ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/272576108

Modelling and analysis of non-cooperative peer-assisted VoD streaming in managed networks

Article *in* Multimedia Tools and Applications · February 2015

DOI: 10.1007/s11042-015-2477-9

CITATIONS
6

READS
112

2 authors:

Saso Gramatikov
Ss. Cyril and Methodius University in Skopje
32 PUBLICATIONS 60 CITATIONS

SEE PROFILE

Fernando Jaureguizar
Universidad Politécnica de Madrid
126 PUBLICATIONS 1,048 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project puntos de calor View project

Project F-NLP: Natural Language Processing in Finance View project

All content following this page was uploaded by Saso Gramatikov on 22 February 2015.

The user has requested enhancement of the downloaded file.

Manuscript
Click here to download Manuscript: article v2.tex
Click here to view linked References

Multimedia Tools and Applications manuscript No.
(will be inserted by the editor)

Modelling and Analysis of Non-cooperative Peer-assisted VoD Streaming in Managed Networks

Sasho Gramatikov · Fernando Jaureguizar

Received: date / Accepted: date

Abstract The growing popularity of the Video on Demand service in the Internet Protocol Television environments and the demand for increased quality of the offered videos are becoming a serious threat for the service providers because the high amounts of video traffic are causing congestion in the delivery networks. One of the most acceptable approaches to solve this issue is the peer-assisted streaming, where the peers participate in the streaming process in order to alleviate the load on the streaming servers and in the core of the network. Although the reliability of the Peer-to-Peer service is considerably improved in the managed networks because of the control that the operators have over the clients' Set-Top Boxes, the failures of the peers still cannot be completely eliminated. The operator can take advantage of the streaming and storage resources of the clients and use them for peer-assisted streaming only while they are watching a video, but not after they finish the streaming session because they may turn off their receiving devices until the next session. In this chapter, we address the issue of the failures of the peers in such environments and their influence on the traffic requested from the servers for providing uninterrupted video experience. For that purpose, we propose a precise mathematical tool for modelling a peer-assisted system for Video on Demand streaming in managed networks with non-cooperative peers, which may decide not to share their resources while they are not active. This tool calculates the performance of the system taking into consideration large variety of system parameters, including the failure probability and the time the peers spend until they decide to turn on the STB and join the network. As the

S. Gramatikov
Faculty of Computer Science and Engineering,
University "Ss. Cyril and Methodius", 1000 Skopje, Macedonia
E-mail: sasho@finki.ukim.mk

F. Jaureguizar
Grupo de Tratamiento de Imágenes, ETSI Telecomunicación
Universidad Politécnica de Madrid, 28040-Madrid, Spain

results from the simulations verify the correctness of the mathematical model, we use it to analyse how the failures of the peers are affecting the system's performance for different system parameters.

Keywords Peer-assisted streaming · VoD · Stochastic model · Failures · Performance

1 Introduction

The advances of the network technologies and the high popularity of the videos made a solid ground for the video domination in the global consumer traffic. The video contents are becoming so popular that, according to the estimations, by the year 2017, they will occupy 80-90% of the globally exchanged traffic worldwide [9]. A significant part of this traffic will belong to the Video on Demand service (VoD) which, according to the same estimations, will triple the amount of traffic that it generates nowadays. The main reason for the popularity of the VoD service is that it offers a great convenience to the clients to watch a large variety of videos at any time. Moreover, it has become more accessible to a larger community as a result of the expansion of the Internet Protocol Television (IPTV), which, apart from the linear TV, started to offer VoD service. However, the delivery of these traffic-intensive contents requires a separate unicast flow for every request of the clients, which is a serious burden for the delivery network and the streaming servers. In addition, the demand for higher quality videos and the growing popularity of the service further increases the amount of the traffic which threatens to congest the delivery networks. Therefore, finding an optimal way to deliver the videos in the network has been a challenging task for the network operators and the research community.

