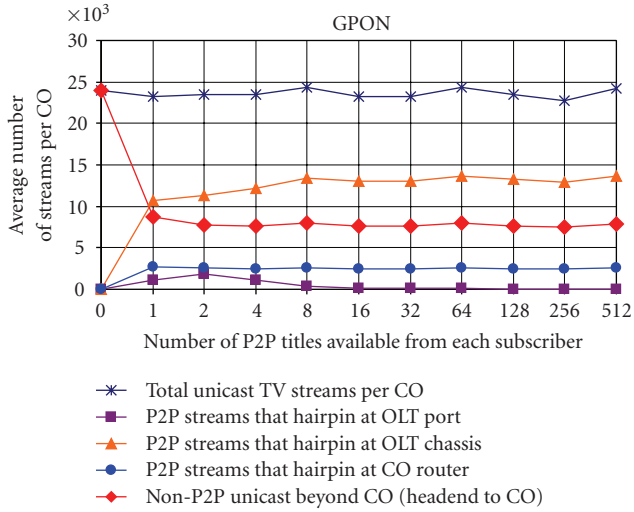


Figure 6: Number of streams with GPON limits on upstream P2P Capacity.

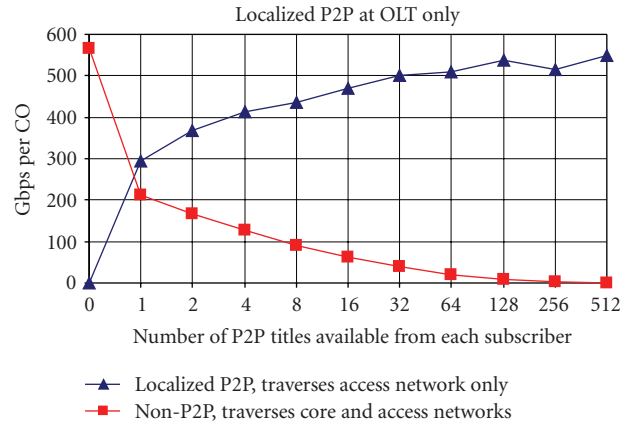


Figure 7: Localized versus non-localized unicast video bandwidth. P2P hairpins at the OLT chassis or OLT port. Customers on the same OLT can share local P2P video. NG-PON.

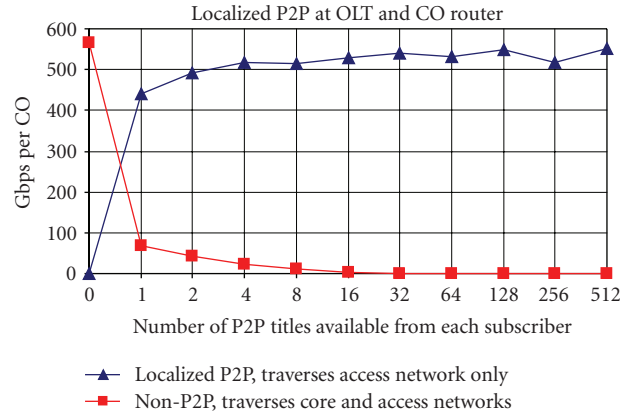


Figure 8: Localized versus non-localized unicast video bandwidth. P2P hairpins at the OLT chassis, or OLT port, or CO-router. Customers at the same CO can share local P2P video. NG-PON.

The most telling curve in Figure 5 is the number of non-P2P unicast streams beyond the CO (from headend servers). With NG-PON and localized P2P, these aggregation and core network streams can drop to nearly zero, less than 0.1% of the total unicast video traffic. This is because the most popular video titles are also the most commonly stored, and demand can be met via P2P. Background traffic has little effect on these results since the GPON resources are sufficient.

4.2. GPON Compared to NG-PON. Simulations were run identical to previous Sections 3 and 4.1, except with access network bandwidth limitations representative of a Gigabit-Passive Optical Networks (GPON):

(i) 1.244 Gbps per-PON upstream instead of NG-PON 2.488 Gbps,

(ii) 10 Gbps OLT uplink instead of NG-PON 40 Gbps,

(iii) 200 Gbps OLT backplane instead of NG-PON 400 Gbps.

Results are shown in Figure 6. Unlike NG-PON, the number of non-P2P unicast streams beyond the CO (from headend servers) does not approach zero. Instead, it reaches a floor of about 7500 streams per CO or about 32% of the total number of unicast video streams.

4.3. Video Core Bandwidth Reduction. This section distills previous results further to determine overall impact of localized P2P in alleviating core and aggregation network bandwidth. Here, the number of video streams is converted into bandwidth by assuming HDTV at 19.3 Mbps per stream.

The figures here clearly show the impact of localized P2P on core and access bandwidth. Figures 7 and 8 show that nearly *all* unicast video (99.9%) can be delivered locally with P2P. Figure 9 shows that NG-PON can fully enable

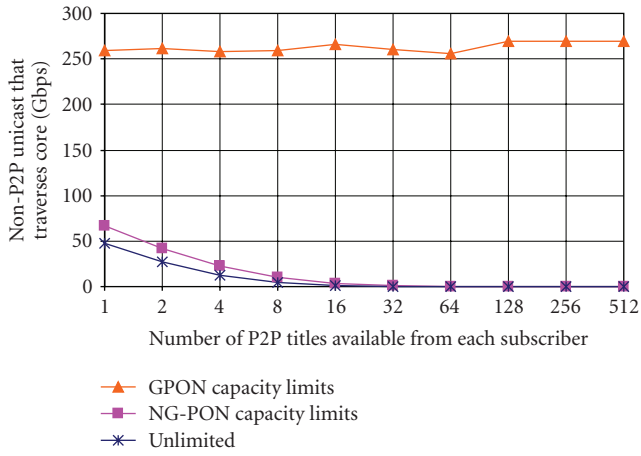


Figure 9: Comparing unicast video streaming core and aggregation bandwidth with GPON, NG-PON, and unlimited upstream capacity for P2P.

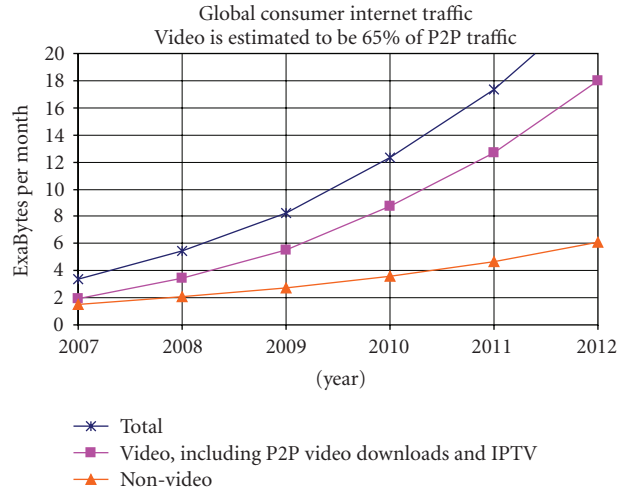


Figure 10: Aggregated Cisco projections for global consumer Internet growth: Video is projected to grow to be almost 75% of total consumer broadband traffic.

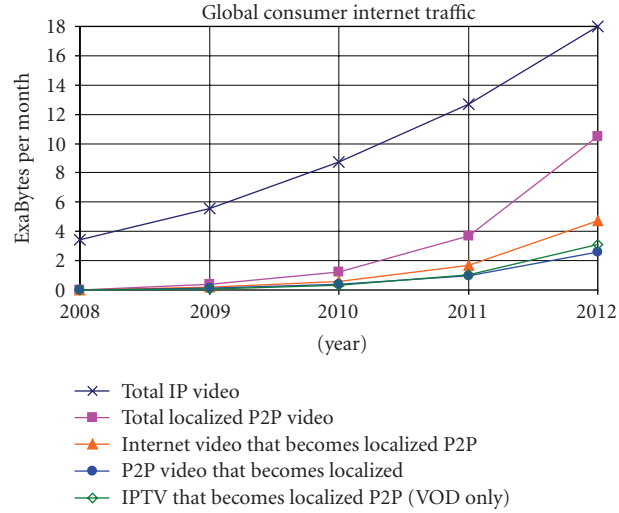


Figure 11: Combined overall projected growth: localized P2P growth and Cisco forecast.

this localization, nearly as well as a theoretical system with infinite capacity ("Unlimited"). However, GPON has insufficient bandwidth to fully enable localized P2P video.

IP Video is either unicast (one-to-one) or multicast (one to-many). Multicast only needs a fixed amount of core and aggregation bandwidth, less than about 5 Gbps. The unicast video bandwidths here are on the order of hundreds of Gbps, and so unicast bandwidth can be considered to be about equal to the entire video bandwidth.

4.4. Overall Reduction in Internet Core Bandwidth. Section 2.1 briefly presented the Cisco study projecting consumer Internet growth [13], and showed that these projections are reasonably similar to trends other reported elsewhere [12, 14, 15]. Numbers from the Cisco study are used in this section, primarily because Cisco projected the growth of the individual components of Internet traffic: IP video, P2P, IPTV, and so forth.

This section assumes that a carrier aggressively pursues network-controlled, localized P2P. Cisco estimated a Compound Annual Growth Rate (CAGR) of over 100% for Internet Video to TV, and for IPTV. A 100% CAGR of localized P2P video is assumed here from 2009 to 2012, resulting in the following percentages of traffic that becomes localized P2P in 2012: Internet video 50%, P2P video 60%, and IPTV (VOD only) 75%. There is currently (in 2008) almost no P2P video traffic that is localized to within a single CO, and so the starting point is zero in 2008. Also, 65% of P2P traffic are assumed to be video here, which is mid-way between the current values of 60% to 70% reported by Cisco. These assumptions and the data in Figure 10 generate the projections of localized P2P video impact on total consumer Internet bandwidth shown in Figure 11.

The data in Figure 11 shows that in 2012, localized P2P is projected to grow to 58.2% of IP video traffic, and 43.5% of all consumer Internet traffic.

5. Summary

This report showed that localized P2P video distribution can greatly decrease core and aggregation network bandwidth growth.

Cisco estimates that video will grow to nearly 75% of consumer Internet traffic in 2012. Over 99% of Internet video plus IPTV bandwidth in the core network are expected to be unicast, since multicast is very efficient in the core.

Assuming ubiquitous use of technologies for P2P localization, calculations here showed that as much as 99.9% of all unicast video traffic can be localized to a single OLT serving area, and even more can be localized to a single CO serving area. Changing content appetites and practical considerations could lower the share of localized P2P somewhat to perhaps 90%, 80%,... of IP video, but this would still be a highly substantial portion.

Assuming aggressive but reasonable penetration rates, in 2012 localized P2P is projected to be able to grow to 58.2% of IP video traffic, and 43.5% of all consumer Internet traffic. This localized P2P video traffic could exceed 10 ExaBytes per month—twice the entire traffic volume of the current consumer Internet!

There is a clear need for sufficient information flows, rights management, and incentives between subscribers and network service providers to enable localized P2P. Further, this report showed that P2P video localization down to an individual access multiplexer such as an OLT, or down to individual COs, can be highly beneficial and should be supported. Mechanisms are necessary to assure video flows with acceptable QoS.

References

[1] K. Kerpez, "Bandwidth impacts of localizing peer-to-peer IP video traffic in access and aggregation networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2008, Article ID 491860, 7 pages, 2008.

[2] Distributed Computing Industry Association (DCIA), http://www.dcia.info/.

[3] Minnesota Internet Traffic Studies (MINTS) data, http://www.dtc.umn.edu/mints/home.php.

[4] K. Cho, K. Fukuda, H. Esaki, and A. Kato, "The impact and implications of the growth in residential user-to-user traffic," in *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 207–218, Pisa, Italy, September, 2006.

[5] "Draft Report of the Study Group on a Framework for Competition Rules to Address the Transition to IP-Based Networks," "New competition promotion program 2010", July 2006.

[6] K. Cho, K. Fukuda, H. Esaki, and A. Kato, "Observing slow crustal movement in residential user traffic," in *Proceedings of the 4th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT '08)*, Madrid, Spain, December 2008.

[7] M. Burke, "Ellacoya networks releases data on broadband subscriber and application usage in Europe," September, 2005, http://www.ellacoya.com/news/pdf/2005/EllacoyaEuropeData.pdf.

[8] A. Parker, "P2P in 2005," http://www.cachelogic.com/home/pages/studies/2005_01.php.

[9] H. Schulze and K. Mochalski, "Internet Study 2007," http://www.ipoque.com/news_&_events/internet_studies/internet_study_2007.

[10] F. Dickson, "P2P: content's "Bad Boy"; tomorrow's distribution channel," http://multimediaintelligence.com/index.php?page=shop.product_details&flypage=flypage.tpl&product_id=21&option=com_virtuemart&Itemid=80.

[11] D. Burstein, "p2p down to 20–25% of traffic," DSL Prime, May, 2008.

[12] Sandvine, "2008 analysis of traffic demographics in North-American broadband networks".

[13] Cisco, "Cisco visual networking index—forecast and methodology. 2007–2012," http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html.

[14] In-Stat, "User-generated video, a global stage for you," 2008.

[15] The Bridge, "Telco video status report—November, 2008," http://www.mediabiz.com/thebridge/index.cfm.

[16] L. Plissonneau, J.-L. Costeux, and P. Brown, "Detailed analysis of eDonkey transfers on ADSL," 0-7803-9455-0/06, IEEE, 2006.

[17] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup service for internet applications," in *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp. 149–160, San Diego, Calif, USA, August 2001.

[18] R. A. Ferreira, A. Grama, and S. Jagannathan, "An IP address based caching scheme for peer-to-peer networks," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '03)*, vol. 7, pp. 3845–3850, San Francisco, Calif, USA, December 2003.

[19] G. Pfeifer, C. Fetzer, and T. Hohnstein, "Exploiting host name locality for reduced stretch P2P routing," in *Proceedings of the 6th IEEE International Symposium on Network Computing and Applications (NCA '07)*, pp. 134–141, Cambridge, Mass, USA, July 2007.

[20] M. Cha, P. Rodriguez, S. Moon, and J. Crowcroft, "On next generation telco-managed P2P TV architectures," in *Proceedings of the 7th International Workshop on Peer-to-Peer Systems (IPTPS '08)*, Tampa Bay, Fla, USA, February 2008.

[21] H. Xie, A. Krishnamurthy, A. Silberschatz, and Y. R. Yang, "P4P: explicit communications for cooperative control between P2P and network providers," http://www.dcia.info/documents/P4P_Overview.pdf.

[22] http://www.openp4p.net/front/p4pwg.

[23] http://digital.venturebeat.com/2008/05/30/bittorrents-plan-ride-the-video-wave-to-the-bank/.

[24] http://www.rightsflow.com/.

[25] http://www.velocix.com/.

[26] B. Falchuk, D. Gorton, and D. Marples, "Enabling revenue-generating digital content distribution for telecom carriers," in *Proceedings of the 3rd IEEE Consumer Communications and Networking Conference (CCNC '06)*, vol. 2, pp. 1144–1148, Las Vegas, Nev, USA, January 2006.

[27] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari, "Will IPTV ride the peer-to-peer stream?" *IEEE Communications Magazine*, vol. 45, no. 6, pp. 86–92, 2007.

[28] T. T. Do, K. A. Hua, and M. A. Tantaoui, "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment," in *Proceedings of IEEE International Conference on Communications (ICC '04)*, vol. 3, pp. 1467–1472, Paris, France, June 2004.

[29] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2Cast: peer-to-peer patching scheme for VoD service," in *Proceedings of the 12th International World Wide Web Conference (WWW '03)*, Budapest, Hungary, May 2003.

[30] C. Huang, J. Li, and K. Ross, "Peer-assisted VoD: making internet video distribution cheap," in *Proceedings of the 6th International Workshop on Peer-to-Peer Systems (IPTPS '07)*, Bellevue, Wash, USA, February 2007.

[31] B. Cheng, X. Liu, Z. Zhang, and H. Jin, "A measurement study of a peer-to-peer video-on-demand system," in *Proceedings of the 6th International Workshop on Peer-to-Peer Systems (IPTPS '07)*, Bellevue, Wash, USA, February 2007.

[32] J. Liu, S. G. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer internet video broadcast," *Proceedings of the IEEE*, vol. 96, no. 1, pp. 11–24, 2008.

[33] V. Janardhan and H. Schulzrinne, "Peer assisted VoD for set-top box based IP network," in *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Kyoto, Japan, August 2007.

[34] X. Y. Yang, P. Hernandez, F. Cores, et al., "DVoDP/sup 2/P: distributed P2P assisted multicast VoD architecture," in *Proceedings of the 20th International Parallel and Distributed Processing Symposium (IPDPS '06)*, Rhodes Island, Greece, April 2006.

[35] F. Picconi and L. Massoulie, "Is there a future for mesh-based live video streaming?" in *Proceedings of the 8th International Conference on Peer-to-Peer Computing (P2P '08)*, pp. 289–298, Aachen, Germany, September 2008.

[36] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: a comparative study of live P2P streaming approaches," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM '07)*, pp. 1424–1432, Anchorage, Alaska, USA, May 2007.

[37] S. Agarwal, J. P. Singh, A. Mavlankar, P. Baccichet, and B. Girod, "Performance and quality-of-service analysis of a live P2P video multicast session on the internet," in *Proceedings of the 16th International Workshop on Quality of Service (IWQoS '08)*, pp. 11–19, Enschede, The Netherlands, June 2008.

[38] M. Piatek, C. Dixon, A. Krishnamurthy, and T. Anderson, "LiveSwarms: adapting BitTorrent for end host multicast," Tech. Rep. UW-CSE- 06-11-01, University of Washington, Seattle, Wash, USA, 2006.

[39] http://www.allyourtv.com/, http://www.hollywoodreporter.com/.

[40] http://en-us.nielsen.com/.

[41] K. J. Kerpez, "Statistical variables for evaluating compatibility of remote deployments," ATIS standards contribution T1E1.4/2001-132, May, 2001.

[42] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system," in *Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference (IMC '07)*, pp. 1–14, San Diego, Calif, USA, October 2007.

Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 793591, 14 pages
doi:10.1155/2010/793591

Research Article

"Q-Feed"—An Effective Solution for the Free-Riding Problem in Unstructured P2P Networks

Sabu M. Thampi[1] and Chandra Sekaran K[2]

[1] *Department of Computer Science and Engineering, L.B.S Institute of Technology for Women, Kerala 695012, India*
[2] *Department of Computer Engineering, National Institute of Technology Karnataka, Surathkal, Karnataka 575025, India*

Correspondence should be addressed to Sabu M. Thampi, smtlbs@gmail.com

Received 2 July 2009; Revised 11 October 2009; Accepted 14 December 2009

Academic Editor: Yuanqiu Luo

This paper presents a solution for reducing the ill effects of free-riders in decentralised unstructured P2P networks. An autonomous replication scheme is proposed to improve the availability and enhance system performance. Q-learning is widely employed in different situations to improve the accuracy in decision making by each peer. Based on the performance of neighbours of a peer, every neighbour is awarded different levels of ranks. At the same time a low-performing node is allowed to improve its rank in different ways. Simulation results show that Q-learning-based free riding control mechanism effectively limits the services received by free-riders and also encourages the low-performing neighbours to improve their position. The popular files are autonomously replicated to nodes possessing required parameters. Due to this improvement of quantity of popular files, free riders are given opportunity to lift their position for active participation in the network for sharing files. Q-feed effectively manages queries from free riders and reduces network traffic significantly.

1. Introduction

A P2P network serves the content among the associate nodes rather than focussing it at a single central server. The barriers to starting and growing such systems are low, since they usually do not require any special administrative or financial arrangements, unlike with centralised facilities. P2P systems recommend an approach to aggregate and make use of the incredible computation and storage resources that otherwise just sit idle on computers across the internet when they are unused. P2P systems are widely used for file-sharing. The fundamental idea of file sharing is to utilise the idle disk space for storage and the existing network bandwidth for search and download [1]. A major benefit of P2P file sharing is that these systems are fully scalable—each additional user brings extra capacity to the system. In a P2P system, participating nodes mark at least part of their resources as "shared", allowing other contributing peers to access these resources. Thus, if node A publishes something and node B downloads it, then when node C asks for the same information, it can access it from either node A or node B. As a result, as new users access a particular file, the system's capability to provide that file increases [2].

There are mainly three different architectures for P2P systems: *centralized, decentralized structured,* and *decentralized unstructured*. In the centralized model, such as Napster [3], central index servers are used to maintain a directory of shared files stored on peers with the intention that a peer can search for the location of a desired content from an *index server*. On the other hand, this design makes a single point failure and its centralized nature of the service creates systems susceptible to denial of service attacks.

Decentralized P2P systems have the advantages of eliminating dependence on central servers and providing freedom for participating users to swap information and services directly between each other. In decentralized structured models, such as Chord [4], Pastry [5], and CAN [6], the shared data placement and topology characteristics of the network are strongly controlled on the basis of distributed hash functions. In decentralized unstructured P2P systems, such as Gnutella [7] and KaZaA [8], there is neither a centralized index nor any strict control over the network

topology or file placement. Unstructured P2P systems also called pure P2P systems are most frequently used in Internet. Nodes joining the network, following some loose rules, form the network. The resulting topology has certain properties, though the placement of objects is not based on any knowledge of the topology [9]. The decentralization makes available the opportunity to utilise unused bandwidth, storage, and processing power at the periphery of the network. It diminishes the cost of system ownership and maintenance and perks up the scalability.

P2P systems continue to grow according to recent measurement studies [10–13]. These studies show that the bandwidth consumed by the P2P file-sharing applications has exceeded that of the WWW applications. Most of the P2P file-sharing systems that rely on voluntary donations from individual participants potentially face the problem of free-riding. Free-riders utilise the resources of a system while contributing not anything to the system. Users who attempt to benefit from the resources of others without offering their own resources in exchange are termed free-riders [14]. The free-riders are selfish nodes which only utilize other peers' resources providing that none or limited contributions in return have greatly jeopardized the fairness attribute of P2P networks. Pure peer-to-peer systems are completely decentralized, resources are shared directly between participating peers, and the consequences of free riding are very terrible. In 2000, a measurement study of the Gnutella file-sharing network [15] found that approximately 70% of peers provide no files and that the top 1% of the peers provide approximately 37% of the total files shared. Similar patterns have been observed in subsequent studies of Napster and Gnutella networks [16]. A different study presented in [17] found that free-riders have increased to 85% of all Gnutella users.

Two different issues affecting the P2P system with free-riders are discussed in [18]. Due to free riding, the number of objects in the P2P system shrinks or grows very slowly. As number of popular files is decreased, the users' interest in the P2P system is drastically reduced so that the users eventually pull out of the system. When users who share popular files jump out of the system, the system turns out to be inferior in terms of the quantity of objects shared. This ultimately led to the fall down of the entire P2P system. The second issue is that the majority of query requests from different nodes are directed towards a few peers holding popular files. Hence, majority of the downloading requests are directed towards those overloaded peers causing network congestion. As the systems' main routine activities are seriously affected, such peers leave the P2P system gradually. Not only do free-riders deteriorate the quality of service for other peers, but they also intimidate the survival of the entire system. These issues point to the need for a rigorous mechanism for effectively managing the free riders in decentralised unstructured P2P systems.

This paper proposes a Q-learning-based approach for handling free riders in decentralised unstructured P2P networks. Thus, the proposed scheme-Q-feed eliminates the probabilistic or heuristic approach by a learning-based scheme. It utilises the past performance of nodes in the network for classifying a node as free rider or not. The proposed scheme mainly positions a neighbour of a peer in three states based on its performance. The idea is that the classification of neighbours based on their status restricts the amount of service being received from the neighbours so that the free riding behaviour can be greatly reduced. It also diminishes the services being provided by a peer to its neighbour. At any time a neighbour can move to higher status if it has shown improved performance. A node is punished for its free riding behaviour. The proposed autonomous replication scheme called Q-replication aims to increase the availability and fault tolerance in unstructured P2P network. The Q-replication autonomously replicates the popular objects to well-performing nodes according to their past performance.