One of the approaches used to solve the problem of network congestion is the Peer-to-Peer (P2P) concept. As it proved to be a successful solution for sharing large files in the Internet, this concept has encouraged many researchers to consider its implementation in live and VoD streaming in the Internet [24, 23]. However, the main issue of the P2P for delivery of video contents is the real-time nature of the VoD service, i.e., the blocks of video data have to be provided in consequent order, at the moment when they are requested. This requires reliable links and participation of the entire community of peers, otherwise, the viewing experience will suffer interruptions and long waiting times. However, in the Internet based solutions, the participation of the peers and the Quality of Service (QoS) cannot be guaranteed because. The participation of the clients largely depends on the clients' will to cooperate. The reliability issues are partially overcome in the managed networks where a desired level of QoS can be provided, and the human factor of participation in sharing of the contents can be significantly reduced because the operators have the control over the users' Set-Top Boxes (STBs) and can easily reserve part of their storage and uplink capacity for streaming purposes. Another convenience of the STBs is that they are becoming cheap storage devices with capacity to store large amounts of data. These facts make the managed networks a suitable environment for implementation of P2P distribution of video contents. The conditions under which the P2P concept becomes beneficial for providing IPTV services are studied in [5, 6, 4].

A more common practice for reducing the traffic in the managed environments is the use of the peer-assisted streaming, where the peers take an important role in the streaming. However, although the peers participate in the streaming process, the servers still remain an inevitable part of the system that guarantees the required QoS. The peer-assisted streaming combined with various content distribution schemes prove to be efficient in reducing [7, 3, 12, 14, 37, 20, 16, 10, 11] and even eliminating the traffic requested from the streaming servers [18, 17]. While some of these systems assume that the peers are perfectly reliable, the authors in [3], deal with the possibility that some of the peers may be failed at the moment of choosing peers to assist in the streaming. In [7, 37], the authors go one step further by considering the cases when some of the peers may fail while they are streaming to the other peers. This situation is handled by the streaming servers which compensate the remaining parts of the videos originating from the failed peers. Moreover, in order to better utilize the streaming resources of the peers, the systems in [11, 16] first try to redirect the peers that are receiving videos from failed peers to other available peers in the system, and, if there are no such peers, they are eventually redirected to the streaming servers. Although these systems treat the failures of the peers, they only consider the contribution of the peers to the improvement of the performance, ignoring the degradation of the service as a result of the failures of the peers.

In the real managed environments, although the operator has the control over the STBs, the resources cannot be guaranteed with certainty because the clients are those who have the final decision on whether to participate in the streaming. The clients might decide to turn off their STB after watching a video and keep them off until their next session. During this period of time, their streaming and storage resources are unavailable, which is a major concern for the QoS and the performance of the entire system. In order to provide uninterrupted streaming experience, similarly like in [7, 37], we propose a system where all the interrupted streams are immediately compensated by the streaming servers. However, unlike the previous systems, the main question that is addressed in this work is how the failures of the peers and the time it takes them to recover will affect the overall performance, especially the load of the servers. On one hand, the interruptions of the streams due to failures of the peers increase the server traffic because of the additional compensation streams, but on the other hand, the failures of the peers reduce the number of requests in the system which implies less traffic on the servers. Hence, one of the objectives of this chapter is to determine the amount of extra traffic requested from the servers due to the compensation for the interrupted streams and the amount of reduced traffic due to the reduced number of requests.

In our previous work [15], we have also proposed a system for peer-assisted VoD contents in managed networks, but its main weakness was the assumption that the peers are fully reliable. In that work, we focused on the design of a mathematical model for the proposed system and used it to analyse how some environmental parameters influence the performance. However, the model cannot be a representative of a real system since it does not take into consideration the failures of the peers. Therefore, our main objective of this work is to include the failures of the peers in the basic mathematical model and analyse their influence on the overall system's performance.