In this paper, the terms "peer" and "node" are used in an interchangeable manner. The P2P system model comprises nodes and files (objects). There are a few neighbouring nodes $(n_1, n_2, n_3 \ldots n_n)$ associated with a peer "p." The peer classifies a neighbour as normal, suspended, or dormant, based on the service it provides or receives. The term "file" stands for any general content in a node or peer. A file can have more than one replica in the system. The number of neighbours connected (links) to a node is called its degree. The topology of P2P networks is modeled as a network with an undirected graph G whose nodes represent hosts and edges represent internet connections between those hosts. Nodes are usually very dynamic, where some can join and leave the network in the order of seconds whereas other nodes stay for an unlimited period. When a user requests a file, a search for the file is initiated and other nodes in the network need to be queried if the file is not available locally. A query is composed of one or more required words.

The remainder of this paper is organized as follows. Section 2 reviews the related work for controlling free riders. The proposed free riding solution is discussed in Section 3. An overview of the Q-replication technique is given in Section 4. The major steps of Q-replication are described as an algorithm in Section 5. Section 6 illustrates Q-replication by means of an example. Section 7 describes the experimental setup for conducting simulations. Section 8 discusses the results of experiments conducted. Finally, Section 9 concludes the paper.

2. Related Work

Though cooperation is a key to many P2P networks' survival and success, understanding it is difficult without efficient methods. To address this requirement, researchers have proposed several approaches to make P2P networks "contribution-aware" and thus combat free riding [19]. There are a few schemes cited in the literature for managing the free-rider problem based on incentives. Also there has been much research in P2P networks using a social structure to improve cooperation by providing good incentives.

A technique for managing the free-riders discussed in [20] considers a P2P network as a social structure where each peer behaves as a person in a society, making judgmental

decisions about other members in the society. The technique utilises the transfer of credit between peers to decrease the path length in queries. A selection strategy is proposed that involves different aspects of peer interactions in P2P networks. The credit transfer mechanism assists to deject mischievous peers by confiscating credits that they have with good peers and moving them to more cooperative ones. Peers that do not cooperate are eventually isolated from those that do cooperate. Further to that, the information about a credit transfer can be authenticated and verified.

In [21], a social network is used to model a P2P system. The P2P network is modelled using a directed graph, where the nodes are peers and the edges are connections between peers. There is a friendship between two peers which is represented by the directed edges in the graph. Each edge has a credit and a payment weight assigned to it, where the credit from one node to another is the payment from the other node to itself. The information about the data transferred between peers is used to describe the strength of the friendship. A balance of friendship is used in a decision function to determine routing paths. The algorithm is robust in a large population and relative high turnover rate. One potential problem for this incentive mechanism lies on the repeated transfer of data along the transaction path, as it may impose extra burden on system performance and network bandwidth if the mean length of path is long.

A completely decentralised system that allows efficient sharing of bandwidth in cooperative content sharing network is discussed in [22]. The system is called "Scrivener." It only requires nodes to track their neighbour's behaviour. It uses a greedy randomized routing algorithm to find a credit path, allowing a node to leverage credit it has with its overlay neighbours to obtain content from an unrelated node that holds the desired content. Scrivener is scalable and prevents freeloaders from exploiting obedient nodes.

In the simple modeling framework discussed in [23], a user in the network is a rational agent with a private and intrinsic characteristic called her type. Users make a decision whether to contribute to a system based on the association linking the cost of contribution and her type. This single parameter mirrors the readiness of the user to contribute its resources. If there are too few providers, then the deciding peer will be less eager to participate on account of the increased load on itself. The rate of contributing, to a peer, is the inverse of the total percentage of providers in the system.

The concept of utility function-based scheme to control free riding in a P2P file sharing system is introduced in [18]. This utility-based scheme creates incentives to motivate users to share interesting files. This method does not allow peers to download files if their utility value is lower than the size of the requested file. Three important utility factors for building fair incentives in the file-sharing context are identified: the total number of files shared, the total size of data shared, and the popularity of the data shared. The authors claim that the proposed scheme can increase the lifetime of the system by 10 times. However, the technique completely relies on the precise information on peers provided by the peers themselves. Hence, malevolent client programs can easily cheat the network and spoil the antifree riding measures.

A scheme that keeps track of the resource utilisation and resource contribution of each participating peer is proposed in [24]. In general the position of each participant in the system is denoted by a single scalar value, called their 'Karma'. Each node has an associated bank-set that keeps track of the node's Karma balance, which is an indicator of its standing within the peer community. The bank-set permits a resource consuming operation by the node only if the node has enough Karma in its account to allow the action. Karma does not require any centralized functionality or trust. Every time a peer's Karma changes, a predefined number of these peers should be reachable. Consequently, the ID of the peers should be known and not be transient. However, unstructured P2P networks do not maintain permanent and trustworthy identification techniques [25].

In a scheme proposed in [26], each node is associated with two parameters: money and reputation. Peers exchange money for service and increase their reputations while doing so. There is a central authority that settles disputes between peers when one believes it overpaid or did not receive enough service. The central authority is a set of randomly chosen nodes in the network. Similar to other schemes, they classify peers into three different types: honest, selfish, and malicious. As the solution relies on a centralised authority, its malicious behaviour will cheat other nodes connected to it. Also the dynamic nature of nodes in unstructured networks makes the technique an unreliable one.

There are different categories of methods for controlling free-riders. A classification of free riding techniques is presented in [19]. The schemes are categorized as monetary-, reciprocity-, and reputation-based approaches. Monetary-based approaches charge peers for the services they receive. Because these services are still very low cost, such approaches are also called micropayment-based solutions. The technique proposed in [24] is an example for monetary-based approach. The main disadvantage is that the proposed solutions require some centralized authority to monitor each peer's balance and transactions. This can cause scalability and single-point-of-failure problems. In reciprocity-based approaches, a peer monitors other peers' behaviours and evaluates their contribution levels. The well-known P2P application BitTorrent implements a reciprocity-based approach by adjusting a peer's download speed according to its upload speed. Reciprocity-based approaches face several implementation issues such as fake services published by peers. Since peer itself provides contribution level information, the credibility is in question. In reputation-based approaches peers with good reputations are offered better services. These approaches construct reputation information about a peer on the basis of feedback from other peers. Reputation-based approaches store and manage long-term peer histories. XRep [27] is an example of an autonomous reputation system. Reputation sharing is achieved in XRep through a distributed algorithm by which resource requestors can evaluate the consistency of a resource offered by a participant before beginning the download.

The distributed framework proposed in [25] primarily focuses on locating free-riders and taking actions against them. Each peer monitors its neighbours, decides if they

Table 1: Q-table of a node.

Node-id	N_1	N_2	N_3	N_4	N_5
Q-value	150	80	35	245	70
Status	Normal	Suspended	Dormant	Normal	Marked dormant state

are free-riders, and takes appropriate actions. The free-riders are classified as noncontributors, consumers, and droppers. This method does not need any interminable identification of peers or security measures for providing a global reputation system. Each peer just stores information about the neighbours' messages which are routed through it and executes the same kind of mechanisms alone and does not depend on any other peer's cooperation. The various counter measures proposed in the paper do not suggest utilising any score value for a peer's usefulness to the P2P system. One solution proposed to limit the network traffic is modification of TTL value based on the type of free-rider. Even for large TTL values, a simple mechanism of TTL reduction is employed and the reduced TTL value is always fixed for different categories of free-riders. The second type of punishment is dropping of queries originated from neighbours identified as free-riders. Another limitation of this technique is that the increase in availability of objects in the peers due to downloading and replication is not considered. Our technique reduces the value TTL of the low performing nodes based on its current TTL value. Also, the popular objects are replicated to well-performing nodes autonomously.

3. Q-Feed

The proposed solution "Q-Feed" is intended for reducing the free riding effect in unstructured P2P networks. It widely employs Q-learning concepts so that the accuracy in managing free-riders is improved significantly. Each node in the network maintains a few data structures called Q-tables. Q-tables contain a list of Ids of neighbouring nodes of a peer and their corresponding Q-values. It also contains the present status of a neighbour as normal, suspended, or dormant state. For each possible action, the Q-learning agent maintains a Q-value that indicates the efficiency of a P2P node in the past. A node in the network thus lives in any one of the three states. The status of a peer indicates the strength of a peer as a good node or not. A good node in the network is a well-performing node in the sense that it hosts more number of popular objects and it whole heartedly participates in the resource discovery process to improve the network performance. It never creates unnecessary traffic. The availability of the node may be high so that it can greatly serve other nodes in the network. In short, a good peer is altruistic and possesses a positive approach for serving other nodes. Hence, a node which is in the "normal state" is a trustable peer as long as its Q-value is within a certain level.

The status of a neighbour for the most part relies on the corresponding Q-value in a Q-table. A Q-table contains the ID of each neighbour along with its Q-value and status (Table 1). The Q-value of a node is modified using a few parameters, which are collected periodically. The parameters are collected based on various actions of nodes and outcome of these actions. A node in one status can move to the next lower state and vice versa when certain criteria are met. A node showing deteriorating performance first moves to the suspended state and then goes to the dormant state. But when the performance is improved, the node can lift its status. A suspended node can move to the normal state or a node in dormant status can move to suspended status after satisfying a few conditions. However, a much low-grade show of a node directs it to the dormant status. Contradictory to this degradation, a node possessing dormant status can also shift to normal status based on its greater performance. Marked dormant state is a special condition to be met by a dormant node to accomplish either suspended state or normal state. A free-rider peer is either in the suspend state or dormant state. The status thus explains the extent at which the free riding behaviour dominates in the peer. The access of network resources for such nodes is limited. Hence, a tight control mechanism is employed for dormant nodes than a node in the suspended state. A Q-value greater than or equal to 100 positions a neighbour in the normal state. A value less than 100 and greater than or equal to 60 shifts a neighbour to suspended state. A peer moves to dormant state if its Q-value in the Q-table is lower than 60. An example of a Q-table for a node is shown in Table 1.

Before initialising the Q-value for a neighbour in the Q-table, the peer P collects from its neighbour N the number of files present in the shared folder (f_{current}), the upload bandwidth (U_{band}), and download bandwidth (D_{band}). Using these values, the initial Q-value is computed as $[(u_1 * (u_{\text{band}}/D_{\text{band}}) + u_2 * f_{\text{current}}) * 100]$, such that $u_1 + u_2 = 1$ where u_1 and u_2 are two weights, and $u_2 > u_1$ to give higher value for the second component in the equation. Thus, based on the number of files shared and values of upload and download bandwidth, an initial Q-value is assigned for the neighbour in the Q-table of P. For higher upload, a larger initial Q-value is assigned. Larger upload bandwidth allows a node for the faster sharing of files hosted by it to other nodes in the network. At the same time, the presence of large number of files in the shared folder increases sharing of files a large extent. The initial value is computed when a P2P node joins the neighbour group of a peer. However, in this work, a minimum Q-value of 100 is assigned for all the neighbours possessing initial Q-value less than 100 to acquire normal status; otherwise the nodes may be directly shifted to other states based on initial Q-value. The assignment of initial value gives opportunity to a node to show its performance and thereby receives rewards for the actions.

The entire process of status transformation of a node in the P2P network is depicted in Figure 1. The different techniques that are followed by each node in different states for managing free-riders are explained in the following parts of the paper.

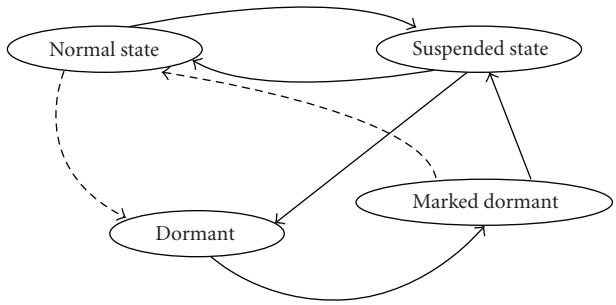


Figure 1: A neighbouring node of a peer in different states.

3.1. Normal State. In the normal state, a peer does not enforce any restrictions for selecting a neighbour for forwarding the queries. A node in the P2P network is initially in the normal state. Several performance parameters are collected continually and based on these a node either stands in the normal state or moves to lower state. The chief deciding factor for changing the status is the Q-value of the neighbour in the Q-table of the peer. A neighbouring node in the normal state should always maintain its Q-value greater than a threshold value L_{th}. Otherwise, the node is degraded to lower status. For a particular period of time "t", a peer of a neighbour collects the values of parameters listed in Table 1 and computes the reward for all the actions in this period (1). The reward is used to modify the Q-value of a neighbour N of a peer P (2). The value of learning rate constant α is preset at 0.2. Such low value assigned to learning rate constant provides gradual increase of Q-value. The Q-value is modified using the computed reward and current Q-value. Higher the reward, an increase in Q-value of a neighbour in the Q-table will occur. The sum of all weights is equal to one such that $w_1 > w_2, w_3, w_4$. A large value for w_1 provides high priority for number of hits occurred in neighbour and number of results produced. The more number of hits near the peer decreases the number of hops to be travelled by a query:

$$\rho = \left[\left(w_1 * \left[\frac{\text{PNHIT}}{\text{PN}}\right] * \text{AvgR} + w_2 * \left[\frac{\text{ONHIT}}{\text{ON}}\right] + w_3 * \left[\frac{\text{NA}}{(\text{NR} - \text{NA})}\right] + w_4 * \left[\frac{\text{NGHIT}}{\text{NGQ}}\right]\right) * 100\right], \tag{1}$$

$$Q_{i,t+1} = Q_{i,t} + \alpha(\rho - Q_{i,t}). \tag{2}$$

For all the neighbours, P maintains the values for all the variables listed in Table 2. The variable PN contains the number of requests generated by a peer P to a neighbour N for the specified period. P receives messages from other neighbours and sometimes these messages may be forwarded to N. The number of such messages is stored in a variable called ON. Similarly, the number of query hits for messages PN and ON queries is recorded. These are represented as PNHIT and ONHIT, respectively. A node specialised in a particular area responds with more number of results for a query matching the area of specialisation. Hence, the number of results for queries submitted by P to N is monitored and at the end of the period, an average of number of results for all the queries submitted by P to N is computed (AvgR).

Because Q-replication algorithm replicates copies of objects to nodes in the network, peer P may also receive and replicates objects. The number of files replicated during "t" by P to N is recorded (NR). The small portion of secondary storage is spared by a node for hosting shared files for the P2P network. All the nodes in the network can access these files by a resource discovery process. A free-rider node may intentionally remove the replicated files from its shared folder to save network resources. To verify this, P retrieves the number of replicated files present at N and the value is stored in NA. A neighbour also dispatches its own query messages (NGQ) to P and the number of hits for those queries is observed (NGHIT). Using the modified Q-value the status of neighbour as normal, suspended, or dormant is decided. If the Q-value is less than a low threshold value, the neighbour node is shifted to *suspended state* and the extreme reduction of Q-value value moves the node to *dormant state*.

3.2. Suspended State. Assume that a maximum TTL value is preset by the P2P system for all the nodes in the network. A neighbour N of a peer P is degraded to suspended state when the corresponding Q-value for the node in the Q-table of peer P reaches a value less than a threshold value, that is, $Q_{i,t} < L_{\text{th}}$. For the queries received from suspended node, the peer P checks its shared folder for a query match and if the query is not satisfied at the peer, the current TTL value is decreased and the query is forwarded to a neighbour of the peer. The TTL value is modified as $\text{TTL}_x = \text{round}(\ln(\text{TTL}))$, $\text{TTL}_x \geq 0$, where ln(TTL) is the natural logarithm of current TTL value and TTL_x is the modified TTL value which will be always less than current TTL value. Because the low TTL value causes reduction in network traffic, the messages associated with suspended nodes are significantly reduced.

The peer P records the number of queries originated from the suspended peer (n_1) for a certain period of time. If the value of n_1 is greater than a certain limit, Q-value for the suspended node in the Q-table of P is modified using reward computed. The reward calculation utilises number of files present in the free-rider, f_{after}, and the number of files present at the time of suspension, $f_{\text{free}} \cdot f_{\text{after}}$ includes number of the old files as well as newly added files in the suspended node. The data is collected by sending a message to the suspended node. The reward is computed as $\delta = (w1 * (f_{\text{after}} - f_{\text{free}})) * 100$ where w_1 is a weight and its value is between 0 and 1. The high value, which is assigned to w1, makes the reward high. This provides high priority to the increased availability. The Q-value of the suspended free-rider is modified as $Q_{i,t+1} = Q_{i,t} + \alpha(\delta - Q_{i,t})$, where α is the learning rate constant [28]. Due to this amendment, the Q-value is increased/decreased slowly for the free-rider based on number of new files being hosted in it. Thus, a node which is placed in the suspended state can gradually move to the normal state or lower state. The moment a neighbour moves

Table 2: Parameters collected from a neighbour in the normal state.

PN:	Number of query requests originated from a peer P to a neighbour N
ON:	Number of query requests received from other neighbours and forwarded by P to neighbour N
PNHIT:	Number of query hits from neighbour N for queries originated from peer P
ONHIT:	Number of query hits from other neighbours for queries forwarded by P to neighbour N
NF:	Number of files (objects) hosted in each neighbour
NR:	Number of files replicated to a neighbour N by P
NA:	Number of replicated files actually present at time t in neighbour N
AvgR:	Average number of results per query for the requests sent to N for the period
NGQ:	number of queries originated from neighbour N towards P
NGHIT:	number of hits for queries originated from N through P (including hits at P)

to the suspended state, the peer P stops sending messages to it.

3.3. Dormant State. Immediately, the Q-value in the Q-table for a neighbour reaches a threshold value U_{th}; that is, $Q_{i,t} < U_{\text{th}}$, the neighbouring node is despoiled to the dormant state. Hence, the peer for which the node is a neighbour allows the free-rider very limited access to the network resources. The value of U_{th} is always far less value than L_{th}. Any queries originated from the free-rider neighbour and the queries routed by it are processed by the peer and the queries are not routed further. If the required object is found, the querying node is informed. Otherwise, the queries originated/routed from the free-rider neighbour are dropped. However, the dormant nodes are low performing nodes because of low object availability and these nodes consume undue network resources by sending messages to other nodes for resource retrieval. The heavy flow of incoming query messages increases the load of a peer and thus the performance of the node may be seriously damaged. In this situation, the load on a peer should be effectively managed. The CPU usage of the peer is measured and if it reaches a maximum CPU-load, the queries arriving from dormant nodes are discarded. The maximum CPU-load threshold value is preset by the peer itself.

The major blow to a free-rider is that the moment a node is declared as dormant, the peer for which dormant node is a neighbour stops sending further query requests to it. Hence, once the node demeans to the dormant state, the peer of the node does not further utilise the resource discovery feedback for manipulating the Q-value of the dormant node in its Q-table. A node in the suspended state is more trustable than a node in the dormant state. Hence, it can simply shift to the normal status by just increasing the availability. But, the dormant node shows very low performance in file sharing because of its low performance shown earlier and so rigid policies should be followed for managing its status change. A node in the dormant state can move to suspended state or normal state if the present Q-value is increased to appropriate levels. This is done in two stages: the first stage employs a polling process and the second stage utilises both the object availability and the results of resource discovery.

In the first stage, the peer of a dormant neighbour does polling among other neighbours which are in normal status. The polling is done after a period of time preset by the system and it is a continuous process. Only well-performing nodes with higher Q-values are merely considered for the polling process. For polling, the average of Q-values of neighbours other than free-riders listed in the Q-table is computed (AvgQ). The neighbouring nodes whose Q-values greater than or equal to AvgQ are selected for conducting the poll. The peer dispatches messages to the chosen neighbours to gather the corresponding Q-value of the dormant node in their Q-tables. If the dormant node ID exists in the Q-table of a chosen node, the Q-value of the node is dispatched to the peer. A peer discards the message if id of the dormant node does not exist. The peer computes the average of the Q-values (AvgD) returned. If AvgD $\geq L_{\text{th}}$, dormant node is *marked* as a suitable candidate for moving to the suspended state. In this scenario a dormant node is said to be in *marked dormant state*. The present Q-value of the marked dormant node in the peer's Q-table is modified with the value of AvgD, that is, $(Q_{i,t+1} \leftarrow \text{AvgD})$. Even though the Q-value after modification is greater than L_{th} or U_{th}, the node remains live in the dormant state until certain conditions are met. This is to ensure that the node hosts some useful objects. If none of the selected node has the particular dormant node as neighbour, the Q-value of the node is not altered. The presence of higher Q-values for the node on other neighbours indicates the good service it may provide to other peers. At the same time, nodes may deliberately collaborate to augment their Q-values. Thus, the result of polling may provide a Q-value which is ambiguous. To generate a feasible solution to overcome this limitation, the increase in availability of the objects and the quality of the objects are taken into account. The quality of the objects is to be considered as the node may attempt to increase the availability by hosting very low popular objects in its shared folder causing very low success rate. The high success rate for the queries inputted through suitable resource discovery mechanism reflects the quality of objects.

The second stage involves a two-step process for the marked dormant neighbour for promotion to higher states. The first step is connected to the increased availability of shared objects. The second step is mainly intended for checking the usefulness of objects hosted in the dormant node. The marked dormant node can thus budge to the suspended state based on *increased availability of objects,*

and number of query hits. The Q-replication relies on the node parameters such as bandwidth, degree of the node, and free storage. There is a possibility that other nodes in the network running the autonomous Q-replication algorithm may replicate objects to the dormant node. The dormant node may also download objects from other nodes in the network after successful search operations. Thus, the availability of the objects may be reasonably improved. Hence, like in suspended state, the peer right now collects the number of shared objects available (F_a) in the shared folder of the dormant node. The number of objects existing in the node (F_d) at the time of degradation to dormant state is recorded earlier. The difference (D_{da}) between F_d and F_a is computed. The number of objects downloaded between the current time and the time at which the node is declared as dormant may be very low or nil in the shared folder of the node if the node exhibits free riding status for which the node is a neighbour for other peers. As mentioned earlier, the incoming queries from dormant nodes are not forwarded favourably to other nodes by the peer. If $D_{da} \geq [F_a + \ln(F_a)]$, the dormant node begins to receive limited number of queries from the peer for which the dormant node is a neighbour. The second component $\ln(F_a)$ ensures that nodes which are hosting additional objects other than those present at the time of degradation to dormant state are also chosen for the second stage of processing.

In the second step, not all the queries that are originated from P or forwarded by P towards the dormant node are processed by P. The Q-value of the marked dormant node is moderately higher due to modification with AvgD. For testing the quality of objects hosted in the marked dormant node, a small number of queries are routed periodically (nth query 20th, 40th, 60th…) by the peer. All the query hits happened in the dormant node are recorded. Hit ratio for number of queries is computed as a ratio between the number of hits and number of queries being processed. If the hit ratio (H_r) is higher than a threshold hit rate (H_{th}), the marked dormant node moves to the suspended state; otherwise, the process is repeated until the required hit rate is achieved. Thus, through multifaceted actions, a dormant free-rider modifies its status.

4. Q-Replication

Replication addresses one aspect of the free riding problem [18]. The objective of a replication technique is to improve availability and enhance system performance. Since a small number of nodes host popular objects, these nodes may receive large quantity of query messages from other nodes. A solution to increase the availability of objects in the network is replication. It makes use of the disk space and the network bandwidth resources of the node downloading a particular file. Thus, more number of nodes satisfy downloading requests from several nodes. This helps to reduce network congestion and CPU overloading problems that the small number of peers experiences.