There are many mathematical models of different P2P systems which aim to give a precise estimation of the behavior of the systems. In [13, 28, 29], the authors propose mathematical models for pure P2P networks for sharing of a general type of contents. A model for sharing contents between multiple P2P communities connected in one network is proposed in [25]. Since the video streaming is different from content sharing, the authors of [22] define a mathematical model for live P2P streaming and the authors of [26] propose P2P mash-based analytical model for VoD streaming. A mathematical model for peer-assisted streaming is proposed in [36], but it only establishes a relationship between the peers' storage capacity, the network size and the off-loaded traffic on the servers and it considers peers with streaming capacity higher than the playback rate of the videos. In [31, 30], the authors present analytical studies of the streaming capacity growth in large scale P2P systems. Although they consider peers with limited streaming capacity and analyze the effect of the failures on the overall streaming capacity of the system, they do not treat the problem of additional traffic in the network for the compensation of the interrupted streams. All these models do not take into consideration the distribution of the contents among the peers and refer to large-scale systems. To the best of our knowledge, there is no mathematical model of a peer-assisted streaming in managed network which, apart from the distribution of the contents in the peers with a streaming capacity lower than the playback rate of the videos, takes into account the failures of the peers and their effect on the system's performance. Therefore, in this work, we propose a mathematical model, which is an extension of our previous work [15] and includes the probability that the clients will fail after they receive the entire content and the time it takes the failed peers to recover. We use this model to give the answers on how the failures of the peers influence the performance of the systems.

The rest of the chapter is organized as follows: After the description of the system for peer-assisted streaming with non-cooperative peers in Section 2, in Section 3, we present the details of the representation of the described system as a stochastic mathematical model. The verification of the model and analysis for various system parameters are presented in Section 4. Eventually, we give the conclusions from our work in Section 5.

2 System overview

The system for peer-assisted streaming is aimed for IPTV service providers which use Asymmetric Digital Subscriber Line (ADSL) technology to deliver the video contents to the clients. It is a hybrid solution that unites the high reliability and scalability of the IPTV architecture and the storage space and unused uplink bandwidth of the P2P architecture. The system consists of streaming servers placed in the network core and clients with STBs connected to a Digital Subscriber Line Access Multiplexer (DSLAM) that gives them access to the streaming servers. The main streaming functionality is provided by the streaming servers. They have limited storage and streaming capacity, but as a whole, they host the entire video library and can serve any request from the clients. The clients, apart from requesting videos, can also assist in the streaming process. In our system, the clients are non-cooperative, i.e., they can decide to turn off their STB after they watch the requested video. Unlike the P2P networks where the clients have the control of the contents they share and the resources they offer to the community, in the managed network that we consider, the STBs are owned by the service provider. Therefore, as long as they are active, part of their unused storage and streaming capacity can be reserved for the needs of the peer-assisted streaming of contents distributed by the service provider. It is important to emphasize that, although the peers take part in the streaming of videos, the streaming servers still have the main role in the entire streaming process. The peers are used only to alleviate the load of the streaming servers and to reduce the network traffic for distribution of the videos from the servers to the clients.

Because of the limited number of connections of a DSLAM, the peers are grouped in local communities. Each peer can assist only in the streaming of the clients within the same local community. The clients from different communities cannot assist in each others streaming because, the cross-community traffic would cause additional burden in the core of the network. The size of a local community is n peers. We assume that the number of peers in the local community subscribed for the VoD service is constant within time period necessary for reaching a stationary state, i.e., there are no new contracts and no terminations of the existing contracts for VoD service. This assumption does not mean that the number of active peers in given time is constant. On the contrary, this number is variable since a peer may be active, but it can also fail once it receives the video it requested. However, although a peer might fail, it is still part of the local community. The failed peers do not leave community, but they temporarily turn off their STB and rejoin the community when they turn on their STBs to watch new contents. Therefore, the size of the local community composed of active and failed peers is considered constant. If the overall number of peers in the community changes, then the results for estimating the system's performance should be recalculated with the new value of n.