P2P-based replication strategy is the topic of various active research projects and most of these projects emphasize replication in structured P2P systems. Only a few replication techniques are cited in the literature for decentralised unstructured P2P networks. Merely a few replication techniques for unstructured P2P networks are cited in the literature. Most of the replication schemes are linked to the search techniques being employed; that is, objects cannot be replicated to further than the nodes on the search path. In all cases, the behaviour of nodes is not taken into account while replicating the objects. Hence, to increase the availability and to replicate the objects autonomously, a novel replication technique is proposed in this work. Our scheme, known as "Q-replication", employs Q-learning for the autonomous replication of objects [28–30]. It is autonomous because the decision to replicate an object to appropriate sites is taken autonomously by a node based on the past performance of peers in replicating objects. Thus in spite of constant changes to the connection, objects are highly available. Like Q-Feed, the proposed replication technique maintains a Q-table which contains peer-IDs and corresponding Q-values of each peer. A Q-value represents how a peer has contributed to the replication activities in the past. As part of replication, a node receives a reinforcement signal from the target node which is intended for hosting the replica. The signal is translated into a reward. Parameters such as bandwidth, degree of the node, and storage cost are utilised. The Q-values are updated appropriately. The Q-replication process selects the target objects for replication based on their popularity. The popularity is computed in a unique way. After replicating an object, the Q-values corresponding to the nodes in the Q-table are updated using the various parameter values returned. A node which goes down frequently and maintains low values for bandwidth, degree, and available storage may produce small value for reward. These nodes show less performance in the replication process. Hence well-performing nodes receive high Q-values and the Q-values of nodes with low performance are reduced further. A shared object is replaced from a node to accommodate a new object by utilising the popularity and the time at which the object was inserted into the shared directory of a node. The Q-replication, thus, distributes the popular objects to well-performing nodes in the network for improving the availability of objects and thereby contributes to the improvement of success rate and fault tolerance without depending on search paths.

The Q-replication scheme is distributed and employed without the coordination of centralized servers. It considers the replica selection problem (which data to replicate) and the replica placement problem (where to place them) and provides simple solutions to them. The replica selection problem deals with a suitable criterion for selecting an object from the shared storage space of a node for replication. The replica placement problem addresses the process of choosing an appropriate node for hosting a replica.

4.1. Selection of Objects for Replication. The objects are chosen for replication according to their popularity. The frequently accessed objects from the shared storage space of a node are treated as popular objects. These files are ranked according to their popularity. The details are stored in a table,

which contains the object name along with its rank and the status of replication. The value of the rank represents the popularity of the object. The high-value rank denotes a most popular item. The status field facilitates to identify already replicated files in a node. The system regularly (e.g., for every 50 requests received by a node) updates the popularities of all the objects in the nodes based on incoming queries for that period. The popularity update process $P_f(t+1)$ at a time t relies on the number of requests received for the object $R_q(t)$ and the total number of requests received by the node $N_q(t)$ up to that period after the previous update and the present value of popularity $P_f(t)$. The popularity is modified as $P_f(t+1) = P_f(t) + \eta[(R_q(t)/N_q(t)) * 100]$ where the value of P_f is greater than or equal to 0 and the value of constant η is in between zero and one. The values thus modified are written into a table (*popularity table*) after removing the existing values. The update equation shows that the update process also utilises the existing popularity value for modification. The initial value of P_f for an object is always zero. If the number of queries received for the time period is nix, the popularity of the object is not altered. Otherwise, the popularity of the file increases with the number of queries being received. For every δ period, the system identifies the possible candidates for replication. This is done by comparing the popularity of a file $P_f(\delta)$ with a threshold popularity value (P_{th}). When the popularity of a file at δ becomes greater than or equal to the threshold value, that is, $(P_f(\delta) \geq P_{\text{th}})$, the process of selecting the target nodes for hosting the replica is initiated.

4.2. Q-Table Creation and Initialisation. The target nodes are selected from the Q-table. The members for the Q-table are assigned after a simple operation: a message (Hello message) is sent to nodes that come within a time-to-live (TTL) limit, which is the number of hops the message should be propagated; the responded nodes become members of Q-table with some initial Q-value. The message forwarding follows a k-random walk [9] procedure. Initially K messages are generated and the messages are propagated through K number of neighbours selected randomly. Neighbouring nodes forward the message to one of their neighbours, from there to next hop. The message has a message-id. Nodes, which have already received a copy of the message, keep the message-id and address of the neighbouring node to which the message was forwarded. Hence, when a node receives the same message another time, it will not be forwarded to a node that has received the message previously but selects a different peer from the neighbour list. The response messages from the peers consist of equivalent values for their current bandwidth (b_w) and available storage (s_{avbl}). Using these values, Q-tables are initialised. The P2P system assigns minimum values for node attributes such as bandwidth ($b_{\min}$) and storage ($s_{\min}$) , which are used for Q-value computation. The Q-values each node in the table is initialised as $Q_r = (b_w/b_{\min} + s_{avbl}/s_{\min}) * 100$. In order to eliminate the random or probabilistic assignment, Q-values are thus initialised with important node attributes bandwidth and storage.

4.3. Selection of Nodes and Replication. A good peer, which can host a replica, should have a high-speed connection, minimum available storage, link with more number of nodes, and it should stay online for a long period. From the possible set of host candidates listed in the Q-table, the best ones according to the bandwidth, available storage, and number of links (degree) are chosen. The objects are copied into nodes, which do not already host the same replica of the target file. Hence, the overwriting of the same file in a node is avoided and at the same time, the process saves bandwidth consumption due to redundant file transfer. In order to choose the possible candidates for hosting the replica, the mean of Q-values listed in the Q-table is computed. Nodes with Q-values greater than or equal to the mean (AvgQ) are selected and a message is sent to each selected node to verify whether a copy of the object exists in their shared folder. *Replication List* of a node is a table that contains a list of object names reserved by other nodes during the object checking process. This evades other nodes to replicate the same object to a node as the same node may be chosen by another node as a target node for replication. If the node is not up, a copy of the object is present, or the object's name appears in the replication list, the node is left out from replicating the chosen file. All other nodes, whose Q-values greater than or equal to AvgQ, are selected as target nodes for hosting replicas.

4.4. Reward Computation. The nodes, which received a copy of the file, send the values for degree (d_d), bandwidth (b_w), and available storage (s_{avbl}), after storing the replica to the node that initiated the replication process. This is the reinforcement signal to the replication system. Based on the reinforcement signal, the reward (ρ_i) is computed for each node in the Q-table as $\rho_i = [([d_d/d_{\min} * w_1] + [b_w/b_{\min} * w_2] + [s_{avbl}/s_{\min} * w_3]) * 100]$, where $w_1 + w_2 + w_3 = 1$. As the bandwidth is a very important network resource, priority is given for it while computing the reward, hence $w_2 < w_1, w_3$. Therefore, the nodes with large bandwidth highly influence the reward. Moreover sufficient storage space should be available in a node for hosting more and more replicas of different objects. In a P2P network, a few nodes have a large number of degrees while most of other nodes have only a small number of degrees. Peers with a large number of degrees make many replicas as peers with a small number of degrees [31]. In addition, replicas on large degree peers are used frequently as those on peers with small degrees. In our strategy, the system assigns a common minimum degree threshold ($d_{\min}$) value to be used for replication to all nodes. In terms of *degree,* the contribution of high-degree nodes to the reward is high as compared to low-degree nodes. At the same time, nodes with only high bandwidth and storage can also participate in the replication process. All these factors ensure the availability of objects within short hop distances.

4.5. Q-Table Update. The reward values are utilised to modify the Q-values (Figure 2). The update process increases, or decreases the Q-values of peers that are being participated

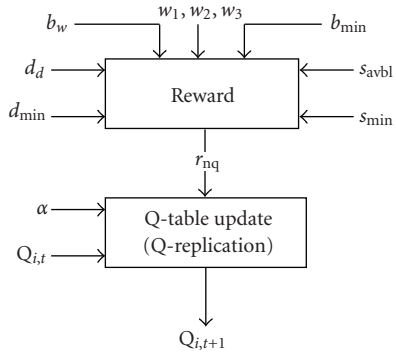


Figure 2: Q-table update (Q-replication).

in the replication process. The nodes, which have not participated, do not modify the present Q-values. The nodes with high Q-values are treated as good peers. The Q-values of nodes, which have created a replica, are updated as $Q_{i,t+1} \leftarrow Q_{i,t} + \alpha(\rho_i - Q_{i,t})$, where α is the learning rate (value of α between zero and one), and $Q_{i,t}$ is the present Q-value. If the reward of replication is high, the Q-value is incremented and it relies on bandwidth, available storage, and degree of the node. The current Q-values are retained for the nodes comprising a copy of the object, that is, $Q_{i,t+1} \leftarrow Q_{i,t}$. Nodes that are not up are punished heavily with zero reward, $\rho_i = 0$ and the Q-values are updated as $Q_{i,t+1} \leftarrow Q_{i,t}(1 - \alpha)$. Assigning high value to the learning rate constant yields a large increase in Q-values of nodes that have placed a replica to their respective directories.

4.6. Object Replacement. Some replicas should be deleted to make space for new replicas if adequate storage space is not available in a node. Our replication scheme removes the objects according to their popularity and age. The age attribute represents the time at which the object was inserted into the directory. If the object is recently added to the shared directory, it may have low popularity value and small value for age. Hence, objects with low popularity values and large values for age are removed for housing new objects.

5. Q-Replication Algorithm

Here we present the entire algorithm for Q-replication for implementation. The Q-replication algorithm chooses the popular objects for replication whenever the popularity of the objects reaches certain threshold. The peers, which are n hops away, are included in a Q-table along with their past performance represented as Q-values. The target peers for hosting the replicas are selected from the peers listed in the Q-table according to the Q-values of nodes. Peers with Q-value greater than or equal to the average Q-value are chosen as target nodes. The shared directory of each chosen peer is searched for the presence of the selected object, and based on the result, the object is replicated. The replication process returns reinforcement signals comprising available storage, bandwidth, and degree of the node. The reinforcement signal is converted into a reward. The Q-values of nodes in the Q-table are updated using the computed reward values. Finally a list of nodes which have received a copy of the replica is displayed. Q-replication is a greedy algorithm with a worst-case time complexity of o(n), where "n" is the number of nodes in the network.

Algorithm 1. Input: Object 'f' for replication; Q-table.

Output: a list of nodes, which have received a copy of the replica.

(1) Select an object f for replication based on its popularity.

(2) Compute the average of Q-values corresponding to Q-table of node x - AvgQ.

(3) For each entry Ti in the Q-table, select nodes with Q-values >= AvgQ.

(4) For each selected node in step 3, check for the presence of the object.

(5) If f exists in the searched nodes or node is not up or the object's name is in the *Replication List*, leave out nodes from replication process

else

(5.1) Select the remaining nodes with Q-values >= AvgQ for replication.

(5.2) Insert the object's name in the *Replication List* of *the* selected nodes.

(6) For each chosen node N:

(6.1) Replicate the object from source node to target node.

(6.2) Remove the object entry from the *REPLICATION LIST* of the node, which has received the replica.

(6.3) Wait for reinforcement signal.

(6.4) Receive reinforcement signals—available free storage after storing the file, bandwidth, and degree of the node.

(6.5) Compute the reward using the parameter values: $\rho_i = [([d_d/d_{\min} * w_1] + [b_w/b_{\min} * w_2] + [s_{\text{avbl}}/s_{\min} * w_3]) * 100]$.

(6.6) Update the Q-value of node, which has received a replica as $Q_{i,t+1} \leftarrow Q_{i,t} + \alpha(\rho_i - Q_{i,t})$.

(6.7) Nodes with a copy of the object, which are excluded in step 5, do not alter their Q-value.

(6.8) Nodes that are not up (step 5) accept zero reward and update Q-values as $Q_{i,t+1} \leftarrow Q_{i,t}(1 - \alpha)$.

6. Applying Our Approach (Q-Replication)

This section illustrates the autonomous replication technique with a simple example. The results shown in the

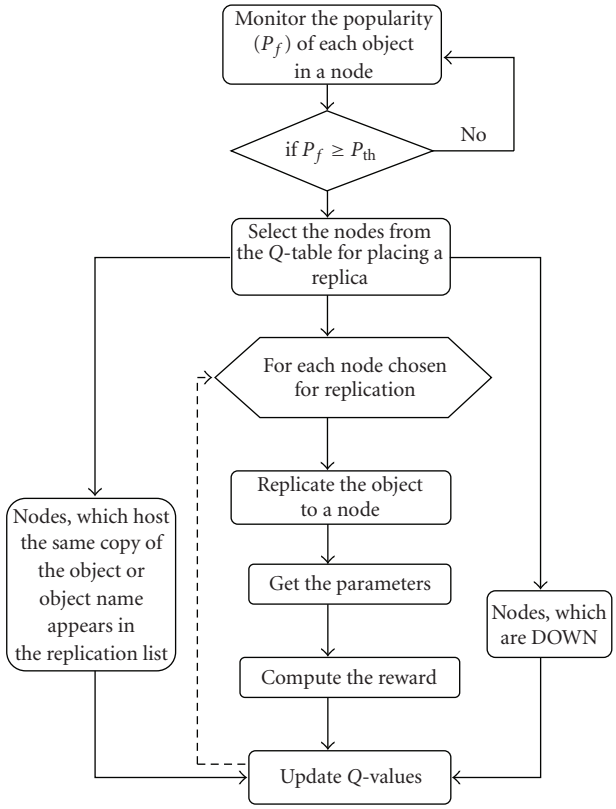


Figure 3: Set of actions taken by a node for replicating an object.

example are only for illustration purpose only and it does not reflect the situation in a real network. A simple P2P network in a graphic representation is shown in Figure 4. The neighbours of node A are nodes B, C, and D. A "HELLO" message is sent with two walkers to nodes within three hops away. The walker messages are fired through the neighbours B and D. From the second hop onwards, only one neighbour of each node is inserted into the replication Q-table. As a result, the Q-table (Table 3) of A contains neighbouring nodes as well as the nodes H, N, E, and F. Based on the attribute values, the Q-table is initialised. The Q-values are modified after each replication action.

Consider a scenario in the network after a few replication operations. There is an object "f_1" available for replication at node "A". Assume that the object is present in none of the nodes listed in the Q-Table, and all the nodes are "up". The average of Q-values is found (AvgQ = 364) and the target nodes are selected from the Q-Table (Table 4). The nodes B, D and N have the Q-value greater than or equal to AvgQ. However, the node "D" is not up; hence, it receives a negative reinforcement. The values of various attributes are collected and the rewards are computed. The value of α is preset as 0.2. The object is replicated to the nodes B and N (Figure 5). The Q-values are updated according to the update policy. The modified values in the Q-table of node A are shown in Table 4. The update operations increase the Q-value of both the nodes B and N. As the status of node D is "down", its Q-value is diminished. All other nodes, which have not participated in the replication, keep the values as such.

Table 3: A Replication Q-Table.

Neighbours and nodes 3-hops away	B	C	D	H	N	E	F
Initial Q-values	482	568	345	234	132	324	185

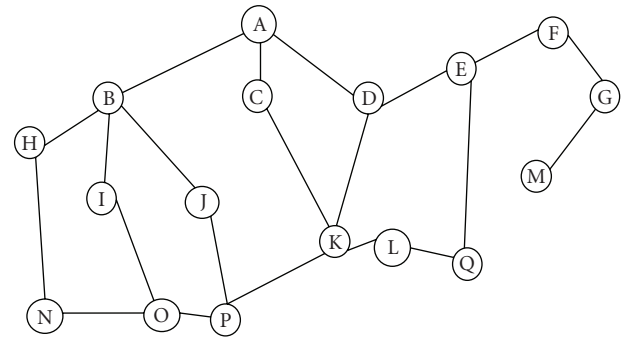


Figure 4: An Unstructured P2P Network.

7. Experimental Setup

For improving the resource discovery process, popular objects are distributed to different nodes in the network. Hence, the efficiency of Q-feed relies significantly on Q-replication as the status change of a node is very much linked to the availability of popular objects in well-performing nodes. The proposed algorithms are simulated using random graphs that have 10000 nodes. The nodes can join the network and establish random connections to existing nodes. There are multiple copies of 3000 objects randomly distributed to all the nodes in the network. Since the files are assigned randomly, the number of files in each node is different. The average degree of a node in the network is 3.5. We assume that all categories of nodes after the configuration of the network are formed due to several conditions preset in the proposed algorithms. The popular objects are replicated to different sites using autonomous replication algorithm. The bandwidth and space for shared storage are assigned to each node randomly from a list containing possible values. The quantities of objects maintained in the system are sufficient to analyse the performance since autonomous replication scheme effectively propagates the objects to various sites. The objects are word, PDF, and text files available as course materials on various subjects such as computer science, electronics, physics, mechanics, and electrical. Thirty thousand keywords are chosen from the course material files and these words are randomly selected as query keywords by all the nodes during searching. Two types of query search are employed: *file name based and keyword based*. In the file name-based search only the objects names in the shared storage space of each node are searched. The objects containing the keywords are looked up in the keyword-based search. In the simulation scenario, all the queries contain keywords alone. Because the files are randomly distributed, the number of files in a node varies form another node. This number will change due to successful downloads and replication. In the beginning of simulation all nodes have

Table 4: Status of Q-table of node "A" before and after replication.

Neighbours and nodes within 3-hops away		B	C	D	H	N	E	F
Q-values after a few operations		682	122	441	336	466	324	175
Parameters	Storage, s_{avbl} $s_{min} = 40$	75	45	60	63	65	69	47
	Bandwidth, b_w $b_{min} = 64$	98	42	71	77	92	73	32
	Degree, d_d $d_{min} = 3$	3	2	3	1	2	3	2
Node Status		up	Up	down	up	up	up	up
Reward ($w_1 = 0.4$, $w_2 = 0.2$, $w_3 = 0.4$)		1485	—	0	—	1292	—	—
Updated Q-values		843	122	353	336	631	324	175

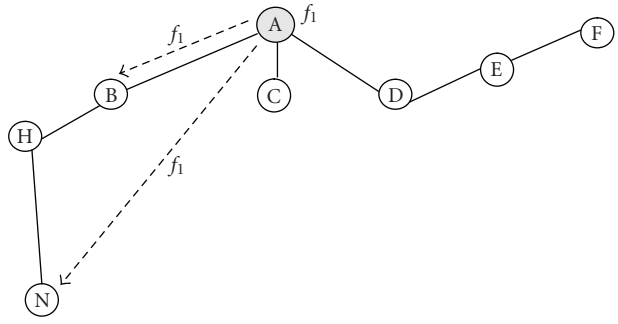


Figure 5: Replication of object "f_1" to nodes B and N.

the same status. Based on several parameters, a node moves to different status to reflect their contribution level in the network.

The "k-random walk" [9] search technique is employed for resource discovery in the network. When a user enters a query, the shared folder of the node is searched for the presence of the required object. If object is not present, the query is further forwarded to the K number of walkers. From there onwards queries are forwarded only through one neighbour of a node in the search path. Neighbours which are in 'normal status' are the candidates for participating in the search process. However, queries are also propagated through marked dormant node according to the prescribed criteria. A counter records the number of queries being processed by a node. The search is terminated either result is found or time-to-live (TTL) is expired. The default TTL value is preset as six. The number of walkers for a query operation is fixed as six. If sufficient number of neighbours are not present, all the neighbours are selected without bothering about the maximum number. Each node generates 100 queries and one query is propagated every 20 seconds on average. However, each node enters the query generation phase in a randomly selected time slot. Hence, the flood of query message production is regulated. One by tenth of simulation time is utilised for each period of operation. Eighty percent of the nodes are up at the time of performing simulation. Fifty percent of "Down" nodes selected randomly change their status to "UP" after every 50 000 queries are propagated and, at the same time, the same amounts of UP nodes obtain the DOWN status.

The simulation tool has been developed using Java language. The tool runs in a Windows operating system environment. The software, which is used for developing the simulation software, is NetBeans, J2SE Development Kit 5.0 and WampServer. NetBeans is a free, open-source Integrated Development Environment, which supports development of all Java application types. WampServer is an open source project and Windows web development environment. It allows creating various applications with Apache, PHP, and the MySQL database. WampServer also includes PHP-MyAdmin and SQLiteManager for managing databases. The simulations are conducted in systems with Intel xeon (Quad Core) processor,12 MB L2 Cache, 1333 MHz FSB, 4 GB, and 146 GB SAS HDD (15 K RPM).

8. Results and Discussion

The experiments are conducted by running the simulation tool number of times. Data for different parameters are collected each time. We have not intentionally made any neighbour of a peer as free-rider in the network. A few performance metrics are employed to evaluate the performance of the proposed techniques.

Number of simulation runs versus % of nodes in different states: all the nodes in the network are linked to one or more neighbouring nodes and the details about these nodes are maintained in the Q-table of a peer. One neighbour of a node may be a neighbour of several other nodes in the network. Hence, the same node may possess different status in Q-tables of other peers. The status of all the neighbours of a peer in different categories as normal, suspended, and dormant is counted after each simulation run. This is repeated for all the peers in the network. The number of times a node exists in each status during a simulation run in the whole network is counted and the status with highest value is selected for further operations. Based on the values so computed, the percentage of nodes in different status is worked out for all runs. The data is collected from only the nodes which are up at the end of each simulation.

The situation for nine runs is plotted and shown in Figure 6. The data in the previous run is kept as such for

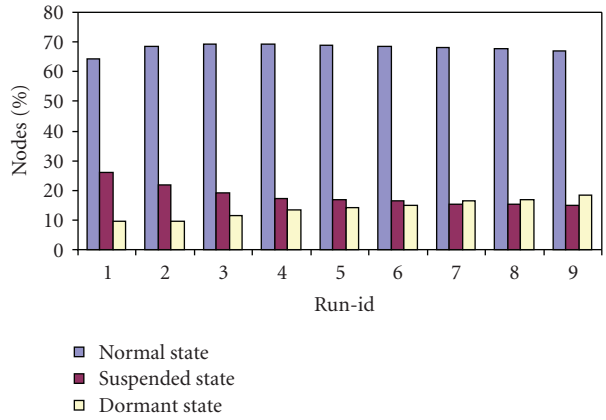


Figure 6: The nodes in different status.

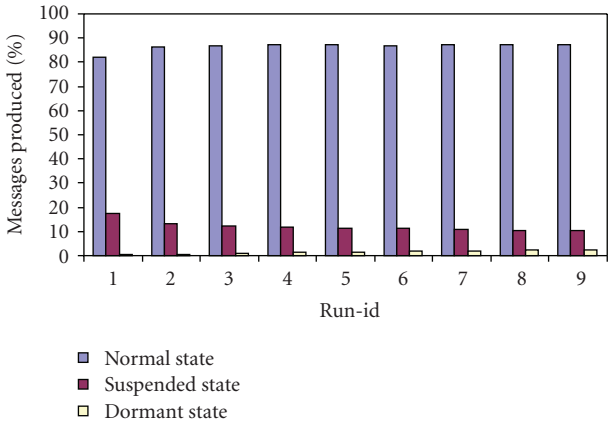


Figure 7: Messages produced for each run by nodes in different status.

the next run. The free-riders are formed during simulation without any external intervention. So, based on the performance of neighbours of peers, the nodes with different status are created. Because of Q-learning, the network takes time to identify the free-riders in the network and each node in the suspended state or dormant state is allowed to improve their status according to the outcome of their actions. As shown in Figure 6, after finishing five runs, the number of nodes possessing the normal status is going in a steady position. But the number of nodes in suspended state is slightly decreased and the number in the dormant state is increased slightly. Most of the low performing nodes are identified within a few simulation runs. This is due to the knowledge the nodes acquired through Q-leaning. The advantage is that a peer gradually identifies well-performing neighbours based on their Q-values and status for future operations. Further to that, Q-replication increases the availability of popular objects by replicating objects to nodes satisfying certain conditions stipulated by the replication algorithm.

Number of Simulation Runs Versus % of Messages Produced in Each Category of Node. Each peer records the number of messages received from each neighbour of the peer and the

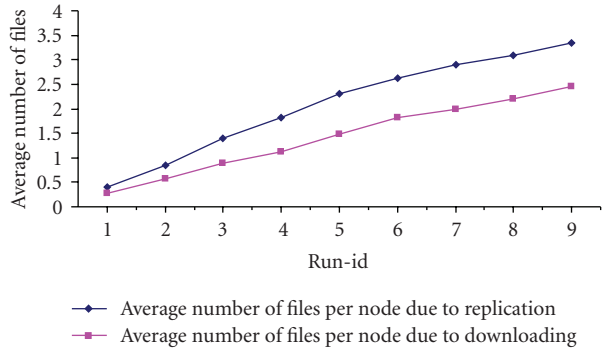


Figure 8: Number of files due to Q-Replication and Downloading.