In our system, the peers have less streaming capacity than the playback rate r of the videos since the current ADSL technology cannot offer upload rates equal to the rates necessary for streaming the required quality of videos. In order to include such peers in the streaming, the videos are divided into m parallel strips. Each strip contains equidistant portions of the video. A video can be played only when all the strips are gathered and assembled into one stream. The division of the contents enables independent streaming of different strips of the same video from multiple peers. The time necessary to stream one strip is equal to the time necessary to stream the entire video, however, the streaming occupies m times less uplink capacity on the peers. The capacity r/m necessary to stream one strip is defined as a channel. In that sense, the streaming capacity k of each peer is expressed as the number of channels that it can simultaneously stream, and the storage capacity s is expressed as the number of strips that it can store. Since the provider has the control over the peers, it can define the size of the resources it reserves for peer-assisted streaming. Therefore, the number of streaming channels k and the number of strips that a peer can store s are determined by the operator and are equal for each peer. Although a peer might have more available channels and storage capacity, the operator uses only k channels and s strips storage capacity for the purpose of the peer-assisted streaming.

The streaming servers are generalized with one server placed at the edge of the network, called Edge Server (ES). Since all the streaming servers have the capacity to store the entire video library and to serve any request, the generalized ES has the capacity to serve any request from its local community. We use this generalization since the main purpose of this work is to analyse the influence of different system parameters on the overall amount of traffic served by the servers in general, not a specific server. The ES is the entry point of each request from one local community. Apart from the role of a streaming server, in order to provide the P2P service, the ES has additional role of an index server which keeps up-to-date information about the availability of the contents on the peers and their current available uplink capacity. Whenever it receives a request, it first checks whether there are peers with available channels that host strips of the required video. Then, it sends a list of peer-strip pairs to the requesting peer. Each item of the list contains the number of the strip of the video and the address of the peer that will serve that strip. Afterwards, the peer establishes independent connection with each peer from the list and initiates streaming session. If the peers cannot completely provide service for all the strips of the video, the remaining strips are served by the ES.

The video content library contains c videos stored both in the peers and the servers. In the peers, the videos are distributed in strips. One peer hosts different strips of different videos. This approach increases the availability of the videos since the peers can store a larger variety of contents. The videos are distributed in the peers in the off-peak hours. The size of the video library c is constant. We assume that the operator does not insert new videos or delete the existing ones on short terms. The probability that a content will be chosen to be stored in a peer depends on the distribution scheme implemented by the service provider and is noted as $P_x(v)$. Although divided into strips, the videos are stored entirely in the servers so that any number of requested strips of a single video can be streamed. The servers host the entire video library.

The peers are generating requests for videos according to the Poisson process with arrival rate λ. The videos from the video library are requested with different probability $P(v)$, where v is the popularity rank of the video within the library. The process of streaming the strips also obeys the Poisson process with service rate $\mu = 1/d$, where d is the average time a client spends watching a video. When the client finishes the streaming session, it decides with probability p_{off} to turn off the STB. This probability is called failure probability. The peers can fail only after the end of a session because we assume that the clients are interacting with their STB only when they decide to watch a video, during the video session and at the end of the session. They do not fail during the session since that failure would be considered as an end of a video session, which is already defined with the service rate μ. If they decide not to turn off the STB, they will not interact with it until their next video session.

After the failure, the average time the client spends until it turns on the STB is T_{off}, also called recovery time. After the recovery, the client requests another video. The recovery time has exponential distribution, and therefore, the process of the recovery of the peers is also modelled as a Poisson process with service rate $\mu_{off} = 1/T_{off}$, also referred to as recovery rate.

An overview of the parameters used in the model is given in Tab. 1.

As the peers are non-cooperative, they may decide not to share their resources after they finish their video session. The main issue that causes this behaviour is that all the streams originating from these peers will be interrupted and the video service will be deteriorated. To prevent this scenario, the system is monitoring the state of the STBs in the period after their streaming session ends, and, if it detects that a peer has failed, it runs a procedure to compensate all the streams that were streamed by the failed peer. In order to avoid long delays for the new recovery streams to begin, the system puts the reliable streaming servers in charge of finishing the interrupted streams. The system excludes the other active peers to compensate the interrupted streams because they also might fail at certain point of the streaming and cause additional processing from the ES. For every failure or recovery of a peer, the ES updates an availability table which is used in the request process for redirecting the requests to other peers in the local community.

The utilization of this model raises a discussion related to the effects that the failures have on the traffic on the servers, especially in the cases of decreased reliability of the clients. On one hand, the failures of the peers increase the load on the servers because the failure of one peer means also interruption of all the strips it is streaming to other peers. In order to provide uninterrupted video service, the remaining content of the strips that were streamed by the failed peer are assigned to be streamed by the server. On the other hand, the failure of the peers implies fewer requests for videos and thus less traffic on the servers. Hence, one of the goals of chapter is to determine the amount of extra traffic on the servers generated for the compensation of the strips streamed by the failed peers and the amount of reduced traffic due to the reduction in the

n	Size of a local community
n_{idle}	Number of idle peers
n_{rcv}	Number of receiving peers
n_{off}	Number of failed peers
n_{act}	Number of active peers
m	Number of strips per video
k	Number of streaming channels per peer
s	Number of strips that can be stored on a peer
c	Size of the video content library
μ	Average rate of service of videos by one peer or server
d	Average duration of a video session
λ	Average arrival rate of video requests generated by one client
w	Average time a peer waits to make a request
p_{off}	Failure probability of a receiving peer
μ_{off}	Average recovery service rate of a failed peer
T_{off}	Average recovery time of a failed peer
τ_{off}	Average recovery time of a peer
$\mathbf{k}(t) = (i, j, l)$	State vector of the system at time t, with i occupied channels on the peers, j occupied channels on the servers and l failed channels
$P_{i,j,l}(t)$	Probability that the system is in state (i, j, l) at time t
$P^{+}_{p,i,j,l}(\Delta t)$	Probability that there will be one arrival on the active peers within period Δt when the system is in state (i, j, l)
$P^{-}_{s,i,j,l}(\Delta t)$	Probability that there will be one departure on the server within period Δt when the system is in state (i, j, l)
$\lambda_{p,i,j,l}$	Average system arrival rate of requests that arrive at the peers when the system is in state (i, j, l)
$\mu_{s,i,j,l}$	Average system video service rate on the server when the system is in state (i, j, l)
$p_{i,j,l}$	Stationary probability that the system is in state (i, j, l)
$P_{p2p}(i, l)$	Probability that a request will be served by a peer when there are i busy channels on the peers and l failed channels
$P(v)$	Probability that a request for item v will be generated in the system
$P_x(v)$	Probability that item v is chosen to be stored on the peers
$w(t, z)$	Probability that z available channels are distributed on exactly t peers
η	Utilization of the streaming capacity of the active peers
θ	Portion of streaming traffic served by the active peers
Φ	Portion of traffic served by the server relative to the the maximum system's throughput

Table 1 Overview of the parameters used in the model.

number of requests. The answers to these questions will be given with the aid of the stochastic model.

3 Mathematical model description

In this section, we present the steps that lead to the mathematical model for the peer-assisted streaming model defined in the previous section. We model the system as a network of queues and present the dependencies of its state probabilities as a system of linear equations. Then, we thoroughly explain the steps for calculating the system's coefficients and, after solving it, we use the obtained results to calculate the utilization of the streaming capacity of the peers and the portion of the overall traffic streamed by the peers and the servers.

3.1 Representation of the system as a network of queues

The general view of the system is an essential step for obtaining a better picture of the entire process of service with non-cooperative peers. It describes the system viewed as a whole, without entering into the insides of the system, i.e., ignoring the fact that the peers can assist in the streaming. In this view, the system is considered as a set of n peers that generate requests for videos, get served and occasionally fail. Since all of these processes are Poisson processes and the number of peers in a local community is finite and constant, the system can be considered as a closed network of queues with finite population [19]. Each queue of this network of queues represents one of the processes in which the peer can enter: process of waiting to make a request, process of receiving a video or process of failure. The relation of these queues is shown in Fig. 1.