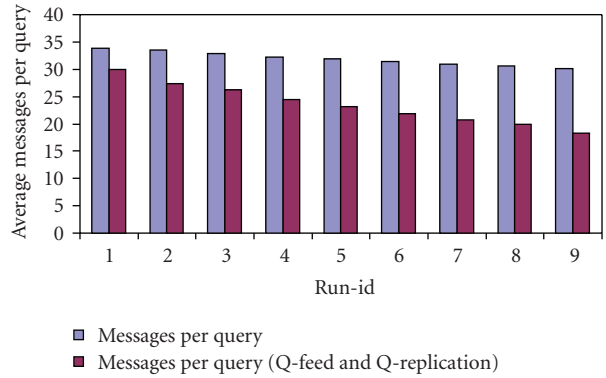


Figure 9: Average messages per query.

messages routed to each neighbour by the peer according to the neighbour status. After each run the total number of messages due to neighbours in different status for all the peers in the network is computed. Using the values for each category and the total number of messages for all categories of neighbours, the overall percentage of each category of messages is computed and plotted. This is shown in Figure 7.

As shown in Figure 7, the number of messages produced by nodes possessing normal state goes marginally in a steady state. But the messages from suspended nodes and dormant nodes are well managed from the initial run onwards. Hence, the positive messages originated from well-performing nodes dominate the network traffic so that messages generated unnecessarily from suspended and dormant nodes undergo tight control mechanisms available in Q-Feed.

Average Number of Files Due to Q-Replication and Downloads. This experiment is conducted to compute the average number of files created as a result of autonomous replication scheme and downloading. The average number of files up to each run is computed and plotted as shown in Figure 8. The number of files present in the nodes before replication is not counted. The number of files in each node increases with time. The proposed replication scheme, Q-replication is not related to the free-rider controlling scheme, Q-Feed. Hence, as popularity of a file increases, the number of copies of the file to be created also increases. Due to the spread of

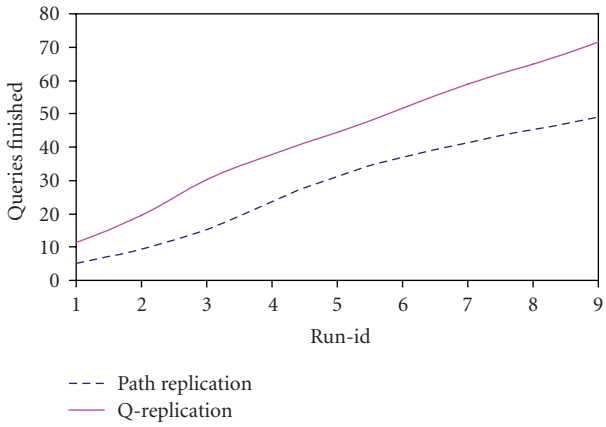


Figure 10: Percentage of queries finished.

popular objects in the network, the query success rate also increases. Each query hit may also create another copy of the object in the query source by means of downloading from the node where query hit occurs. Hence, the availability of objects in the network also relies on query success rate. The Q-replication contributes more number of replicas of popular objects.

Average Messages Per Query. The aim of this experiment is to find the average number of messages generated for each successful query for k-random walk with and without Q-replication and Q-Feed. The results are plotted as graph in Figure 9. The messages per query are declining at a snail's pace for the case which doesn't employ the proposed techniques. The decrease in messages for the proposed solution relies on availability of popular objects due to Q-replication and the stringent measures of Q-feed for reducing the ill-effect of free-riders.

Queries Finished (k-Random Walk with Path Replication) . simulation experiments are conducted in a random network comprising 10 000 nodes to compare the performance of path replication [9] and Q-replication on random K-walk search technique [9]. Path replication replicates an object along the path of a successful "walk". It does not cover nay other node in the network for hosting replicas. The number of walkers are limited to six. The results for queries finished in each simulation run are shown in Figure 10. The success rate of random k-walk search with Q-replication is higher than the success rate produced for random k-walk with path replication technique. The influence of Q-replication in success rate improvement is very high as compared to path replication in each simulation interval. Each simulation run creates new replicas of popular objects in various nodes. Q-replication creates replicas of popular objects in more number of nodes and at the same time, path replication relies only the nodes on the search path in which the target nodes to host replicas are not selected based on their performance in the past.

9. Conclusions

The proposed solutions for managing free-riders widely employ the Q-learning concepts. The Q-Feed algorithm is a greedy algorithm for controlling services among low performing neighbours of a peer. A neighbour may be in one among the four states at a time based on its services rendered to a peer. The nodes possessing low status are encouraged to increase their position. Simulation results show that Q-Feed effectively manages free-riders in the network. The average number of messages generated by for each successful query is significantly reduced. In addition, Q-replication productively replicates the popular objects among well-performing nodes and the algorithm assists the neighbours to prevail from free riding behaviour by hosting popular objects in the shared folder. Like monetary based techniques, no central monitoring agent is required for Q-feed. Q-feed encourages all free riders to attain higher status.

The proposed replication approach utilises the popularity of the objects and the objects are distributed with more copies to various sites based on site selection logic. The popularity is computed according to the queries received on a particular objects and the total number of queries received by the node for a certain period. The target nodes are selected neither randomly nor probabilistically, but they are chosen based on their past performance. The replication scheme does not rely on nodes on the search path. Other nodes can also host the same replica of the object, provided that the sites satisfy certain criteria. The replacement of a file follows a different approach and it depends on its popularity and age.

References

[1] C. Erten and R. MacManus, "P2P: Introduction and real world applications," Read/Write Web, March 2007, http://www.readwriteweb.com/.

[2] S. Tewari, *Performance study of peer-to-peer file sharing*, Ph.D. thesis, University of California, Los Angeles, Calif, USA, 2007.

[3] A. Kim and L. Hoffman, "Pricing napster and other Internet peer-to-peer applications," Tech. Rep. CPI-2001-02, George Washington University, 2002.

[4] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup service for internet applications," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computers Communications (SIGCOMM '01)*, pp. 149–160, San Diego, Calif, USA, August 2001.

[5] A. Rowston and P. Druschel, "Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware '01)*, Heidelberg, Germany, 2001.

[6] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computers Communications (SIGCOMM '01)*, vol. 31, pp. 161–172, San Diego, Calif, USA, August 2001.

[7] The Gnutella, http://www.gnutellaforums.com/.

[8] The KaZaA, http://www.KaZaA.com/.

[9] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proceedings of the International Conference on Supercomputing*, pp. 84–95, New York, NY, USA, June 2002.

[10] M. Hefeeda, *Peer-to-Peer Systems*, School of Computing Science, Simon Fraser University, Surrey, Canada, 2004.

[11] 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 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, no. 5, pp. 314–329, Lake George, NY, USA, October 2003.

[12] S. Saroiu, K. Gummadi, R. Dunn, S. Gribble, and H. Levy, "An analysis of Internet content delivery systems," in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI '02)*, pp. 315–327, Boston, Mass, USA, December 2002.

[13] S. Sen and J. Wang, "Analyzing peer-to-peer traffic across large networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 2, pp. 219–232, 2004.

[14] M. Feldman and J. Chuang, "Overcoming free-riding behavior in peer-to-peer systems," *ACM SIGecom Exchanges Archive*, vol. 5, no. 4, pp. 41–50, 2005.

[15] E. Adar and B. A. Huberman, "Free riding on Gnutella," *First Monday*, vol. 5, no. 10, pp. 134–139, 2000.

[16] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Multimedia Computing and Networking*, vol. 4673 of *Proceedings of SPIE*, pp. 156–170, San Jose, Calif, USA, January 2002.

[17] D. Hughes, G. Coulson, and J. Walkerdine, "Free riding on Gnutella revisited: the bell tolls?" *IEEE Distributed Systems Online*, vol. 6, no. 6, pp. 1–18, 2005.

[18] L. Ramaswamy and L. Liu, "Free riding: a new challenge to peer-to-peer file sharing systems," in *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS '03)*, Big Island, Hawaii, USA, January 2003.

[19] M. Karakaya, I. Korpeoglu, and Ö. Ulusoy, "Free riding in peer-to-peer networks," *IEEE Internet Computing*, vol. 13, no. 2, pp. 92–98, 2009.

[20] V. Ponce, J. Wu, and X. Li, "An approach to combating free-riding in peer-to-peer networks," *International Journal of Parallel, Emergent and Distributed Systems*. In press.

[21] W. Wang, L. Zhao, and R. Yuan, "Improving cooperation in peer-to-peer systems using social networks," in *Proceedings of the 20th International Parallel and Distributed Processing Symposium (IPDPS '06)*, pp. 446–453, April 2006.

[22] A. Nandi, T.-W. Ngan, A. Singh, P. Druschel, and D. S. Wallach, "Scrivener: providing incentives in cooperative content distribution systems," in *Proceedings of the ACM/IFIP/USENIX 6th International Middleware Conference (Middleware '05)*, vol. 3790 of *Lecture Notes in Computer Science*, pp. 270–291, Grenoble, France, November-December 2005.

[23] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica, "Free-riding and whitewashing in peer-to-peer systems," in *Proceedings of the ACM SIGCOMM Workshop on Practice and Theory of Incentives in Networked Systems (PINS '04)*, pp. 228–235, Portland, Ore, USA, August-September 2004.

[24] V. Vishnumurthy, S. Chandrakumar, and E. G. Sirer, "KARMA: a secure economic framework for P2P resource sharing," in *Proceedings of the Workshop on the Economics of Peer-to-Peer Systems*, 2003.

[25] M. Karakaya, I. Körpeoğlu, and Ö. Ulusoy, "Counteracting free riding in peer-to-peer networks," *Computer Networks*, vol. 52, no. 3, pp. 675–694, 2008.

[26] Z. Zhang, S. Chen, and M. Yoon, "MARCH: a distributed incentive scheme for peer-to-peer networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM '07)*, pp. 1091–1099, Anchorage, Alaska, USA, May 2007.

[27] E. Damiani, S. De Capitani Di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pp. 207–216, ACM Press, 2002.

[28] S. M. Thampi and C. Sekaran K, "Autonomous data replication using Q-learning for unstructured P2P networks," in *Proceedings of the 6th IEEE International Symposium on Network Computing and Applications (NCA '07)*, pp. 311–317, Cambridge, Mass, USA, July 2007.

[29] S. M. Thampi and C. Sekaran K, "Review of replication schemes for unstructured P2P networks," in *Proceedings of IEEE International Advance Computing Conference (IACC '09)*, pp. 794–800, Thapar University, Patiala, India, 2009.

[30] S. M. Thampi and C. Sekaran K, "Q-learning based collaborative load balancing using distributed search for unstructured P2P networks," in *Proceedings of the 33rd IEEE Conference on Local Computer Networks (LCN '08)*, pp. 797–802, Montreal, Canada, October 2008.

[31] Y. Gotou, *Replication methods for enhancing search performance in peer-to-peer services*, M.S. thesis, Department of Informatics and Mathematical Science, Graduate School of Engineering Science, Osaka University, Osaka, Japan.

Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 826351, 11 pages
doi:10.1155/2010/826351

Research Article

A Hybrid Approach to Assess the Network Awareness of P2P-TV Applications

Dario Rossi and Paolo Veglia

INFRES Department, TELECOM ParisTech, 46 rue Barrault, Paris, France

Correspondence should be addressed to Dario Rossi, dario.rossi@enst.fr

Received 24 June 2009; Accepted 30 November 2009

Academic Editor: Dave Marples

In this work, we develop a general methodology to assess the level of network awareness and friendliness of P2P-TV applications. The methodology is based on a combination of active and passive measurement techniques and can be applied to any P2P-TV system since it is designed to work considering such systems as a black-boxes. As an interesting case study, we then apply this methodology to PPLive, one of the most popular P2P-TV systems nowadays. Focusing on the video content distribution, we consider several per-path and per-peer metrics, investigating which of them mostly biases PPLive download preferences. Furthermore, in order to refine the picture of PPLive peer selection policy, we not only study the impact of different metrics in isolation, but also assess the joint impact of different metrics at the same time. Our main finding is that PPLive seems mainly bandwidth greedy, but does not show any preference toward peer proximity based on RTT delay; at the same time, our results also suggest that this choice alone may provide a nonnegligible level of geographical clustering among peers as a beneficial side effect.

1. Introduction and Motivations

We agree with [1] that "*the Obama inauguration marks a historic day in US politics and a remarkable day for the popularity of Internet streaming video. We look forward to watching more great things to come.*" Live streaming of video content over the Internet is finally hitting the masses: it is not hype that P2P-TV systems are candidates for becoming the next Internet killer applications, as testified by the growing success of several commercial systems, among which PPLive [2] is perhaps the most popular.

Yet, despite valuable research already investigated such commercial applications [3–15] still little information is available about their internal algorithms, which are proprietary and closed. Specifically, a major concern is that, unless such systems are considerate of the underlying network, the very same potentialities of P2P-TV may constitute a worry for network carriers, especially in the case of HD streaming. Indeed, notice that while download rate is limited by the stream rate, the upload rate may grow much larger depending on the number of served peers [3]. Furthermore, a P2P application confining most of the generated traffic within the Autonomous System (AS) could noticeably reduce operators' transit costs on peering links [5].

In the last years, a significant research effort in P2P networking has been devoted to techniques for achieving a better mapping between the overlay topology and the underlying physical network [16–18]. This can be obtained by biasing the peer selection process, so to choose overlay neighbors based on their proximity on the underlying IP network, that is, in other words to make the overlay "aware" and "friendly" toward the underlay. Such network-aware techniques can be either enforced at the overlay level alone (by inferring peer proximity based on latency measurement, coordinates systems [16], Autonomous System information [17], etc.) or even by allowing ISPs to participate in this process [18]. At the same time, we point out that most of these techniques have been developed in the more general context of P2P applications and have only more recently been cast to P4P-TV [19–24] which can be explained by the fact that other classes P2P applications, such as file-sharing, can be more easily engineered since they lack the need to preserve the temporal coherence of the fetched content.

Although a number of relevant works [3, 4, 7, 10–13] recently targeted the study of popular Internet P2P-TV applications, still the question remains whether such systems are aware of (and behave friendly with respect to) the underlying IP network, which is precisely the aim of this work. From a high level perspective, we can define two categories of metrics that pertain to network awareness: *pathwise metrics* (such as RTT delay, available bandwidth, packet loss, etc.) are determined by the conditions on the end-to-end path between two peers in the overlay; conversely, *peerwise metrics* (such as Autonomous Systems, geographical location, /16 IP prefix, access capacity, etc.) only depend on properties of a single peer. Our methodology (i) exploits *active techniques to artificially enforce pathwise metrics* in a controlled testbed, while (ii) adopts a *passive technique to infer preference of peerwise metrics* from observation of the traffic, gathered from multiple active probes, in uncontrolled live experiments. We point out that we implemented a slightly modified version of the methodology presented in this work in [14], which is available as an open source software tool at [15].

We apply this methodology to the study of PPLive where the combination of both techniques allows us to draw conclusions that are otherwise precluded using either methodology alone. By means of active measurement, we find that PPLive is mainly bandwidth greedy, but does not show any preference concerning, for example, IP distance or RTT delay. As ISP friendly peer selection and, more generally, network awareness in P2P system has only recently become a topic of interest, it would have rather been surprising if PPLive already explicitly considered this. Yet, by means of passive measurement, we further observe that bandwidth preference alone may provide a nonnegligible level of geographical clustering among peers as a beneficial side effect. While this is an interesting and positive finding, the overall results suggest that supplementary effort is needed in order to further increase the level of network friendliness—which, as PPLive is continuously evolving [25], may very well happen in the near future.

Summarizing, the main contributions of this paper are as follows.

(i) First, we propose a black-box methodology, based on a combination of active and passive measurement techniques, to assess the level of network awareness and friendliness of currently deployed Internet P2P-TV applications.

(ii) Second, we apply our methodology to the analysis of PPLive finding that geolocalization, while not explicitly enforced by the application, actually arises as beneficial side effect of bandwidth preference.

(iii) Third, the methodology allows to investigate the relative preference of different metrics as well; in the case of PPLive, we find that the peer selection process is continuously updated, with a relative preference among pathwise properties that depends on the actual magnitude of the metrics.

The reminder of this paper is organized as follows. After having overviewed related effort in Section 2, we describe the methodology and dataset in Section 3. We then apply the methodology to the analysis of PPLive reporting experimental results of active and passive techniques in Sections 4 and 5, respectively. Finally, Section 6 concludes the paper.

2. Related work

Recently, live streaming P2P-TV systems attracted the attention of the research community: as a result, a number of relevant works exist that exploit measurement techniques to study the behavior of such systems [3–13].

Many valuable work considers a single system, which is analyzed by active crawling as in the case of PPLive [3], CoolStreaming [4], and UUSee [5]. Other works instead focus on very specific aspects of a P2P streaming system, for example, node degree of popular versus unpopular channels [6] and node stability [7], while quality of service is of concern in [8, 9]. Finally, work also exists that attempts at comparing similarities and differences of the above systems as [10], which analyzes the traffic pattern generated by these applications from both a network as well as from a transport layer perspectives. Only very recently, work started to appear that focuses more closely on the study of P2P-TV network awareness [11–13]. Yet, although these works share some similarity in their aim with ours, as detailed in the following the adopted approaches are rather different.

For instance, authors in [11] exploit the application-layer logs of UUSee: interestingly, among their finding they observe a clustering phenomenon among peers in the same ISP, similar to the one pointed out in this paper. At the same time, we stress that our work differs from [11] concerning two important points. A first difference arises in the methodology employed, which in the case of [11] limits the applicability of the effort, indeed, authors not only have knowledge of the P2P-TV system inner workings, but also base their analysis on application-layer logs, which requires thus to instrument the application under analysis and is thus clearly not applicable in case of closed source proprietary systems. A second difference arises instead in the analysis of the results, and on the conclusions that can be gathered. The authors in [11] state that UUSee actually biases the selection of its neighbor peers via both (i) bottleneck bandwidth inference as well as (ii) RTT delay measurement. The authors further observe that a significant fraction of neighbor peers falls into the same ISP, despite that UUSee does not explicitly take into consideration ISP membership. These observations allows them to conclude that the reason behind the clustering is that "as connections between peers in the same ISPs have generally higher throughput and smaller delay than those across ISPs, they are more inclined to be chosen as active connections." Interestingly, in our case we show that a similar geographical clustering can be observed in PPLive, which we also show *not* to be sensitive to RTT preference. To the best of our knowledge, this clustering effect solely induced by bandwidth preference has not been observed by other researchers to date.

A first step toward a methodology that can be applied to any system as a black-box is undertaken by both [12] (which, however, limitedly exploits an active measurement technique) and in our earlier work [13] (which instead only considers a passive measurement technique). In more detail, authors in [12], set up an active testbed to investigate the congestion control algorithms of different P2P-TV applications. Using active probes, authors enforce pathwise properties (such as artificial bandwidth limitations, packet loss, and delay) and examine P2P-TV reaction to adverse network conditions. Inspired by [12], we refine their setup, by (i) considering a more controlled setup and by (ii) jointly applying different impairments, so to assess the relative importance of multiple pathwise properties.

However, not all metrics potentially exploited by the overlay for neighbors selection and chunk scheduling can be artificially enforced via active measurement techniques—as, for instance, is the case of peer geographical and AS locations. In [13] we thus investigated such peerwise properties by means of a large pan-European measurement campaign, exploiting a purely passive measurement approach. In this work, we exploit the same dataset considered in [13] by performing a correlation-based analysis inspired by Principal Component Analysis (PCA) technique. Thus, although the dataset used in this work is the same as in [13], the analysis technique differs from the simpler one adopted in our earlier work [13], where we quantitatively weighted the relative amount of bytes exchanged with a peer that exhibits a specific peerwise property as an indication of P2P-TV system preference toward that property.

In this work, we define a methodology that jointly exploits active and passive measurements, encompassing thus and going beyond [12, 13]; by applying the methodology to the analysis of PPLive as a case study, we show that only the combination of both methodologies allows us to draw conclusions—such as the geographical clustering effect induced by bandwidth preference—that are otherwise precluded using either methodology alone.

3. Methodology

As previously outlined, our aim is to define a methodology able to tell whether a P2P-TV system has some level of knowledge of the underlying IP network, and whether it exploits this knowledge to bias the selection of the overlay neighborhood—especially for what concerns the chunk download preference. From a high level perspective, we can define two categories of metrics that pertain to network awareness.

(i) *Pathwise metrics*, such as IP path length (hops), loss rate, RTT delay, and available bandwidth,, are determined by the conditions on the end-to-end path between two peers in the overlay.

(ii) *Peerwise metrics*, such as Autonomous Systems, geographical location, /16 or /24 IP prefix, access capacity, instead only depend on properties of a single peer.

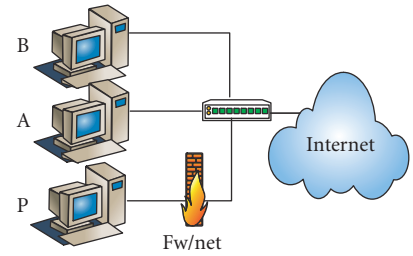


Figure 1: Pathwise metrics: active testbed setup.

In this work, we use two separate sets of experiments to assess the awareness of a P2P-TV system with respect to metrics falling in either of the two above categories.

(i) On the one hand, we exploit an *active measurement* technique to enforce controlled artificial conditions (such as path length, delay, loss and bottleneck bandwidth) on a specific network path in an Internet-scale testbed.

(ii) On the other hand, we adopt a live *passive measurement* approach, where we perform contemporary live measurement of unmodified peers from multiple vantage points, in order to investigate properties (such as AS or geographical location) belonging to real overlay peers.

3.1. Pathwise: Controlled Testbed. For preference related to path metrics, we setup a controlled testbed to enforce artificial network conditions as in [12], from which our approach differs for two main reasons. First, we decide to *completely* control the path metrics. This means that, unlike [12] where impairments are additionally enforced beyond the actual network conditions, we know precisely the conditions of the different peers involved in the experiments. Second, we not only test the impact of each metric in isolation, but also investigate their combined effect as well.

The configuration used for all active experiments is shown in Figure 1. We use three modern desktop PCs equipped with dual-core processor running native installations of Windows XP and of the P2P-TV application, which in our case is PPLive 2.4. Two machines A and B are connected to a network switch through their 100 Mbps Ethernet interfaces. Traffic is observed at the probe PC P, which is connected to the switch through a machine, referred to as *Fw/Net* in the picture, running a linux kernel 2.6 and acting as a bridge, firewall, and network emulator. Notice that a large population of users, as well as the primary source of the video itself, is reachable through the Internet.

At start-up, all machines A, B, and P run undisturbed clients for 5 minutes. During this start-up period (where we verify that play-out starts and that the clients are visually synchronized within a 1-2 seconds range), P naturally receives most of the traffic from Internet hosts. Then, at time $F_{\text{on}} = 5$ minutes, firewall rules are established at *Fw/Net* to block traffic coming from the Internet toward P, which can thus only receive traffic coming from either A or B. In this

Table 1: Peerwise metrics: multiple live measurement setup. Host IDs are internal to the site.

Host	Site	CC	AS	Access	Nat	FW
1–4	BME	HU	AS1	High-bw	—	—
5			ASx	DSL 6/0.512	—	—
1–9	PoliTO	IT	AS2	High-bw	—	—
10			ASx	DSL 4/0.384	—	—
11–12			ASx	DSL 8/0.384	*Y*	—
1–4	MT	HU	AS3	High-bw	—	—
1–3	FFT	FR	AS5	High-bw	—	—
1–4	ENST	FR	AS4	High-bw	—	*Y*
5			ASx	DSL 22/1.8	*Y*	—
1–5	UniTN	IT	AS2	High-bw	—	—
6–7				High-bw	*Y*	—
8			ASx	DSL 2.5/0.384	*Y*	*Y*
1–8	WUT	PL	AS6	High-bw	—	—
9			ASx	CATV 6/0.512	—	—

case, hosts A and B will still receive the video from the remote Internet peers, but our probe P will be forced to receive the totality of the video from A and B.