The waiting queue is serving the *idle* peers and its size in stationary state is n_{idle}. Idle peers are all those peers which are waiting to request a video. Although they are not receiving a video, they participate in the streaming of contents to other peers. The peers in the receiving queue are called *receiving* peers because they requested a video, and therefore, are receiving strips of the video. The size of this queue in stationary state is n_{rcv}. The receiving peers are also participating in the streaming of contents to other peers. The idle and the receiving peers are altogether called *active* peers because they both participate in the process of streaming. Their cumulative number is noted as $n_{act} = n_{idle} + n_{rcv}$. The failure queue is "serving" the $failed$ peers and has average size in stationary state n_{off}. The failed peers do not participate in the streaming process, i.e., they neither send nor receive streams.

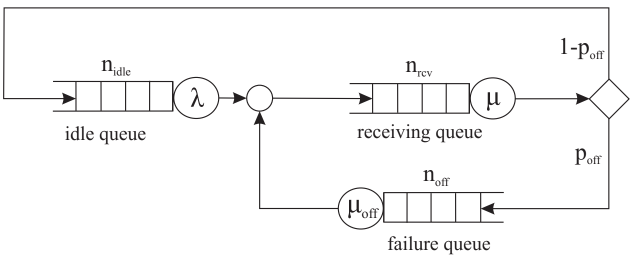


Fig. 1 General representation of the system as a closed network of queues with finite population.

For each request, the ES tries to find peers that have the strips of the required content and have available channel for streaming the strip. The strips that cannot be provided by the peers are streamed by the server. No matter the availability of the contents and the occupancy of the peers, each request is immediately served, i.e., the peer leaves the waiting queue and enters the receiving queue. It stays in this queue until it watches the video, which is in average $d = 1/\mu$, and then, it decides with probability p_{off} whether to turn off the STB. If so, it enters the failure queue, where it stays in average $T_{off} = 1/\mu_{off}$. After this special service, the peer re-joins the community when it turns on the STB to make a new request and enters the receiving queue. This process is also called recovery process. We omit the case that a peer enters the idle queue after the recovery since it is straight forward that a client would not turn the STB on just to leave it idle without watching a video. In the opposite case, the peer decides to keep the STB on with probability $1 - p_{off}$ and let its resources available for serving other peers in the community. It stays in the waiting queue in average $w = 1/\lambda$ until it requests a new content and enters the receiving queue.

A more detailed representation of the system can be given if each peer that makes a request for m different strips of one video is treated as m peers that make independent request for a strip of a different video. In this case, the system is considered to have mn peers that make independent requests of only one strip of video. Applying this approach to the network of queues from Fig. 1 and representing the server and the peers as separate service facilities, the detailed representation of the system gets the form shown in Fig. 2. In Fig. 2, the video service queue is represented by two different queues: a server, represented by an $M/M/mn$ queue and a representative peer which has the streaming and storage capacity of all the peers in the local community modelled as an $M/M/kn$ queue. The probability that the requested strip can be found on a peer that has available channels is P_{p2p} and is a complex function that depends on the current state of the system and the distribution of the contents on the peers. In the detailed representation of the system, the sizes of the waiting and the failure queues are mn_{idle} and mn_{off} since each peer is treated as m peers that request one strip. The failures of the peers give more complexity to the system since once a peer fails, the system has to handle all the streams that were streamed by the failed peer and pass them to the server. Thus, at the moment of a failure, some channels on the server are released because the streams received from the peers are over, but some channels will be instantly occupied for providing uninterrupted service to the peers that were receiving strips from the failed peer.

In order to obtain the portion of the traffic that is served by the peers, we have to determine the average number of occupied channels on the peers and on the server. Although we can obtain the stationary value of the number of idle, receiving and failed peers from Fig. 1, the percentage of requests served by the peers or the server separately cannot be easily determined because of the complexity of calculation of the probability P_{p2p} and the additional traffic burden on the servers caused by the failures.

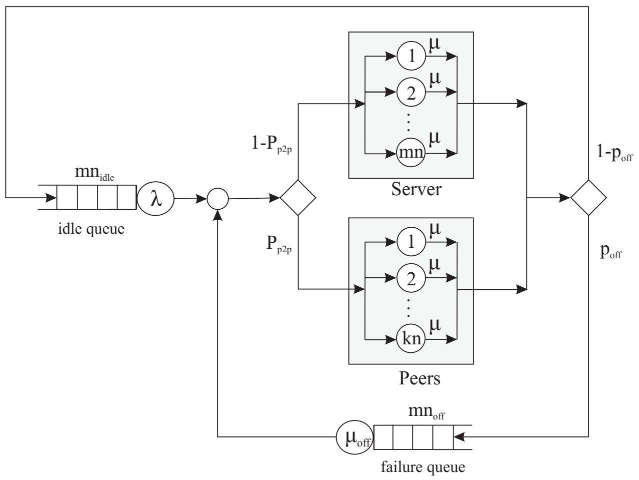


Fig. 2 Detailed representation of the system as a closed network of queues with finite population.

3.2 The system's behavior as a birth-death process

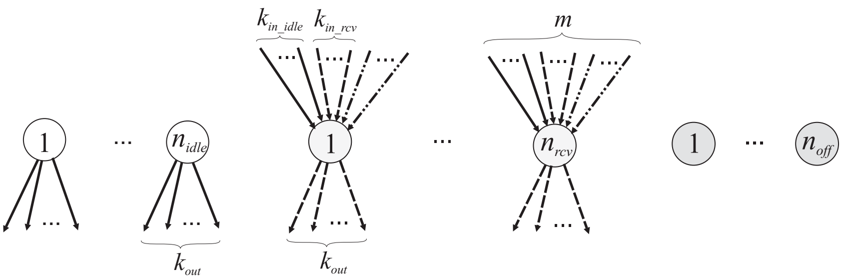
In order to get the average number of busy channels of the active peers and the average number of busy channels on the server for serving the active clients and compensating for the failed peers, first, we define the set of states of the system in the time. Since each peer in the network of queues must be in any of the four queues (Fig. 2), the state of the system at time t can be uniquely defined as $\mathbf{k}(t) = (i, j, l)$, where $i \in \{0, 1, ..., kn\}$ is the number of busy channels on the peers, $j \in \{0, 1, ..., mn\}$ is the number of busy channels on the server and $l \in \{0, 1, ..., kn\}$ is the number of failed channels. The values of the state triple must satisfy the condition $i + j + l \leq mn$.

The average number of busy channels on any facility can be calculated if the probability $P_{i,j,l}(t) = P(\mathbf{k}(t) = (i, j, l))$ of every possible state is found. Based on the exponential nature of the service times of all the facilities, the whole system represents a Markov population process, and therefore, in an infinitesimal small time period Δt, there can be only one request, one end of streaming, one failure, one recovery or none of the four events. More precisely, if the system is in a state $\mathbf{k}(t+\Delta t) = (i, j, l)$ at time $t+\Delta t$, also referred to as a central state, only one of the following events could have happened during the interval Δt:

1. the system had been in state $\mathbf{k}(t) = (i, j-1, l)$ and a request for a strip arrived at the server from an idle peer,
2. the system had been in state $\mathbf{k}(t) = (i, j-1, l+1)$ and a request for a strip arrived at the server from a failed peer,
3. the system had been in state $\mathbf{k}(t) = (i-1, j, l)$ and a request for a strip arrived at an active peer from an idle peer,

4. the system had been in state $\mathbf{k}(t) = (i-1, j, l+1)$ and a request for a strip arrived at an active peer from a failed peer,
5. the system had been in state $\mathbf{k}(t) = (i, j+1, l)$ and a streaming of a strip has been completed on the server without failure of the peer,
6. the system had been in state $\mathbf{k}(t) = (i, j+1, l-1)$ and a streaming of a strip has been completed on the server with failure of the peer,
7. the system had been in state $\mathbf{k}(t) = (i+1, j, l)$ and a streaming of a strip has been completed on a receiving peer without failure of the peer,
8. the system had been in state $\mathbf{k}(t) = (i+2, j-1, l-1)$ and a streaming of a strip has been completed on a receiving peer with failure of the peer,
9. the system had been in state $\mathbf{k}(t) = (i, j, l)$ and neither request arrived nor a stream has been completed on any of the service facilities.