We then introduce, starting at $R_{\text{on}} = 10$ minutes, artificial network emulation rules (such as packet loss, RTT delay, bottleneck bandwidth limitation, etc.) on the path that joins our probe P to the hosts A and B from which he is receiving the video content. We point out that our aim is to understand how the system biases its peer selection during *normal* operation; we verify that probe P is correctly receiving the video stream. When a path metric X is considered in isolation, we artificially worsen network conditions (e.g., increase packet loss rate, delay, etc.) solely on the path from machine A to P, by properly configuring the queueing discipline on *Fw/Net*. Rules on path from B to P are instead enforced only to investigate the relative importance of different metrics (e.g., delay on A $\rightarrow$ P and loss on B $\rightarrow$ P). Finally, artificial network conditions are turned off at time $R_{\text{off}} = 20$ m and firewall limitations are removed at $F_{\text{off}} = 25$ m.

The above setup allows us to focus on the *breakdown* of the bit-rate received at P between its contributors (i.e., A, B, and Internet hosts), and to express in a visually simple way the network awareness of the P2P-TV application. Consider indeed the period when artificial network emulation rules are applied on the path from A to P, for instance, clearly, in absence of bias with respect to a given metric X, we expect the breakdown of the traffic received at P to be unaffected from variations of X. Conversely, a varying breakdown will reflect system awareness to X, with the extent of the breakdown variation as rough indication of the system sensitivity to X.

3.2. Peerwise: Multiple Live Measurement. The analysis of preference related to peer properties is instead based on a large testbed, setup in the context of the NAPAWINE project [26]. In this case, we mine the data gathered from the experiment in order to infer additional information concerning the P2P-TV system bias on peerwise properties.

The main idea is to use a *correlation-based* analysis of any given peerwise metric X and the amount of bytes exchanged between contributor peers. As peerwise metrics X, we will consider the Autonomous System (AS) and geographical Country (CC) properties. In this case, our aim is to test whether two peers that belong to the same AS/CC exchange more data than faraway peers, gauging the importance of X in the peer selection process. Deferring further details on the quantification of this bias, along with considerations concerning the fallacies of correlation-based analysis to Section 5, we now briefly describe the Internet-scale experiment setup used to gather the passive measurement dataset.

More precisely, partners run PPLive clients on PCs connected either to the institution LAN, or to home networks having cable/DSL access. In more detail, as summarized in Table 1, the setup involves a total of 46 peers, including 37 PCs from 7 different industrial/academic sites, and 7 home PCs. Probes are distributed over four European countries and connected to 6 different Autonomous Systems, while home PCs are connected to 7 other ASs and ISPs. Therefore, the setup is representative of a significant number of different network environments. Several 1-hour long experiments were performed during April 2008, when partners watched the same channel at the same time and collected packet-level traces. Since P2P-TV applications are mostly popular in Asian countries [3], we tuned PPLive to CCTV-1, a popular channel, during China peak hours. Nominal stream rate is 384 kbps, Windows Media 9 Encoder is used, and the perceived video quality is similar for all partners. We point out that during the experiments, several of our peers act as amplifiers [3]; that is, they exhibit an upload/download ratio significantly larger than 1, which further justifies potential ISPs worries concerning P2P-TV systems.

Results reported in this paper refer to 44 hours of video, during which our probes exchanged about 70 GBytes of data in 157 million packets with nearly 1 million external peers.

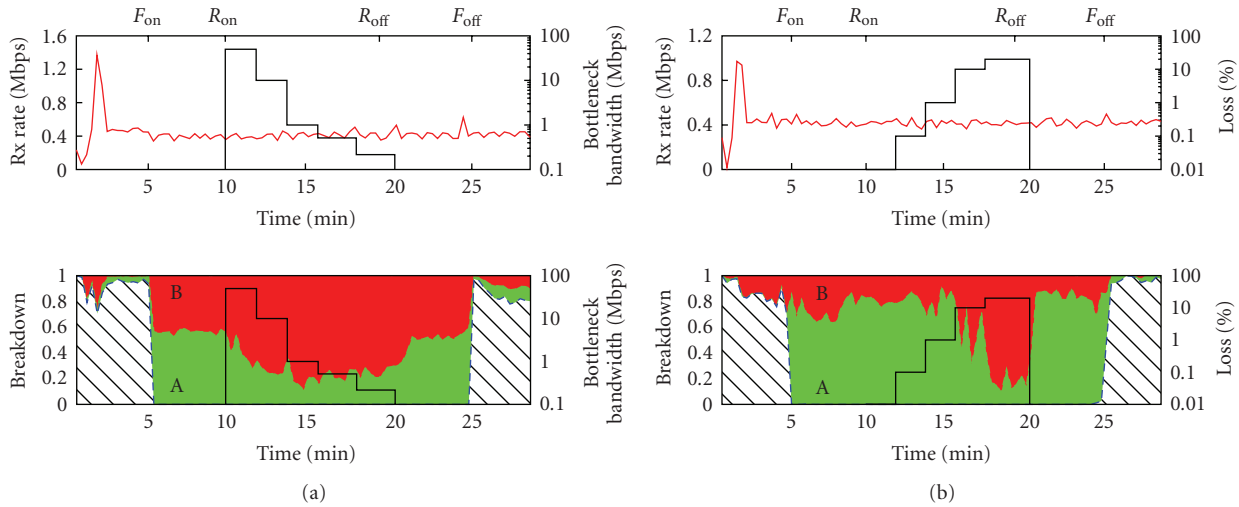


FIGURE 2: Pathwise metrics: PPLive aggregated received rate at P (top) and its breakdown between A, B and Internet hosts (bottom) for varying bottleneck bandwidth (a) and packet loss (b). Profiles of the bottleneck bandwidth and packet loss impairments are reported as solid black lines directly in the plot and refer to the right y-axis. PPLive shows a great sensitivity towards path capacity variations while it reacts only to very high loss rates.

We stress once more that, despite the dataset reported in Table 1 is a subset of the one considered in [13], nevertheless the analysis of the dataset carried on in Section 5 differs from our earlier work [13].

4. Experimental Results: Pathwise Metric

Let us start by inspecting pathwise preference, by means of the controlled testbed depicted early in Figure 1. Initially, we study pathwise metrics in isolation, enforcing either (i) decreasing bottleneck bandwidth, (ii) increasing packet loss rate, (iii) increasing RTT delay, (iv) increasing IP hop count on the path from host A to the probe P.

We then inspect the relative importance of the above metrics in the peer selection process by jointly considering different metric combinations, applying one condition (e.g., bottleneck bandwidth) on the path from the host A to the probe P, and a different condition (e.g., packet loss rate or RTT delay) on the B $\rightarrow$ P path.

4.1. Bottleneck Bandwidth. Results of the first experiment are reported in Figure 2(a). Time of the experiment runs on the x-axis, while the firewall start and end times are reported on the top axis as a reference. A decreasing bandwidth profile is enforced starting at R_{on} by means of a token bucket filter, with steps of $C = \{50, 10, 1, 0.5, 0.25\}$ Mbps every 2 minutes, as shown by the thick black line. Values of the bottleneck bandwidth are reported on the right y-axis, and the bottleneck is removed at R_{off}. The time evolution of the aggregated *received* rate at P is reported in the top portion of the plot, averaged over 20 seconds intervals. It can be seen that, after an initial start-up phase $t < F_{on}$ in which the incoming rate peaks up to 1.2 Mbps, the aggregated received throughput at P is steady around 400 Kbps, which account for both signaling and video traffic. Moreover, notice that the aggregate rate is undisturbed during the *whole* experiment, hinting to the fact that traffic shaping did not perturbed the perceived quality of service.

Bottom plot of the figure report the *breakdown* of the traffic incoming at P with respect to the different hosts that sent the traffic to P: host A is depicted at the bottom with light (green) color, host B with dark (red) color, and the remaining Internet hosts with a dashed pattern. It is easy to gather that, before firewall rules are in place $t < F_{on}$, more than 80% of the incoming traffic is received through Internet hosts. As soon as firewall rules start at $t = F_{on}$, P is forced to receive traffic exclusively from hosts A and B; since during $F_{on} < t < R_{on}$, no bottleneck bandwidth is enforced yet, the traffic splits roughly equally between A and B, as the network condition and play-out time of hosts A and B are alike. Then, as soon as a 50 Mbps bottleneck bandwidth kicks in at R_{on}, PPLive *immediately* starts preferring the unconstrained host B, which then provides the most significant portion of the traffic to P.

This observation is important as it means that (i) PPLive is extremely sensitive to the bandwidth and (ii) it may overreact or perform faulty bandwidth estimations. This behavior might be due to the token bucket shaper used to enforce bandwidth limitations, causing strange arrival patterns that mingle the bandwidth estimation algorithm. Moreover, packet time stamping may also bias the results (e.g., by poor timing due to clock drift, clock skews due to NTP synchronization, etc.). More likely, since packets of the same chunk are sent out in bursts [3], implementation of hardware card and drivers may play a very important role as well, especially due to *interrupt coalescing*. This feature, aimed at avoiding the overkill of raising an IRQ signal for every packet received, makes the cards wait during a short time-window for the arrival of other packets prior to notify the reception to the upper layers. Since packet time-stamping

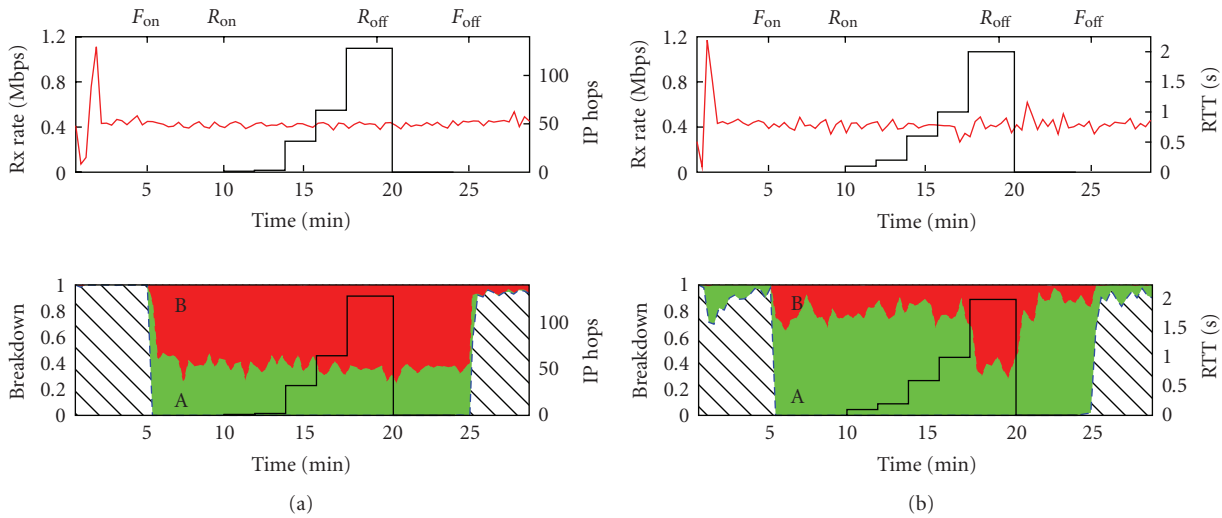


FIGURE 3: Pathwise metrics: PPLive aggregated received rate at P (top) and its breakdown between A, B and Internet hosts (bottom) for varying IP hop distance (a) and RTT delay (b). Profiles of the IP hop distance and RTT delay impairments are reported as solid black lines directly in the plot and refer to the right y-axis. Here, PPLive shows no awareness of IP distance and a mild sensitiveness towards Round Trip Time.

is then performed by the OS, this means that interrupt coalescing has a possibly very nasty impact on the bandwidth estimation as well, because two consecutive packets received by the card during the same interrupt coalescing window will then be sent to the upper layer almost at the same time. As a consequence, bandwidth and capacity estimation tools that are based on packet pair or packet dispersion will rather measure the PCI bus speed more than bottleneck in the network.

Finally, bottleneck limitations are removed at R_{off}, which partly removes the breakdown bias toward host B; this holds until the firewall limitations are removed as well at F_{off}, after which contributors are again to be found mainly among Internet hosts.

4.2. Packet Loss. We then conduct similar separate experiments for the other considered metrics, enforcing a single impairment at any time. Results for the packet loss rate experiments are reported in Figure 2(b) using the same similar visual presentation (i.e., aggregated rate, breakdown, and packet loss profile).

We notice again that the incoming traffic rate at P is steady for any packet loss rate; indeed, since only the path A $\rightarrow$ P is impaired, P has still the possibility to receive the rest of the video from B. In this experiment, starting at time R_{on}, we increased packet loss rate experienced by host A using the profile $L = \{0.01, 0.1, 1, 10, 20\}\%$.

As the picture clearly shows, PPLive is not very sensitive to packet loss; noticeable changes in the breakdown happen only when packet loss percentage *exceeds* 10%—indeed when loss rate is 10%, still more than half of the data is downloaded from the impaired host A. If we couple this observation to the fact that, despite packet loss increases considerably, the sent traffic rate (measured at A and not shown in the picture) does not increases proportionally, we can conclude that PPLive seems to use an effective FEC techniques, as already observed in [12].

4.3. IP Distance. We then test whether PPLive is aware of the host proximity, which we measure in terms of the number of IP routers that packets cross on their path across the network. Applications using raw UDP sockets can infer IP proximity by means of the Time To Live (TTL) field of IP packet header, which is initially set to an OS-dependant value (namely, 60 or 64 for BSD and Linux, 128 for Windows) and then decremented by one unit at each hop in the network. We artificially increase the number of hops on the path from A $\rightarrow$ P by subtracting the number of additional hops $H = \{1, 2, 32, 64, 100\}$ from the IP TTL header field at *Fw/Net*.

As before, a change in the hop profile is enforced every two minutes, and the results are shown in Figure 3(a). As there is no noticeable change in the breakdown irrespectively of the additional hops values, we can conclude that PPLive is either unaware the IP hop distance between two hosts, or that its inner algorithms do not rely on this piece of information.

4.4. RTT Delay. Finally, we verify whether PPLive does instead take latency measurement into account. We increase the delay on the A $\rightarrow$ P path so that the Round Trip Time (RTT) equals RTT $= \{0.1, 0.2, 0.5, 1, 2\}$ s, and report the results in Figure 3(b).

The plot clearly asserts that PPLive is not very sensitive to the delay, as the breakdown does not show any noticeable change until the round trip delay grows very large (RTT $>$ 1 s). We can thus conclude that PPLive peers do not bias their download policy in terms of nodes proximity, neither in terms of IP hop distance, nor in terms of delay. In other words, PPLive does not seem to implement proximity techniques such as [16–22].

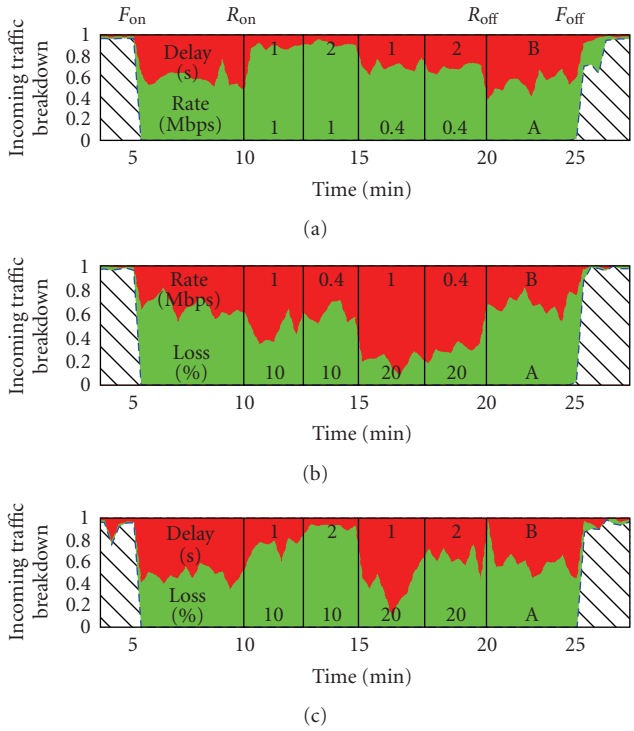


FIGURE 4: Pathwise metrics: combined impairments, considering RTT delay versus bottleneck bandwidth (a), bottleneck bandwidth vresus packet loss (b), RTT delay versus packet loss (c). Relative importance of impairments depends on their actual magnitude.

Yet, another very important remark can be gathered from the picture. Indeed, when RTT $>$ 1 s, the breakdown drastically drops; this suggest that PPLive *does* actually measure RTT, which is however not used afterward for the proximity-based video contributors selection. This behavior can be explained by recalling that one of the main aims of scheduling in P2P-TV streaming is to reduce as possible the play-out delay of the whole system. Therefore, despite that useful video content may be available at high RTT peers, such peers are preferentially discarded as they may not *timely* contribute to the video content delivery, and as such they would increase the whole system play-out delay. In other words, it seems that PPLive, streaming to keep a low end-to-end play-out delay, uses RTT as a "sanity-check" and disregard such peers on purpose to avoid the system pollution.

4.5. Combined Pathwise Metrics. In order to further refine the knowledge concerning PPLive network awareness, we investigate how PPLive reacts to different combination of impairments, so to sketch a relative order of importance of the above metrics. As we show that PPLive is not sensitive to IP hop count, we now limitedly consider packet loss, RTT delay and bottleneck bandwidth limitations, applying an impairment X on the A $\rightarrow$ P path, and another impairment Y on B $\rightarrow$ P at the same time.

For the sake of readability, we consider only a couple of values for each metric (i.e., $X_{hi} > X_{lo}$ and $Y_{hi} > Y_{lo}$) and investigate PPLive behavior on the four different operational points resulting from their combination $(X, Y) \in (\{X_{hi}, X_{lo}\} \times \{Y_{hi}, Y_{lo}\})$. Results are shown in Figure 4, which reports for the sake of readability the value of the impairment applied to a specific path directly on the plot. In this case, to avoid cluttering the pictures, we no longer report the aggregated received rate but limitedly depict its breakdown.

4.5.1. Delay versus Rate. Top plot of Figure 4 reports the case in which we apply a delay RTT $= \{1, 2\}$ s on the B $\rightarrow$ P path and enforce a rate limitation of BW $= \{0.4, 1\}$ Mbps on A $\rightarrow$ P. It can be seen that preference goes toward bandwidth limited host and is mainly driven by the bottleneck bandwidth; indeed, almost all content is downloaded from A when the rate limit is set to 1 Mbps, irrespectively on the delay toward B. Once the bottleneck rate drops to BW $=$ 0.4 Mbps, the number of video chunks downloaded from B slightly increases (even though the rate limit would allow almost the whole content to be downloaded from A), but no noticeable effect of RTT variation is shown.

4.5.2. Rate versus Loss. Middle plot of Figure 4 refers to a rate limitation BW $= \{0.4, 1\}$ Mbps on B $\rightarrow$ P and loss rate $L = \{10, 20\}$% on A $\rightarrow$ P. In this case, contrary to the previous experiment, we see that both metrics have an impact in determining the breakdown. When $L = 10$% breakdown is roughly equal for A and B, with a slight bias toward A when bandwidth toward B drops at 0.4 Mbps. When losses instead grow to $L = 20$%, the bandwidth limited host is always preferred, though the actual bandwidth limit still slightly influences the breakdown value.

4.5.3. Delay versus Loss. Finally, bottom plot of Figure 4 refers to a delay enforcement of RTT $= \{1, 2\}$ s on B $\rightarrow$ P and loss rate $L = \{10, 20\}$% on A $\rightarrow$ P. Again, both metrics play an important role in determining the breakdown, depending on the impairment level. As expected, loss $L = 10$% does not constitute a significant impairment, as such lossy path is preferred toward high RTT path, when RTT $=$ 1 s, peer A is almost completely ignored. Behavior changes completely as soon as losses grow to $L = 20$%, in which case breakdown favors completely B when RTT $= 0.5$ s and is more fairly split when RTT $= 1$ s.

The above observations allow us to conclude that the *relative* preference of pathwise metrics is weighted on the ground on the actual magnitude of the impairment. In principle, we point out that by exploring a wider number of (X, Y) impairment couples, it should be possible to get an even finer picture of the relative preference, but this falls outside the scope of this paper.

5. Experimental Results: Peerwise metric

In this section, we refine the picture of PPLive awareness by mining data gathered in the multiple-vantage point testbed. Following the methodology defined in [3], we extract from our traces about 16500 peers that contribute by providing

video content. We focus on peers' Autonomous System (AS) and geographical location, which we represent by Country Code (CC) information. For each probe peer x in the testbed, we analyze the CC and AS properties of all its contributors peers y, gathered by *whois* and open IP databases queries. We point out that contributors y in this case may be either probes taking part in the experiment, or external peer of real users. As such, we no longer control the properties related to their path, and we need to infer them from packet level traces in case of need.

As core tool in this case, we use a correlation-based analysis, inspired by Principal Component Analysis (PCA) technique. While PCA is often used for dimensionality reduction—that is, to transform a set of correlated variables into a smaller subset of uncorrelated variables, called principal components—in our case our aim is to gauge the extent of the correlation, so to show the existence of a dependence (if any) between these variables.

More precisely, let us consider a set of N contributor peers $p_1 \cdots p_N$ observed during an experiment. By denoting with $X_i = X(p_i)$ the value of property X for peer p_i and similarly with $Y_i = Y(p_i)$ the value of property Y for the same peer, we measure the correlation between X and Y over the whole experiment as

$$\rho(X,Y) = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}, \tag{1}$$

where μ_X and μ_Y are the means of X and Y over all samples, σ_X and σ_Y are the sample standard deviations of X and Y, respectively. Usually, (1) is referred to as the Pearson product-moment correlation coefficient which, dropping the sum bounds for the sake of readability, can be rewritten as

$$\begin{aligned}\rho(X,Y) &= \frac{\sum X_i Y_i - nE[X]E[Y]}{(N-1)\sigma_X \sigma_Y} \\ &= \frac{N\sum X_i Y_i - \sum X_i \sum Y_i}{\sqrt{N\sum X_i^2 - \left(\sum X_i\right)^2}\sqrt{N\sum Y_i^2 - \left(\sum Y_i\right)^2}}.\end{aligned} \tag{2}$$

5.1. Autonomous System and Geolocation. We are interested in assessing if PPLive is AS- and CC-aware, and whether its video scheduling policy exploits such information; that is, if in other words our PPLive probes tend to download video content from contributors falling in the same AS or CC. By mining the experimental data, we find that, despite only 1.3%(1.4)% of peers fall in the same AS(CC) of the probe, about the 12.8%(13.1%) of bytes are downloaded from them to further quantify this evident degree of geolocalization among contributors, we evaluate the coefficient of correlation ρ between the amount of bytes RX received from any given contributor x and the fact that this contributor belongs to the same AS or CC. Considering all probes x in the experiments, and by using the indicator function $I(x,y) = 1$ when both x and y belong to the same AS or CC, we obtain $\rho(\mathrm{RX}, \mathrm{AS}) = 0.21$ and $\rho(\mathrm{RX}, \mathrm{CC}) = 0.17$, respectively, which accounts for modest (though not negligible) correlation.

This is however surprising, since the controlled testbed early suggested that PPLive peers are greedy in terms of bandwidth, but that are not sensitive otherwise to the fact that contributors are "close" in underlay terms (e.g., IP distance or latency).

5.2. Bandwidth. We are therefore interested in assessing if (and to what extent) the geolocation can be a (rather desirable) *side effect* of PPLive bandwidth sensitivity. Therefore, we further evaluate the bandwidth (BW) between probes and contributor by measuring the throughput of chunks, which are typically sent out in packet bursts, to further investigate the existence of correlation between peerwise metrics.

5.2.1. Bandwidth Estimation Techniques. Since we are unaware of the technique actually employed by PPLive to measure the available bandwidth, we adopt a hands-on approach; we estimate BW using multiple techniques and require an agreement of our observations over *all* techniques. Considering only the downstream traffic direction of a contributing peer toward one of our testbed probes, we evaluate the BW over *windows* of fixed length. We express the window length in terms of either (i) a number of consecutive packets N or (ii) a temporal duration ΔT. Let us denote by t_i and B_i, respectively, the arrival time and size of the ith packet downloaded by probe x from contributor y during the current observation window. In case of fixed-length packet trains, we estimate the bandwidth BW_N over the current window as the amount of bytes carried by the train of consecutive N packets as

$$\mathrm{BW}_N = \sum_{i=1}^{N} B_i/(t_N - t_1). \tag{3}$$

In case of fixed-duration trains, we estimate the bandwidth as

$$\mathrm{BW}_{\Delta T} = \sum_{i=1}^{N(\Delta T)} B_i/\Delta T, \tag{4}$$

where $N(\Delta T)$ is the number of packets received during ΔT.

As far as the window length N and duration ΔT are concerned, we point out that their choice is made complex not only by the fact that we are unaware of the chunk size and chunk start time, but also from the fact that the estimation can be severely influenced by factors such as interrupt coalescing (which however affects both our estimates and PPLive methodology as well). We argue that choosing large values of N and ΔT would yield less noisy results, but due to chunk scheduling policy, it may introduce a *bias* in the result. Intuitively, counting the number of packets over large time windows equals to count the number of chunk exchanged, rather than their actual transmission throughput. Using large windows, peers that more actively contributed to the transmission will thus appear as *both* preferred and high-bandwidth, introducing thus an artificial correlation between the two terms. To avoid this bias, shorter windows should be preferable. At the same time, too small values of N and ΔT should be avoided, indeed, as packets are sent out in bursts, it may happen that interrupt

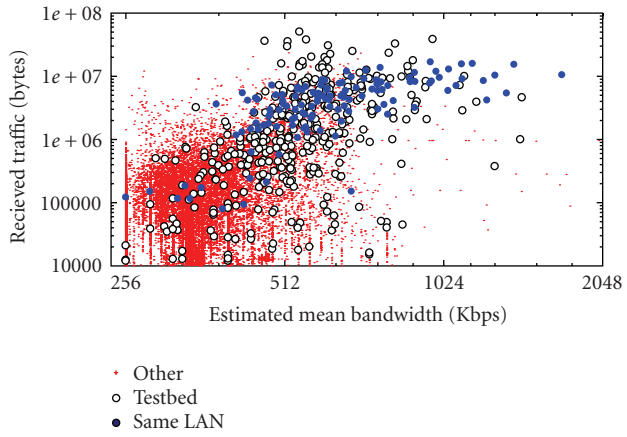


Figure 5: Scatter plot of received traffic versus estimated mean bandwidth (log-log scale).

coalescing (which we verified to be present in out traces) can squeeze the packet arrival pattern and increase the estimated BW.

In reason of the above observations, we select values of $N = \{5, 10, 20\}$ packets and $\Delta T = \{50, 100, 250\}$ ms. For every peer pair, we then construct a series of several BW samples gathered during the whole experiment, of which we then compute the mean and 99th percentile (p_{99}) values; we argue that both statistics are relevant, since the mean value is indicative of instantaneous network conditions, whereas p_{99} may be more representative of peer y access capacity. For lack of space, we will only report a subset of results, selecting $N = 10$ packets and $\Delta T = 100$ ms, since we verified that the same conclusions hold using the other parameters as well.

Scatter plot of the amount of received traffic versus mean bandwidth $\mathrm{BW}_{\Delta T}$ is depicted in Figure 5, using $\Delta T = 100$ ms and log-log scale. Dark points represent exchanges between any two probes in the same LAN, white points are used for probes belonging to our testbed but belonging to different institutions, whereas any other contributor is represented with a small dot. First, hosts within our testbed, and especially hosts within the same LAN, achieve higher rates with respect to Internet hosts. Moreover, the estimated $\mathrm{BW}_{\Delta T}$ values are sound and consistent with our expectation.

5.2.2. Correlation-Based Analysis. Another interesting observation to gather from Figure 5 is that host achieving higher data rates also tend to contribute more data, and that this behavior is consistent across all three host groups.

To further quantify this behavior, we evaluate the coefficient of correlation $\rho(\mathrm{RX}, \mathrm{BW})$ between the amount RX of bytes received by a given probe and the bandwidth BW toward that contributor. For comparison purposes, we also evaluate the coefficient of correlation between the estimated bandwidth BW between two peers and the fact that they belong to the same AS or CC (using the indicator function as before). Though we are aware of the fallacies of correlation based analysis, we point out that we do not seek to *prove* directional cause-effect relationship between variables, but that we rather compare the magnitude of the correlation and relatively weight their impact.

Table 2: Correlation $\rho(X, Y)$ between bandwidth (BW), received bytes (RX) and peer localization information (CC, AS) for different groups of contributor peers (LAN, Internet, all).

	Y :	$\mathrm{BW}_{\Delta T}$		BW_N	
X :		mean	p_{99}	mean	p_{99}
AS		0.28	0.44	0.27	0.29
CC		0.27	0.42	0.26	0.26
RX	all	0.43	0.54	0.45	0.53
	LAN	0.62	0.75	0.51	0.61
	!LAN	0.33	0.40	0.37	0.45

Results are reported in Table 2 for different bandwidth estimation methods. In case of $\rho(\mathrm{RX}, \mathrm{BW})$ we also consider different peers subsets, namely, peers falling in the same LAN, peers that do not belong to the same LAN and all the peers altogether. As expected, we observe that irrespectively of the BW evaluation method considered, there is medium correlation between received bytes and bandwidth $\rho(\mathrm{RX}, \mathrm{BW})$, which is clearly stronger for peers belonging to the same LAN. Moreover, notice that this correlation is stronger with respect to $\rho(\mathrm{RX}, \mathrm{AS})$ or $\rho(\mathrm{RX}, \mathrm{CC})$ reported earlier, even when all peers are considered. Also, notice that the correlation between bandwidth and geolocation $\rho(\mathrm{BW}, \mathrm{AS})$ (i.e., the fact that high bandwidth contributors can be found within the same AS) is of the same order of magnitude of $\rho(\mathrm{RX}, \mathrm{AS})$ (i.e., the fact that video content is downloaded from contributors within the same AS). Overall, these observations suggest that, even outside the LAN environment, peers are primarily looking for bandwidth and that the early noticed geolocalization may be a beneficial side effect of the enforced bandwidth preference *alone*. Finally, we point out that part of the correlation might be explained by means of (i) mutual dependency between AS and BW, as well as (ii) additional hidden factors which causes both AS and bandwidth preference. At the same time, such hidden factors (e.g., preference based on IP/16 address similarity) are likely to play only an additional role beside the one played by the bandwidth, to which we shown early PPLive being extremely sensitive to.

6. Conclusions

This work proposed a methodology, based on the joint use of active and passive measurement technique for the analysis of the network awareness of currently deployed Internet P2P-TV system. The technique has been designed so to consider P2P systems as a black-box and as such can be applied to future systems as well. As a case study, we applied the methodology to the analysis of PPLive a very popular system nowadays, gathering interesting results, that we briefly summarize here.

First of all, by means of active testbed methodology, we find PPLive to be extremely sensitive to bandwidth, only mildly sensitive to losses and mostly unaware of IP distance, expressed in terms of either delay or IP hop

count, which is in agreement with [12]. Refining further this picture, we find that actually the peer selection process is continuously updated, with a relative preference among pathwise properties that depends on the actual magnitude of the impairment.

Interestingly, by the correlation analysis of peerwise preference gathered through the passive technique, we find that the very same bandwidth sensitivity of PPLive seems to induce a desirable side effect, namely, a moderate geoclusterization of peers within the same AS and CC. Yet, it seems that PPLive does not, for the time being, explicitly enforce AS-awareness, which remains thus a new exciting challenge for the next steps of its evolution.

Acknowledgments

This work was funded by EU under the FP7 Collaborative Project "Network-Aware P2P-TV Applications over Wise-Networks" (NAPAWINE). The authors wish to thank all the project partners for having taken part to the multiple live measurement experiment.

References

[1] C. Labovitz and S. Iekel-Johnson, "The Great Obama Traffic Flood," January 2009, http://asert.arbornetworks.com/2009/01/the-great-obama-traffic-flood.

[2] PPLive, http://www.pptv.com.

[3] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A measurement study of a large-scale P2P IPTV system," *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1672–1687, 2007.

[4] B. Li, S. Xie, Y. Qu, et al., "Inside the new Coolstreaming: principles, measurements and performance implications," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '08)*, pp. 1705–1713, Phoenix, Ariz, USA, April 2008.

[5] C. Wu, B. Li, and S. Zhao, "Multi-channel live P2P streaming: refocusing on servers," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '08)*, pp. 2029–2037, Phoenix, Ariz, USA, April 2008.

[6] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt, "Measurement of a large-scale overlay for multimedia streaming," in *Proceedings of the 16th International Symposium on High Performance Distributed Computing (HPDC '07)*, pp. 241–242, Monterey, Calif, USA, June 2007.

[7] F. Wang, J. Liu, and Y. Xiong, "Stable peers: existence, importance, and application in peer-to-peer live video streaming," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '08)*, pp. 2038–2046, Phoenix, Ariz, USA, April 2008.

[8] X. Hei, Y. Liu, and K. W. Ross, "Inferring network-wide quality in P2P live streaming systems," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 9, pp. 1640–1654, 2007.

[9] S. Agarwal, J. P. Singh, A. Mavlankar, P. Baccichet, and B. Girod, "Performance and quality-of-service analysis of a live P2P video multicast session on the Internet," in *Proceedings of the IEEE International Workshop on Quality of Service (IWQoS '08)*, pp. 11–19, Enschede, The Netherlands, June 2008.

[10] T. Silverston and O. Fourmaux, "Measuring P2P IPTV systems," in *Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '07)*, June 2007.

[11] C. Wu, B. Li, and S. Zhao, "Exploring large-scale peer-to-peer live streaming topologies," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 4, no. 3, 2008.

[12] E. Alessandria, M. Gallo, E. Leonardi, M. Mellia, and M. Meo, "P2P-TV systems under adverse network conditions: a measurement study," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '09)*, pp. 100–108, Rio de Janeiro, Brazil, April 2009.

[13] D. Ciullo, M. A. Garcia, A. Horvath, et al., "Network awareness of P2P live streaming applications," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing (IPDPS '09)*, Rome, Italy, May 2009.

[14] D. Rossi, E. Sottile, S. Valenti, and P. Veglia, "Gauging the network friendliness of P2P applications," in *Proceedings of the ACM SIGCOMM, Demo Session*, Barcelona, Spain, 2009.

[15] http://www.infres.enst.fr/~drossi/P2PGauge.

[16] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *Proceedings of the ACM SIGCOMM*, Portland, Ore, USA, 2004.

[17] R. Bindal, P. Cao, W. Chan, et al., "Improving traffic locality in BitTorrent via biased neighbor selection," in *Proceedings of the International Conference on Distributed Computing Systems (ICDCS '06)*, 2006.

[18] V. Aggarwal, A. Feldmann, and C. Scheideler, "Can ISPS and P2P users cooperate for improved performance?" *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 3, pp. 29–40, 2007.

[19] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in *Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '02)*, pp. 177–186, Miami Beach, Fla, USA, May 2002.

[20] R. Rejaie and A. Ortega, "PALS: peer-to-peer adaptive layered streaming," in *Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '03)*, pp. 153–161, Monterey, Calif, USA, June 2003.

[21] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: peer-to-peer media streaming using collectCast," in *Proceedings of the ACM International Multimedia Conference and Exhibition (MM '03)*, pp. 45–54, Berkeley, Calif, USA, November 2003.

[22] D. Ren, Y.-T. H. Li, and S.-H. G. Chan, "On reducing mesh delay for peer-to-peer live streaming," in *Proceedings of the Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '08)*, pp. 1732–1740, Phoenix, Ariz, USA, April 2008.

[23] R. Iqbal, B. Hariri, and S. Shirmohammadi, "Modeling and evaluation of overlay generation problem for peer-assisted video adaptation and streaming," in *Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '08)*, pp. 87–92, Braunschweig, Germany, 2008.

[24] D. Purandare and R. Guha, "An alliance based peering scheme for P2P live media streaming," *IEEE Transactions on Multimedia*, vol. 9, no. 8, pp. 1633–1644, 2007.

[25] G. Huang, "Experiences with PPLive," in *Proceedings of the ACM SIGCOMM P2P Streaming and IP-TV Workshop (P2P-TV '07)*, Kyoto, Japan, August 2007, Keynote Speach.

[26] "Network-Aware P2P-TV Application over Wise Networks," http://www.napa-wine.eu/cgi-bin/twiki/view/Public.

Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 216582, 14 pages
doi:10.1155/2010/216582

Research Article

Providing Adapted Contextual Information in an Overlay Vehicular Network

José Santa, Andrés Muñoz, and Antonio F. Gómez-Skarmeta

Department of Information and Communications Engineering, University of Murcia, Campus de Espinardo, 30100 Murcia, Spain

Correspondence should be addressed to José Santa, josesanta@um.es

Received 29 May 2009; Revised 23 October 2009; Accepted 21 November 2009

Academic Editor: Dave Marples

Current vehicular networks are developed upon commercial solutions based on cellular networks (CNs) or vehicular ad-hoc networks (VANETs), both present in numerous research proposals. Current approximations are not enough to cover the communication necessities of several applications at the same time, and they are not suitable for future vehicular pervasive services. The vehicular network presented in this paper fills the existent gap between solutions lacking in flexibility, mainly supported by an infrastructure deployment, and those highly local and distributed, such as sole-VANET approximations. In this manner, an overlay communication platform which can work over the CN basis has been designed and developed. This architecture is complemented by an additional support of an information system located at the infrastructure side. Moreover, since most of the information received from current notification services is not relevant for the driver, an additional subsystem has been devised to provide adapted information to users. This has been carried out by means of an ontology model which represents users' preferences and contextual information. Finally, using a whole prototype of the telematic platform, the performance of this interring process has been evaluated to point out its impact on the system operation.

1. Introduction

Vehicular networks are becoming essential for telematics services within the Intelligent Transportation Systems (ITSs) field. The usefulness of wireless data communications in vehicles is noticeable in current monitoring solutions, which help companies to manage their transport fleets. These systems are mainly based on a navigation unit and a communication channel, which is usually established through the cellular network (CN). These first systems have been, therefore, the starting point of vehicular communications. However, new generation services conceived for future cars need a more suitable communication platform to connect the vehicle with the environment [1]. Unfortunately, the solutions provided by the current research on this topic are highly case-specific in most cases. Ideally, a network should be used to connect vehicles among them and to the infrastructure, and it should cover all communication necessities of all possible services aimed at the vehicle or the road side [2].

According to the current state of vehicular networks, connectivity requirements can be divided into vehicle to vehicle communications (V2V) and communications with the infrastructure, following a vehicle to infrastructure (V2I) or infrastructure to vehicle (I2V) pattern. Examples of V2I and I2V communication technologies can be found in monitoring systems, traffic information systems such as Radio Data System (RDS) or Traffic Message Channel (TMC), and electronic fee collection systems. The most extended technologies in commercial products which follow these communication patterns are CN, FM radio, and Dedicated Short Range Communications (DSRC). Regarding V2V solutions, vehicular ad hoc networks (VANETs) using WLAN (wireless LAN) and DSRC are the prevailing technologies, and they are mainly used in safety applications. These services use V2V communications to propagate messages over a network created by vehicles. VANET solutions fit satisfactorily in services which require a low latency to communicate with surrounding vehicles. However, they suffer from routing problems in long transmission ranges,

where multihop techniques must be used [3]. In such situations an infrastructure access could improve the performance.

Maintaining the driver up-to-date about potential collisions is considered as one of the main goals on vehicular communications developments. Although this kind of safety applications has practically received all the attention in the past, new technological improvements and the globalization of services offered in other environments have led to the consideration of new generation services for vehicles. Information provision systems such as meteorology, congestion, or repair services are only the first step. Thus, some examples of new information services for future vehicles are digital identification of variable road signs, provision of context-aware information about the traffic state, and reception of commercial or cultural information depending on the location and the user's preferences.

The platform described in this paper works in this line, proposing an overlay vehicular network enhanced with context determination capabilities to provide adapted notifications to users. Section 2 introduces the advantages of a hybrid vehicular network which integrates CN advantages using a decentralised approach by means of a P2P (Peer to Peer) technology. Section 3 relates this proposal to the current literature about overlay networks and context management in the vehicle field. The networking platform and the ontology-based contextual subsystem are then explained in detail in Sections 4 and 5, respectively. Section 6 describes the prototype developed to validate the enhanced platform, and Section 7 evaluates it in terms of the processing time needed to infer contextual information adapted to users. Finally, Section 8 points out the conclusions of this paper and it gives some future research directions.

2. Approximation

As stated in the previous section, both VANET and CN have valuable features for vehicular communications. VANET solutions offer reliable connectivity among close vehicles, due to the same cars create a scalable and cooperative mesh where each vehicle acts as a router. On the other hand, CN offers long range communications thanks to a direct connectivity to the Internet through an operator's network. This communication paradigm has two extra advantages: the unlimited rate of equipped vehicles, and the use of a proved deployed technology. Advantages of both VANET and CN approaches are combined in this paper so as to establish a starting point to integrate all the services they offer individually. By using a P2P paradigm over the cellular network, we obtain an architecture which takes advantage of the benefits in both approaches. P2P networks create a virtual decentralised architecture where individual nodes can communicate without knowing physical details about the underlying network. Latency limitations for close V2V communications are initially inherited from CN. A CN connection cannot match the latency times of VANET systems between nearby cars. However, new improvements point to CN technology as a valid carrier of vehicular transmissions for a wide number of services, and it can now be considered as a suitable complement to VANET approaches [4]. The UMTS (Universal Mobile Telecommunications System) technology has greatly increased data rates and reduced the end-to-end latency. The cost of the communication channel is also an important issue in CN. However, due to special agreements between operators and service providers, the cost of CN data connections is gradually decreasing.

A V2V network allows vehicles to communicate and propagate information over a limited area. Such a strategy is useful to notify nearby vehicles about traffic hazards, road conditions, traffic jams, and other local events. However, a V2I link offers extra benefits. In this line, our proposal represents a step forward, using an infrastructure system to provide context-aware information to drivers depending on their preferences. Broadly speaking, the drivers take advantage of a global system capable of processing all the events received from the roadside. This system can give a global vision of the road network state, processing information from vehicles and roadside hardware, and performing monitoring tasks. Thus, it could be possible to notify vehicles about traffic problems which affect a long highway, send a warning message when forecasting a congestion, or report any pollution problem in an area, for instance. These examples are only some of the possibilities such a system offers by means of traffic data analysis and a combination of V2V and V2I communications, hence overcoming limitations of current traffic information systems like TMC.

Information provided to drivers can also be adapted using such a system. (In this paper the words "driver" and "user" are indistinctly employed to refer the person who uses the system in a vehicle.) The architecture described in the next sections includes an inference process, supported by an ontology model, which adapts information provided to vehicles according to the driver's preferences. Drivers and road operators can modify the system behavior, and relevant points of interest (POI) can be notified to the vehicle, according to customized context rules. This service exemplifies the potential of infrastructure-based services, as a complement to V2V possibilities.

3. Related Work

The suggested proposal is based on an overlay network which works over the UMTS cellular network. Neither the use of overlay nor cellular networks in a general ITS communication platform are broadly treated in the literature. In [5] the use of overlay networks over the CN basis for ITS is defended, to solve deployment limitations which can be found in VANET solutions. In [6] a P2P approach is used in a vehicular network; hence vehicles are organized in dynamic communication groups to exchange PVT (Position Velocity Time) information. However, a method to interconnect these groups and provide a link with the infrastructure is not given. UMTS is considered as an appropriate communication technology for P2P in [7], but the adaptation of underlying protocols to mobility conditions is identified as a key issue.

The JXTA technology [8], used in the communication platform of this paper, is considered in a sensor network over GPRS (General Packet Radio Service) in [9]. However, a special rendezvous node must be installed inside the operator's network, because GPRS does not support multicast. The communication architecture described in this paper has been evaluated using the HSDPA (High-Speed Downlink Packet Access) technology, which improves the overlay network performance and supports multicast. In [10], authors describe the implementation of a GPRS-based network to disseminate traffic information. These services are of special relevance in our proposal; hence, the implementation of the presented platform includes some of them. The form in which traffic events are disseminated in that work is similar to the group-based approach used in this paper. However, contrarily to [10], here it is not necessary to continuously track the position of vehicles in order to determine the ones to be notified about incidences. An evolved development of this concept is given in [11], where the MBMS (Multimedia Broadcast Multicast Service) is used to propagate traffic events inside service areas. The same idea is considered in the platform presented in this paper, but at a logical level through the overlay network.

The integration of the vehicle in the traffic context is a new concept which has recently appeared in ITS. This is integration require not only a suitable networking architecture, which allows for V2I and I2V communications, but also a proper support of the remote infrastructure. In [12] a global architecture for processing vehicle information and disseminate traffic events is given. The system implementation is not included, but it considers the base cellular network as a good candidate to develop a primitive mechanism based on SMS (Short Message Service) to exchange information. A platform with an extended ubiquitous nature is presented in [13]. Here, high-level protocols used in the Internet, such as HTTP (HyperText Transfer Protocol), are used to provide ITS Web services. Nevertheless, it considers sporadic WLAN connections as Internet access, thus limiting the availability of the platform. In [2] a multipurpose system to provide contextual information to vehicles offers extended navigation capabilities to users, such as the current state of a specific road. The telematics architecture described in this paper uses ontologies to model the traffic environment and user's profiles in order to offer similar capabilities. In [14], ontologies are used to implement a common interface for road operators. This enables the implementation of semiautomatic monitoring tasks. In [15] the ITS platform where ontologies are used is more general, and they are considered for exchanging information among different components of the architecture. This comprises an homogeneous method of sharing data. In [16] an ontology for modeling the vehicle concept is formally created, although the application field is quite specific. The work included in [17] describes the idea of modeling user's preferences by means of profile ontologies. This idea is also followed in our paper to adapt contextual information to the user's needs.

4. Networking Platform

The main goal of this paper resides in the development of a suitable networking platform for implementing ubiquitous services which span V2V, V2I, and I2V communications. These services are then provided to new generation vehicles in order to extend the applications offered in them. Figure 1 shows an overall view of the architecture of this platform. The lower part of the diagram involves the P2P vehicular network, whereas the upper part illustrates the additional support of the infrastructure to provide contextual information.

Every vehicle drives along roads with service provision capabilities, and each coverage area and its associated services are registered in a global entity called Group Server (GS). The services considered in the system have an informative nature and exploit V2V, V2I, and I2V capabilities. Hence, they include safety services such as breakdown or repair notification services or tourism and travel information about the current place. The notification mechanism is carried out according to a publish/subscribe scheme, where vehicles subscribe to some services and receive asynchronous notifications.

4.1. Overlay Vehicular Network. As shown in Figure 1, the entities involved in the overlay network communicate by means of JXTA or, directly, through TCP. Using a P2P network with JXTA technology, vehicles communicate between themselves and with the infrastructure. As previously stated, the communication technology used is cellular networks, by means of an Internet link through the operator's infrastructure. The geometrical information about every coverage area and its P2P communication groups is stored in GS. Thus, every service available in each area uses a P2P group, which limits the propagation of messages and it is changed when the vehicle enters a new area. Vehicles pass from one coverage area to another through a hand-off process aided by a navigation system. The vehicle uses a TCP/IP-based protocol to send a hand-off request to the GS, which replies with the P2P connection details and the geometry of the new area.

Messages transmitted over the JXTA overlay network are routed through logical pipes, and they contain information about the source of the message, the type of event, and the payload. If a message is sent by a vehicle, the packet includes the current location and an integrity factor to bound the goodness of the computed position. When a vehicle which is subscribed to a service sends a message in this communication system, it is received by all the vehicles in the area which are also subscribed. This mechanism offers a V2V communication paradigm. However, it must be noted in Figure 1 how an infrastructure entity, placed in every coverage area, listens to all events notified in the zone; this is the Environment Server (ES). Thus, ES is in charge of processing all the events sent in a V2V scheme, and besides it plays a forwarding role between vehicles and the infrastructure in a V2I communication scheme, but it does not communicate with other ESs. Because the ES is connected to the rest of the roadside hardware, it is

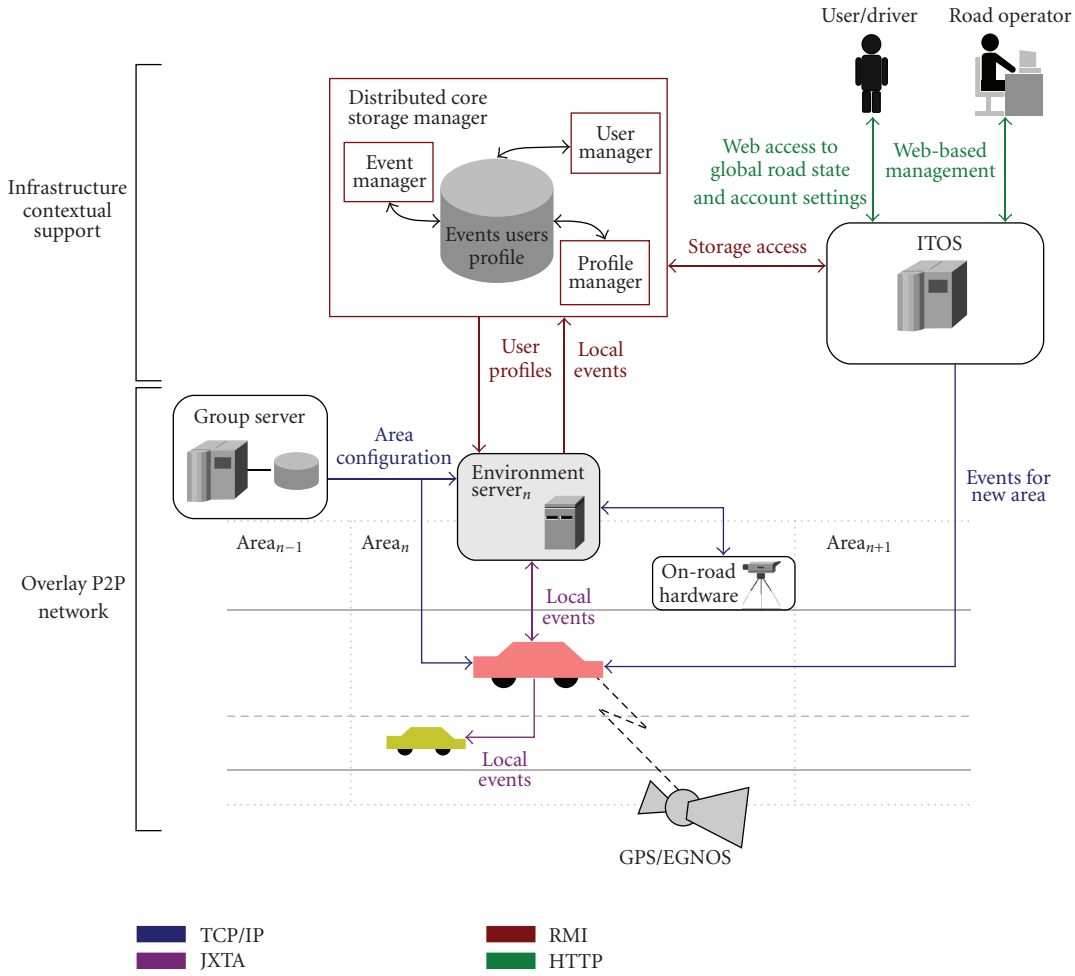


Figure 1: Overlay communication architecture and information processing system.

able to send certain notifications to vehicles, such as "your speed is over the limit", using a unicast mechanism. The communication between the roadside hardware and the ES can be physical, by means of a serial connection, for instance, or remote, using a wired or wireless network. Moreover, roadside devices could be part of the P2P network, if they have sufficient resources.

Environment Servers are logical entities and, therefore, they can be installed on physical computers located at the roadside or executed in a remote server at the core infrastructure. As can be appreciated in the vehicular network design, scalability of the system is directly related to the deployment of service areas. Since these areas are virtual, it is only necessary to update the database of service zones to add or remove areas. Moreover, when physical Environment Servers are not needed, only a new instance of the software must be executed at the core system if a new ES is necessary. This choice, apart from reducing scalability costs, offers more flexibility and eases maintenance tasks.

4.2. Protocol Details. It is clear that several communication protocols are necessary to maintain the connection of vehicles with the network and route information messages properly. The different messages used in the vehicular network are described next.

Table 1 includes the messages exchanged between the vehicle (V) and the Group Server (GS). This communication is carried out by a message exchange over TCP, using the cellular network interface directly. The first process between these two entities is the communication management. An initial *Area_Update_Query* message is used by the vehicle for registering itself against GS. A *Disconnect* message serves for ending the communication. The roaming process uses the *Area_Update_Query* and *Area_Update_Response* messages. The first one is used when a vehicle detects the end of its current coverage area. In that moment, it sends a new *Area_Update_Query* message with its new location. GS replies with an *Area_Update_Response*. This message contains the available services in the zone and the P2P parameters to enable the subscription to them. When the user wants to use a defined service, he sends an *Event_Subscription* message listing the services in which he is interested. It is important to note that each service usually has a different P2P group in each area, in order to offer different communication groups.

Regarding the communication between the Environment Server (ES) and GS, Table 2 summarizes all the exchanged

Table 1: Vehicle-Group Server communication.

Process	Messages	Sender
Communication Management	Area_Update_Query	V
	Disconnect	V
Roaming Process	Area_Update_Query	V
	Area_Update_Response	GS
Event Subscription	Event_Subscription	V

Table 2: Environment Server-Group Server communication.

Process	Messages	Sender
Communication Management	ES_Registration	ES
	ES_Registration_OK	GS
	ES_Registration_ERROR	GS
	ES_Registration_Update	ES
	ES_Registration_Update_Response	GS
Message Propagation	Neighbour_Groups_Query	ES
	Neighbour_Groups_Response	GS

messages over TCP. The *ES_Registration* message is used by ESs to connect with GS. This packet contains the location of the ES and thus GS is able to search for the P2P parameters of all services available in the zone and reply with an *ES_Registration_OK* message. If any error occurs, GS sends an *ES_Registration_ERROR*. This packet includes the reason of the error. Since Environment Servers are static entities which may be connected during long time periods, they need to update their configuration periodically. *ES_Registration_Update* message performs this task, by requesting an update of the P2P parameters. GS answers with this area information by means of *ES_Registration_Update_Response*. The second group of messages is used in the message propagation process. Sometimes it is necessary to ensure that all vehicles near the problem receive critical events. The location of such incidences may be close to the border of the coverage area. Consequently, it is necessary to propagate the associated message to the neighbouring areas. In order to perform this task, Environment Servers listen to all events and use the *Neighbour_Groups_Query* message to ask for the neighbouring P2P groups for the service in question. GS replies with this information through *Neighbour_Groups_Response*. Now Environment Servers can forward the message to neighbouring areas.

Finally, Table 3 includes the three messages used by services to send messages over the P2P network. These packets are sent using a JXTA unicast communication pipe. The first message in this table is *Vehicle_Event*. It contains information sent from a vehicle service edge. All vehicles located in the area along with the Environment Server receive these messages. *Environment_Event* messages are sent by Environment Servers. These are used, for example, in the message propagation method described previously. Messages *Vehicle_Event* and *Environment_Event* contain the following fields:

Table 3: Vehicle-Vehicle/Environment Server communication.

Process	Messages	Sender
Message Passing	Vehicle_Event	V
	Environment_Event	ES
	Specific_Environment_Event	ES

(i) source: the vehicle ID, which is currently considered as the license plate number (*Vehicle_Event*), or the Environment Server identifier (*Environment_Event*), which is a unique string;

(ii) location: the GPS position of the sender entity, using latitude and longitude;

(iii) integrity: the integrity value of the position [18], indicating the reliability of the location calculated by the navigation system;

(iv) payload: The content of the message.

The last packet listed in Table 3 enables Environment Servers to send messages to specific vehicles, and it does not include any information about position (neither location nor integrity). This message is used in a service which provides contextual information adapted to user profiles, as described later in the paper.

For the moment, privacy and integrity issues have not been considered in the different messages exchanged among the entities of the communication platform. Those messages sent over TCP could be secured using Secure Socket Layer (SSL) or Transport Layer Security (TLS). In the case of the P2P traffic, it is envisaged to use JXTA secure communication pipes, which use certificates to secure signaling and data packets.

4.3. Infrastructure Support. A complete information system at the infrastructure side has been developed in the platform. The distributed core storage manager system, represented by a dashed line in Figure 1, is implemented by means of remote objects which are used by the rest of the infrastructure entities through Remote Method Invocation (RMI). By using these objects, ESs communicate with the core storage system, forwarding new events from vehicles and road hardware. The Internet Traffic Operation Server (ITOS) provides a Web access with a complete view of road events. To this end, ITOS analyzes the roadside information, accessible via the core storage manager. This Web application offers a differentiated access to users and operators, by means of an authentication stage. Operators, unlike normal system clients or users, have an administration account with management capabilities. Furthermore, any user of the Web application is able to check the state of the roads, at home or even using the on-board computer, via the Internet connection.

From this architecture explanation it can be seen that information from subscribed services is sent or received by means of events. These come directly from other vehicles or from the infrastructure, via ES entities. Events from the

roadside have a local scope, because they comprise context-aware information exchanged in a V2V pattern. Due to hand-offs between coverage areas, it is necessary for vehicles to receive important events of the new area they enter, which were previously collected from the roadside. ITOS carries out this task through a TCP/IP-based protocol, as Figure 1 shows.

5. Management of Contextual Information

The efficient management of contextual information is a key feature of the architecture presented in this proposal. As explained in this section, this information is represented here by the points of interest (POI) of the road and drivers' profiles. The platform uses a rule-based reasoning system to match the preferences contained in the drivers' profiles with the characteristics of nearby POI. As a result, relevant information for the drivers is inferred when they are driving along services areas. This functionality is offered as a service implemented in the platform, which is accessible by users. The adaptation of the contextual information to each particular driver is the ultimate goal of the context manager subsystem introduced in this section.

5.1. Ontology-Based Modeling and Inference. The ability of being *context-aware* is receiving a considerable attention when applying it to mobile devices, as these devices are particularly affected by environmental changes. A general motivation is that context-awareness can serve to compensate for the abstraction that is required to make systems react and adapt "sensibly" to changing environments and situations. According to [19], "context is any information that can be used to characterize the situation of an entity. An entity is a person, a place, or object that is considered relevant to the interaction between a user and an application." Therefore, a context-aware system needs to maintain a contextual interpretation of a situation. This implies the use of some type of knowledge representation directed to introduce context information in the system and the capability to reason about this knowledge.

One appealing alternative to represent context information is offered by means of Semantic Web ontologies [20]. An ontology, related to the computer science field, is a formal representation of a set of concepts within a domain and the relationships between those concepts. From this conceptualization, it is possible to define a specific situation in the domain by instantiating the concepts and their relations. A Semantic Web-based ontology uses the same conceptualization idea, and moreover it is explicitly augmented with semantic axioms (e.g., subsumption of concepts, disjointness, etc.). Several advantages are acquired by using domain representations based on Semantic Web ontologies. Firstly, the information represented by this kind of ontologies can be easily and broadly exchanged among heterogeneous applications which share the same conceptualization. Another fundamental capability provided by adding semantics to a domain representation is the execution of inference processes over such a domain. One of the results of these processes is the extraction of new knowledge that was implicit in the domain. This kind of inference process is achieved by triggering some predefined set of rules about concept hierarchy, types of relationships, instances of concepts, and so forth. stated in the ontology. Taking this idea further, we can define our own rule set to infer new knowledge according to the contextual information modeled in the domain (i.e., POI and drivers' profiles). Thus, the overlay network system will react by offering relevant information to the drivers regarding the context in the ongoing situation.

There are several languages for describing Semantic Web ontologies, such as RDF (Resource Description Framework) [21] and OWL (Web Ontology Language) [22]. Furthermore, the latter has been adopted as standard by the W3C (http://www.w3.org/2004/OWL/). In particular, the ontologies developed in the overlay communication platform are written in OWL-DL, a broadly used OWL version which takes Description Logic (DL) [23] as its formal underlying language. In this paper ontologies are presented at a high level of abstraction, since such a level is enough to show their aim and functionality in the platform. The reader is referred to the references above for OWL's syntax and further details.

Once the contextual information has accordingly been modeled by means of OWL-DL ontologies, this model can now be used to produce rules, called *context rules* henceforth. Context rules provide a means to infer how the overlay network system should react (e.g., to send notifications) according to the context in the road side (e.g., nearby POI or situation of interest such as traffic jams) and drivers' profiles. Namely, these rules allow expressing some specific situations based on the vehicle context and the desirable reaction that should be performed in the system when such a situation occurs. In this sense, the rules introduce knowledge to be interpreted by the system according to the current context of a situation. The deductive process necessary to execute context rules on ontologies is achieved by means of the Jena framework [24]. Jena provides a set of methods for loading and managing OWL ontologies, together with a rule-based reasoning engine. Now, let us see each of these components in detail.

5.2. Determining the Vehicle Context. The first question posed when defining the vehicle context is which elements comprise this kind of context. *Location* is the most widespread context source [25]. As a result, the location-based services (LBSs) are considered as a particular case of general context-aware applications. Proliferation of navigation devices has attracted the LBS concept to different areas such as creation of new services in mobile phones or intelligent transport systems. Apart from location, there exist other context sources which are also important. All these sources are identified by the "5Ws" [26]: *who*, the identification of a user of the system and the role that user plays within the system in relation to other users; *what*, the recognition of activities and tasks users are performing; *where*, the tracking of the location where a user or object is geographically located at each moment; *when*, defining the instant of access to the system; and *why*, the capacity to infer

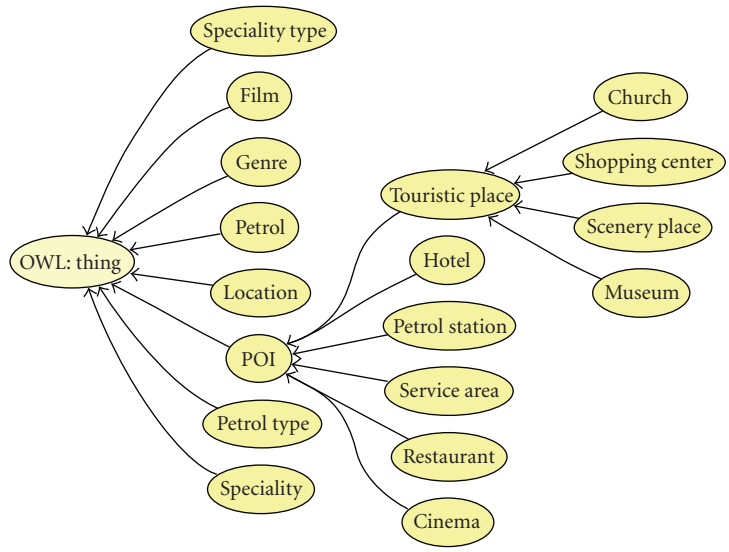


Figure 2: Hierarchical representation of Environment ontology schema.

and understand the intentions and goals behind activities, which is one of the hardest challenges in this field, but at the same time a fundamental one for enabling the system to anticipate users' needs.

In order to consider all these factors, not only should location technologies (e.g., presence sensors) be employed, but also other fields such as artificial vision or environment sensing (e.g., temperature, humidity, and pressure) could be present in the architecture. As a result, in this paper we gather the elements needed in the platform to determine the vehicle context by means of three main concepts: vehicle location, service, and identity. These concepts are built upon the vehicles and drivers information retrieved by the overlay network.

5.2.1. Vehicle Location. The vehicle's geographical position is a key feature with respect to the propagation and reception of information in a specific area of interest (*where*). The navigation system included in the vehicle enables its localization in the road network. On the other hand, the communication architecture manages the notifications received and transmitted from and to the vehicles both at local and global levels. Indeed, this centralized management allows the architecture to notify previous events, mainly related to traffic incidents, which are still valid in the current context (*when*).

5.2.2. Service. Each service represents the type of information which users or applications are interested in. Since each communication service is goal-oriented, it is possible to determinate what activity is the user performing due to his interest in a particular type of information (*what*). In the proposed architecture, both users and applications can subscribe to specific communication services, thus avoiding the rest of notifications related to other services which are not relevant for them. A diversified offer of services may help to delimit the activity performed by the driver. For instance, subscribing a tour-guided service or asking information for public parking places gives meaningful information about the type of ongoing activity.

5.2.3. Identity. The proposed context management system enables the customization of the information provided by the platform once the driver is identified (*who*). This process is composed of two steps. The first one is based on the identification of vehicles, which allows the overlay network to identify the driver and retrieve his profile. The second step uses these drivers' profiles to execute the service according to the preferences included in them. Both drivers' profiles and identification of vehicles are supported by ontology models and RFID (Radio Frequency Identification), respectively.

5.3. Ontology Models and Information Management. The representation of contextual information in this proposal has been modeled by means of two OWL-DL ontologies: The *Profile* ontology, which defines a schema with the different driver's preferences about restaurants, hotels, museums, and so on (e.g., price and quality rates); and the *Environment* ontology, which represents the points of interest which can be found in each service area: restaurants, hotels, petrol stations, or touristic places such as churches, shopping centers, museums, and so on. The schema of this latter ontology is depicted in Figure 2. The use of these ontologies suits smoothly with the information management requirements in the ITS platform depicted in Figure 1. This requirement is based on the distributed composition of such an architecture. Consequently, the ontologies are spread among its different components. Notice that it is important to distinguish between the schema and the instances of an ontology. While there is only one schema that represents the domain being modeled, there could be several instances of that schema for the specific information captured. For example, there is

(at least) one instance of the Profile ontology schema for each driver. On the other hand, there exists an instance of the Environment ontology schema for each correspondent service area which represents the available POI in such an area.

In order to process the local notifications in the service areas, the ontologies have been allocated in their appropriate component of the architecture. The instances of the Profile ontology are maintained in the Profile Manager, as part of the Distributed Core Storage element. The ITOS Web front-end gives access to the drivers' profiles stored here. Contrarily, the instances of the Environment ontology are distributed among the ESs. This organization is justified because Environment Servers are responsible for updating the state of the road and POIs in their areas, whereas the drivers' profiles should be in a global and centralized point, so as to be rapidly transferred to the correspondent ES when a driver is detected in its area. The instances of the Environment ontology can be locally managed in each ES or remotely, via the ITOS Web front-end.

5.4. Adapting Contextual Information. When a vehicle is detected in a delimited service area, the ES associated to that area obtains the driver's profile from the Profile Manager and then it infers new information of interest for the driver (if any). Figure 3 shows a sequence diagram with the elements involved during the management of contextual information. Observe that the method by which the ES detects the location of the vehicle is independent from the inference process and the knowledge model. In our proposed platform, an RFID system is used to detect the presence of a vehicle. After detecting it, the RFID system sends a `Reader_Notification` message to the ES. This message includes the vehicle's RFID tag identifier, which determines the driver unambiguously. This notification protocol has been deployed over a TCP/IP connection. Then, the ES requests the driver's profile to the Profile Manager by invoking the `profileRequest` remote method with the RFID tag identifier as parameter. Once the driver's profile is retrieved, the ES processes the local information about the area, together with this profile, to infer the interesting notifications in that area. When the inference process is executed, the ES sends a `Specific_Environment_Event` message to the correspondent vehicle by means of the P2P JXTA network. This message contains the resulting matches together with a matching ratio. Finally, the on-board unit in the vehicle is in charge of displaying this information for the driver in an appropriate manner.

The inference process in the ES follows the next steps. First, the driver's profile is combined with the instance of the Environment ontology maintained in this ES. Then, the context rules defined in the system are executed over the combination of this contextual information. Context rules are in the form of

$$X_1 \wedge X_2 \wedge \cdots \wedge X_n \longrightarrow Y_1 \wedge Y_2 \wedge \cdots \wedge Y_m, \qquad (1)$$

that is, an "*if-then*" implication between a conjunction of antecedents X_i and consequents Y_j (the symbol $\wedge$ represents the logical conjunction "and"). These rules contain concepts, relationships, and instances described in OWL-DL ontologies. More specifically, each X_i/Y_j is a triple (s, p, o), where s (*subject*) is an instance of a concept, p (*predicate*) is a relationship among concepts or a data property of a concept, and o (*object*) is a concept, an instance, or a data value. Note that s, p, and o could be variables to be instantiated when evaluating the rules. Each triple represents an assertion in the domain model, that is, a fact about the current context. The semantic of these rules states that whenever all the triples specified in the antecedent are true in a domain, the triples in the consequent must also be true in such a domain.

Finally, context rules are evaluated by a deductive inference process, which is performed by the Jena rule-based reasoning engine integrated into each ES. For each context rule available, the reasoning engine checks if its antecedents are satisfied in the contextual information maintained in the ES. As seen before, this information is obtained after combining the correspondent driver's profile and POI data, represented as instances of the Profile and Environment ontologies, respectively. The inferred information in the consequents of each triggered rule is then added to the driver's profile and notified to the ES. Consequently, the ES can now send such inferred information to the driver's vehicle, as explained above.

The next rule *menuPrice* is shown as an example of context rule. Variables are represented by a string starting with the symbol "?". The rest of elements are concepts or relationships defined in the Profile and Environment ontology, denoted by the prefixes *prf* and *srv*, respectively. Particularly, this rule searches for a driver's profile (1) which contains preferences about restaurants (2); according to a preferred maximum and minimum menu price (3),(4) the driver is willing to pay. Then, if any restaurant in the Environment ontology (5) has a menu (6) whose price is less or equal than the preferred maximum price ((7), "*le*" is the comparison operator *less or equal*) and greater or equal than the preferred minimum price ((8), "*ge*" is the comparison operator *greater or equal*), the evaluated restaurant is added to the list of driver's profile matches (9), specifying the menu price condition as satisfied (10):

```
    [menuPrice:
(1) (?profile rdf:type prf:Profile)
(2) (?profile prf:restaurantProfile ?rp)
(3) (?rp prf:maxMenuPrice ?maxmp)
(4) (?rp prf:minMenuPrice ?minmp)
(5) (?res rdf:type srv:Restaurant)
(6) (?res srv:menuPrice ?mp)
(7) le (?mp,?maxmp)
(8) ge (?mp,?minmp)
    ->
(9) (?profile prf:matches ?res)
(10) (?res srv:matcheswith ''menuPrice'')]
```

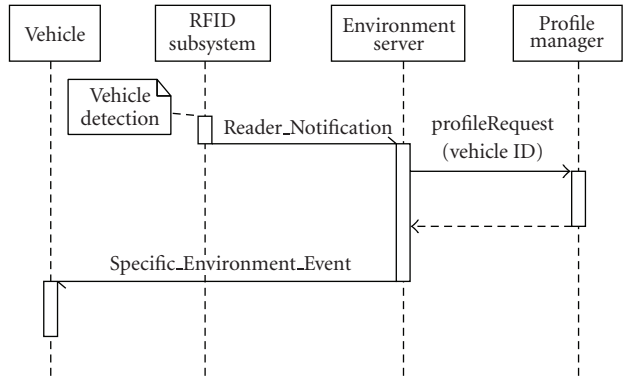


Figure 3: Sequence diagram for adapting contextual information.

Observe that some functions such as mathematical operations (e.g., comparisons of greater and lesser) could also be part of the antecedents and consequents. Context rules like the previous one are written by system administrators according to a set of templates which are used to collect the drivers' requirements. This process is assisted by a Web interface at ITOS. Administrators are assisted in writing the rules by a tool called ORE (Ontology Rule Editor) (http://sourceforge.net/projects/ore) [27] which has been developed within our research group. Since these context rules are defined by using elements of the Profile and Environment ontologies, they can be combined with any specific instances of both ontologies. As a result, the same rules can be executed in any ES, but the results could be different depending on the contextual information maintained in each ES.

6. Prototype Set-Up

A prototype vehicle [28] has been set up with the necessary hardware and software to deploy the on-board part of the architecture previously presented. An FV-21 positioning sensor and a UMTS Huawei E220 modem, which support HSDPA, have been used to provide positioning (through the on-board navigation system [29]) and communication capabilities. The Environment Server entity has been implemented and installed on several Linux-based PCs, using a Java environment and the Jena framework. To test the connection between an ES and the rest of road side hardware, an RFID reader has been set up to detect the presence of vehicles through a tag affixed to the windscreen. The same RFID approach or any other positioning technology (such as cell determination in UMTS) could be added to the platform, in order to improve the reliability of the system when vehicles are identified in service areas.

The photographs included in Figure 4 show the previous set-up. The RFID reader (a WaveTrend AA-R500-SP) has been installed at the test place using an ad hoc gantry. A laptop is connected via serial port to the reader, in order to send presence notifications to the corresponding ES. This communication has not been secured for the moment, but it is envisaged to secure these messages through SSL, as in the rest of TCP connections. From a wide set of tests with the RFID gantry, it has been checked that good identification rates are obtained at speeds under 50 km/h; hence, it is specially appropriate for urban environments or service stations, for instance. The Group Server and the Internet Traffic Operation Server have been developed in Java and then installed over a high-performance server, in order to cover high rates of queries. The distributed core storage manager is composed of a set of RMI entities, which perform information management tasks (Profile Manager, User Manager and Road Event Manager), and a remote MySQL database which physically stores all data. The RMI entities work as independent applications, and each one publishes a remote object which is used by the ITOS and ES. In our prototype, the RMI applications and the MySQL server run on the high-end computer, but they could have been installed on different hosts.

A new module has been included in the software platform of the on-board computer [30], which is based on Java and OSGi (Open Services Gateway Initiative), to offer JXTA communications through multicast pipes. This module makes P2P programming tasks easier to the developer. A service has also been implemented to access the networking architecture through several reference services. This is the Message Console, which appears in Figure 5. The user connects to the services he is interested in by pressing the available buttons on the right part of the window. When the activated services appear in the current coverage zone, their corresponding images appear below, and it is possible to receive events from outside or send a new one. The central part of the window shows all the received messages about all active services (road incidences, weather, tourist information, etc.). In this example, the vehicle has received information about hotels and cinemas. The service which provides such information is called "On Road Information" and uses the RFID system to locate the vehicle at a specific place. When the ES receives the presence notification, it asks the core infrastructure for the user's profile. Then, the ES infers interesting information for the driver, via its local database about the environment. Finally, the ES sends this information to the vehicle, including the matching rate for

(a) RFID reader. (b) Computer connected with the reader.

(c) Vehicle used in the operation tests. (d) RFID tag.

Figure 4: Hardware set-up of the RFID identification prototype.

every point of interest notified. In the example, the hotels which best suit the user's preferences are "AC Elche," "NH Amistad," and "NH Rincon de Pepe," with a matching rate of 43%. This suitability rate is calculated considering the number of matching features between the user's profile and the POI element.

The navigation capabilities of the on-board software are also showed in Figure 5. This feature is currently implemented to support both Google Maps (in the image) and digital cartography. The software draws both previous road problems, received during the hand-off, and the new ones, by means of icons over the road. The polygon which covers the current coverage area is also depicted using a light red line. When the vehicle is close to a road event (initially sent by a vehicle, a roadside sensor or manually fixed by an operator), the application shows a warning about it and the remaining distance to reach the incidence. Further details about the incidence are also given through a spoken alert, by means of a speech synthesizer. In the example, a warning about the proximity of a breakdown on the road has been raised. This event is represented as a yellow mark on the map. As can be seen, the user is always informed of a notification by means of the warning icon, the spoken alert, and the navigation map. The central text panel also notifies traffic incidences from other vehicles (V2V) or roadside sensors (I2V). Messages received from the information service which provides contextual notifications about near POIs are also showed in this part of the interface. All these messages are also read by the integrated speech synthesizer for safety reasons.

In Figure 6 there is a screenshot of the Web application located at ITOS, implemented using JSP. In the map view provided, a road operator has logged in and it is currently reading information about a meteorology event reported by a vehicle in the surroundings of Madrid. Road incidences are depicted over a Google Maps interface using different marks for each incidence type. All road event types can be seen on the left part of the window. Further information about incidences can be accessed by clicking on the icon. A normal user or driver is only allowed to see information about the road network, but the operator can also insert, delete, or modify events and manage the users registered in the system. Each user can, nevertheless, change his own profile via this application. Among other views, there is a specific one dedicated to edit users' profile by means of several templates, enabling them to change their preferences. In this manner they can vary the information received from the infrastructure by means of the adaptation scheme previously explained. To date, several features can be selected about several kinds of POI, such as cinemas, restaurants, petrol stations, and service areas.

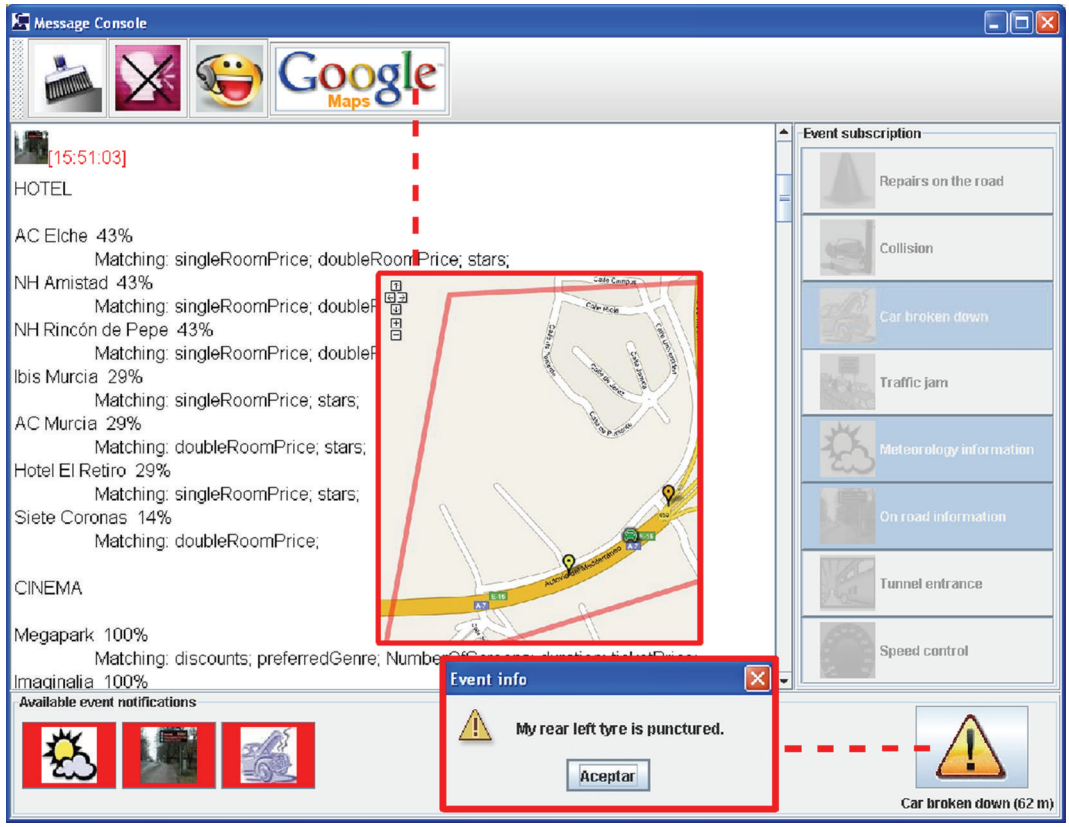


Figure 5: On-board software with traffic and POI notification services.

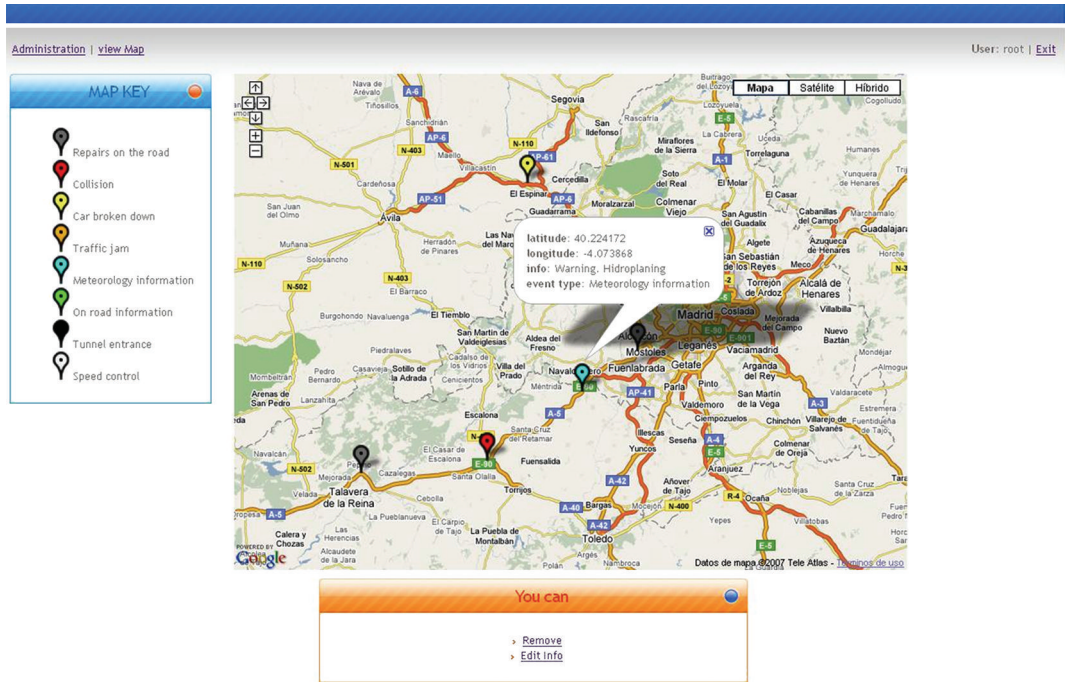


Figure 6: Road incidences monitoring at the ITOS Web application.

7. Performance of the Information Adaptation Subsystem

One key aspect when evaluating the information adaption subsystem is the amount of time employed during the inference and delivery of interesting notifications to the driver. In vehicular environments, the notifications must be timely sent to the vehicle before it leaves the coverage area related to this information. Regarding the performance of the overlay network, an exhaustive analysis under real conditions is given in [31]. Here, it has been proven the feasibility of the vehicular network to send messages following V2V, V2I, and I2V communication patterns. According to collected results, 95% of delay values are under 500 miliseconds, although the V2V case presents a greater jitter, because both the uplink and the downlink channels of UMTS are used. For the evaluation of the information adaptation subsystem, only the I2V link is used, but the performance study is directed to the time needed for interring contextual notifications to users.

In order to evaluate the performance of the information adaptation subsystem, we have developed a model that contains enough information to deal with real situations and different users' profiles. This model is composed of eleven concepts in the Profile ontology, which represent the different users' preferences about POIs (type of fuel, food likes, preferred cinema genre, etc.). Moreover, eighteen concepts in the Environment ontology represent the available POIs in this model. These can be seen in Figure 2. An Environment Server has been set up to simulate the area where the car is currently driving along. This server contains one hundred and ten instances of POI concepts, such as instances of restaurants, cinemas, and hotels. Moreover, thirty-two context rules have been defined to match the drivers' profiles with these points of interest. Four different instances of user profiles have been proposed to test the system responses. Each profile is centered on different types of services. Thus, P_r is more keen on restaurants, P_c on cinemas, P_h on hotels, and P_m is a combination of these three. The performance test has been divided into three steps.

(1) *The initialization of the knowledge base.* In this step, the Environment and Profile ontologies are retrieved in the ES from the permanent storage device, together with the POI instances. Moreover, the specific drivers' profiles (instances of the Profile ontology) for the scenario are also retrieved, together with the context rules associated to these profiles. Ontologies, instances, and context rules are locally stored in the ES.

(2) *Inference process.* The context rules are applied over each of the drivers' profiles in order to match them with the relevant POIs in the Environment ontology.

(3) *Collection of matches.* In this step, the POIs matched with each driver's profile are retrieved and notified to the correspondent driver, along with the matching ratio according to each profile.

Table 4: Evaluation results of the system performance.

Profile	Knowledge Base Initialization (ms)	Inference Process (ms)	Collect Matches (ms)
P_r	2484	250	62
P_c	2625	281	63
P_h	2890	313	47
P_m	3410	381	105

The Environment Server employed for this test is an Intel Pentium D 3 Ghz with 1 GB of RAM running under Linux, with the same Java and Jena set-up considered in the final platform. The previous scenario has been simulated with this equipment to easy the performance study under consideration, but the whole platform has been checked in real conditions, as can be seen in [31].

Table 4 shows the obtained results for the four profile tests performed. The most expensive operation is the initialization of the knowledge base, normally two or three seconds. However, since this process is performed once during the ES setting-up, it does not affect to the normal operation of the system. The table reflects that the initialization time depends on the knowledge base size. As a result, since P_m contains the profiles used in the three previous tests, the initialization carried out with this profile takes longer. More relevant are the results of the inference process and the collecting matches tasks, both critical processes when a vehicle is driving through an ES area. The inference process is more time-consuming than the collection of matches, due to the triggering of rules on the whole knowledge base. Contrarily, the collection of matches is only executed on the POIs retrieved by the inference process. In this task, the fetched POIs are decreasingly ordered according to the matching ratio. Finally, they are parsed into a text string and sent to the vehicle. The results in Table 4 show that the inference process takes longer as the profile is more complex, but it still stays within acceptable levels. Furthermore, the time consumed in the collecting matches task depends on the number of matches found, always being lower than the time in inference process.

Regarding the specific values in Table 4, the results for the profiles P_r, P_c, and P_h are similar, although the time for each task slightly increases from restaurant profile P_r to cinema profile P_c and from P_c to hotel profile P_h. Taking into account that the ontology model and the number of instances (i.e., hotels, restaurants, cinemas, etc.) are equal in the three cases, the differences stem from the driver's preferences. Examining the correspondent preferences with respect to the available POIs, it is detected that seven restaurants are matched with the available POIs, while the number of matching cinemas and hotels is eight and ten, respectively. Moreover, when matching the POIs in the cinema profile, it is necessary to check all the available movies in each one, which are represented by a relationship to the concept `Film` in the model (see Figure 2). Likewise, the higher number of matches of hospitals in profile P_h increases the time of inferencing and collecting matches. In relation to profile P_m,

it contains the three previous profiles. As a result, the times for initialization, inference, and collection of matches are higher than in the rest of profiles. This increase is particularly higher in the last task, because the number of matches is considerably higher than in the previous cases. However, the resulting times for the two critical tasks in the information adaption subsystem, namely, inference and collection of matches, are below 400 miliseconds in all cases, thus being acceptable for any vehicle in an ES coverage area.

8. Conclusions and Future Work

The work presented in this paper covers an integral design of a telematic infrastructure for ITS services deployment oriented to the provision of contextual information. A communication architecture has been designed, developed, and extensively evaluated in a previous work [31]. The additional support of the infrastructure side is a key factor to offer global processing and contextual integration. The overlay network designed is based on the concept of P2P communication groups, which are used to bound the propagation of messages generated by vehicles and the ones received from the infrastructure in each service area. Vehicles perform a handover process through these areas in order to maintain a communication channel with a concrete service. This idea has been exploited through a network platform which comprises both the on-board unit and several entities located at the road side and the core infrastructure. The UMTS cellular network is used as communication technology, since continuous improvements in operator's infrastructures encourage its use in many ITS services.

Vehicle integration in the traffic context is a key work in the paper. In addition to the event-based communication channel, which enables vehicles and road-side units to exchange local information, a complementary support at the infrastructure side offers global processing capabilities and information adapted according to the user's preferences. A Web-based system integrates digital cartography functionalities to allow users and road operators to monitor the road network state. Moreover, an ontology-based reasoning subsystem adapts contextual information according to the user's profiles. This part of the work has been extended with an RFID-based identification platform, in order to detect the vehicle presence at specific locations and send adapted POI notifications. One of the two new software modules added in the on-board unit presents a user interface able to warn the user about traffic incidences and show POI notifications received from the infrastructure.

Considering the whole prototype of the platform presented here, it is appreciable the work needed to integrate the multiple technologies used in the architecture. In the operation tests carried out in final stages of the project, it has been proven that all functionalities identified in the paper work correctly, although further work is needed to accomplish a final solution ready for the massive usage in such a dynamic environment as the transportation one. These technical improvements are focused on providing a secure message passing and reliable communications, upgrading the P2P network to be proof against UMTS discontinuities.

As can be seen, the telematic platform is able to support a wide range of ITS services related to information provision. In fact, several reference services have been included in the implementation. In next steps, this platform will be updated according to some ongoing and future research lines. New advances in cellular communications have recently appeared (such as HSUPA, or High-Speed Uplink Packet Access), and some others are going to do so in next months. Hence, new experimental tests and platform upgrades will be done regarding the communication architecture. The MBMS technology is expected to present a great technological advance for traffic information provision. The advantages of this technology could be combined with a group-based P2P scheme as the one presented. WiMAX evaluation is an ongoing work at the University of Murcia, and since the proposed overlay network is independent on the communication technology, the integration of both concepts is an interesting future line. At the service level, current works are directed to take advantage of digital cartography to provide extended services at both the vehicle and infrastructure sides.

Acknowledgments

The authors would like to thank the Spanish Ministry of Education and Science for sponsoring this work under the Grants TIN2008-06441-C02-02 (in frames of the SEISCIENTOS project) and AP2006-4154 (in frames of the FPU Program) and the Spanish Program to Aid Groups of Excellence of the Séneca Foundation, under Grant 04552/GERM/06.

References

[1] S. Fuchs, S. Rass, and B. Lamprecht, "Context-awareness and collaborative driving for intelligent vehicles and smart roads," in *Proceedings of the International Workshop on ITS Ubiquitous Roads, IEEE Global Information Infrastructure Symposium*, Marrakech, Morocco, July 2007.

[2] W. Kiess, J. Rybicki, and M. Mauve, "On the nature of inter-vehicle communications," in *Proceedings of the Workshop on Mobile Ad-Hoc Networks*, pp. 493–502, Bern, Switzerland, February-March 2007.

[3] S. Biswas, R. Tatchikou, and F. Dion, "Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety," *IEEE Communications Magazine*, vol. 44, no. 1, pp. 74–82, 2006.

[4] C. Wewetzer, M. Caliskan, K. Meier, and A. Luebke, "Experimental evaluation of UMTS and wireless LAN for inter-vehicle communication," in *Proceedings of the 7th International Conference on Intelligent Transport Systems Telecommunications (ITST '07)*, pp. 287–292, Sophia Antipolis, France, June 2007.

[5] J. Rybicki, B. Scheuermann, W. Kiess, C. Lochert, P. Fallahi, and M. Mauve, "Challenge: peers on wheels—a road to new traffic information systems," in *Proceedings of the Annual International Conference on Mobile Computing and Networking (MOBICOM '07)*, pp. 215–221, New York, NY, USA, September 2007.

[6] M. Aoyama and H. Takeichi, "Adaptive self-organizing overlay network for car-to-car communications," in *Proceedings of the 9th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD '08)*, pp. 605–610, Phuket, Thailand, August 2008.

[7] T. Hoßfeld, K. Tutschku, F.-U. Andersen, H. De Meer, and J. O. Oberender, "Simulative performance evaluation of a mobile peer-to-peer file-sharing system," in *Proceedings of the Next Generation Internet Networks: Traffic Engineering (NGI '05)*, pp. 281–287, Rome, Italy, April 2005.

[8] Sun Microsystems, *JXTA Technology: Creating Connected Communities*, 1st edition, 2005.

[9] S. Krco, D. Cleary, and D. Parker, "P2P mobile sensor networks," in *Proceedings of the Annual Hawaii International Conference on System Sciences*, pp. 324–332, Kohala Coast, Hawaii, USA, January 2005.

[10] B. M. Masini, C. Fontana, and R. Verdone, "Provision of an emergency warning service through GPRS: performance evaluation," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems (ITSC '04)*, pp. 1098–1102, Washington, DC, USA, October 2004.

[11] G. Jodlauk, Q. Mu, and G. Gehlen, "Cellular communication based mechanism for road traffic information distribution," in *Proceedings of the ITS World Congress*, New York, NY, USA, November 2008.

[12] S. Goel, T. Imielinski, K. Ozbay, and B. Nath, "Grassroots—a scalable and robust information architecture," Tech. Rep. DCS-TR-523, Rutgers University, North Brunswick, NJ, USA, June 2003.

[13] C. Bisdikian, I. Boamah, P. Castro, et al., "Intelligent pervasive middleware for context-based and localized telematics services," in *Proceedings of the ACM International Workshop on Mobile Commerce*, pp. 15–24, Atlanta, Ga, USA, September 2002.

[14] A. Sáez Esteve, V. R. Tomás López, J. J. Samper Zapater, and L. van der Berg, "Improving a traffic operator interface using an ontology and intelligent rules," in *Proceedings of the Euro American Conference on Telematics and Information Systems (EATIS '07)*, pp. 1–8, Faro, Portugal, May 2007.

[15] L. Deirdre and M. Rene, "Primary-context model and ontology: a combined approach for pervasive transportation services," in *Proceedings of the 5th Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom '07)*, pp. 419–424, New York, NY, USA, March 2007.

[16] C. Schlenoff, R. Washington, and T. Barbera, "An intelligent ground vehicle ontology to enable multi-agent system integration," in *Proceedings of the International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, pp. 169–174, Waltham, Mass, USA, April 2005.

[17] M. Golemati, A. Katifori, C. Vassilakis, G. Lepouras, and C. Halatsis, "Creating an ontology for the user profile: method and applications," in *Proceedings of the International Conference on Research Challenges in Information Science*, pp. 407–412, Ouarzazate, Morocco, April 2007.

[18] R. Toledo-Moreo, J. Santa, M. A. Zamora-Izquierdo, B. Ubeda, and A. F. Gomez-Skarmeta, "A study of integrity indicators in outdoor navigation systems for modern road vehicle applications," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, Nice, France, September 2008.

[19] A. K. Dey, G. D. Abowd, and D. Salber, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Human-Computer Interaction*, vol. 16, no. 2–4, pp. 97–166, 2001.

[20] T. Berners-Lee, J. Hendler, and O. Lassila, *The Semantic Web*, Scientific American, 2001.

[21] G. Klyne and J. J. Carroll, "Resource Description Framework (RDF): Concepts and Abstract Syntax," February 2004, http://www.w3.org/TR/rdf-concepts/.

[22] M. Dean, D. Connoll, F. van Harmelen, et al., "Web ontology language (OWL)," Tech. Rep., W3C, February 2004.

[23] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, Cambridge, UK, 2003.

[24] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson, "Jena: implementing the semantic web recommendations," in *Proceedings of the 13th International World Wide Web Conference*, pp. 74–83, ACM Press, May 2004.

[25] T. Strang and C. Linnhoff-Popien, Eds., "Proceedings of the 1st International Workshop on Location- and Context-Awareness (LoCA '05)," vol. 3479 of *Lecture Notes in Computer Science*, Oberpfaffenhofen, Germany, May 2005.

[26] K. Brooks, "The context quintet: narrative elements applied to context awareness," in *Human Computer Interaction International Proceedings*, Erlbaum Associates, Crete, Greece, 2003.

[27] A. Muñoz, A. Vera, J. A. Botia, and A. F. G. Skarmeta, "Defining basic behaviours in ambient intelligence environments by means of rule-based programming with visual tools," in *Proceedings of the European Conference on Artificial Intelligence*, Riva de Garda, Italy, August-September 2006.

[28] A. F. Skarmeta, H. M. Barbera, M. Z. Izquierdo, B. U. Minaro, F. C. G. de Leon, and L. M. T. Balibrea, "Mimics: exploiting satellite technology for an intelligent convoy," *IEEE Intelligent Systems*, vol. 17, no. 4, pp. 85–89, 2002.

[29] J. Santa, B. Úbeda, R. Toledo, and A. F. G. Skarmeta, "Monitoring the position integrity in road transport localization based services," in *Proceedings of the IEEE Vehicular Technology Conference (VTC '06)*, pp. 2801–2805, Montreal, Canada, September 2006.

[30] J. Santa, B. Úbeda, and A. F. G. Skarmeta, "A multiplatform OSGi based architecture for developing road vehicle services," in *Proceedings of the 4th Annual IEEE Consumer Communications and Networking Conference (CCNC '07)*, pp. 706–710, Las Vegas, Nev, USA, January 2007.

[31] J. Santa and A. F. Gomez-Skarmeta, "Potential of cellular networks in vehicular communications," in *Proceedings of the ITS World Congress*, New York, NY, USA, November 2008.