The explanations of these conditions are quite straight-forward except the one for condition (8). According to this condition, the central state (i, j, l) can be reached when a receiving peer finishes the reception of a stream from the peers and immediately fails, provided that the system was previously in state $(i+2, j-1, l-1)$. The reason for decreasing the number of channels on the peers for reaching the central state by two, instead by one, is that if one stream that originates from a peer finishes and the peer fails, it means that the peer that fails is receiving peer, which also streams one stream to the other peers in the community. Therefore, besides the finished stream, there is also one interrupted stream due to the failure of the peer. This comes from the fact that if a receiving peer fails, the number of finished strips that come from other receiving peers on the failing receiving peer is the same as the number of interrupted streams.



(a) Idle peers (b) Receiving peers (c) Failed peers

Fig. 3 Illustration of peer-assisted streaming.

This can be proven if we take a closer look at the peer-assisted streaming process in Fig. 3. The figure shows the three different categories of peers: the idle peers which only stream strips, the receiving peers which stream and receive strips and the failed peers which do not participate in any of these activities. The number of outgoing streams k_{out} that originate from the active peers (idle and receiving) is the same, because as long as the peers are active, there is no importance to the ES whether the peers are idle or receiving a stream in the moment when it assigns them to serve a request for a strip. The total number of incoming streams on every receiving peer is the number of strips that composes the video m. From all these input streams, k_{in_idle} originate from the idle peers, k_{in_rcv} originate from the receiving peers and the rest of the streams originate from the server. The total number of outgoing strips from the receiving peers is $n_{rcv}k_{out}$ and the total number of incoming streams that each receiving peer receives from the receiving peers is $k_{in\_rcv} = (n_{rcv}-1)k_{out}/(n_{rcv}-1) = k_{out}$.

This means that when k_{in_idle} streams finish and the receiving peer fails, there will be also $k_{out} = k_{in\_rcv}$ interrupted streams. The failure of these strips will increase the number of failed strips in the system by k_{out} but it will also increase the number of strips served by the servers by k_{out} because the server has to compensate for the interrupted strips. As a conclusion, the end of k_{in_rcv} strips with failure of the peer implies reduction of the overall number of strips streamed by the peers by $2k_{in\_rcv}$ and increment of the number of failed streams and the number of streams served by the server by k_{in_rcv}, i.e., the central state (i,j,l) can be reached only from the state $(i+2k_{in\_rcv}, j-k_{in\_rcv}, l-k_{in\_rcv})$. Substituting $k_{in\_rcv} = 1$ will give the originating state $(i+2, j-1, k-1)$ in the condition (8).

From the conditions (1)-(9), the probability $P_{i,j}(t+\Delta t)$ that the system will be in state $\mathbf{k}(t+\Delta t) = (i,j,l)$ can be expressed as:

$$
\begin{aligned}
P_{i,j}(t+\Delta t) = {} & P_{i,j-1,l}(t)P^{+}_{s,i,j-1,l}(\Delta t) + P_{i,j-1,l+1}(t)P^{+}_{s,i,j-1,l+1}(\Delta t) + \\
& + P_{i-1,j,l}(t)P^{+}_{p,i-1,j,l}(\Delta t) + P_{i-1,j,l+1}(t)P^{+}_{p,i-1,j,l+1}(\Delta t) + \\
& + P_{i,j+1,l}(t)P^{-}_{s,i,j+1,l}(\Delta t) + P_{i,j+1,l-1}(t)P^{-}_{s,i,j+1,l-1}(\Delta t) + \\
& + P_{i+1,j,l}(t)P^{-}_{p,i+1,j,l}(\Delta t) + P_{i+2,j-1,l-1}(t)P^{-}_{p,i+2,j-1,l-1}(\Delta t) + \\
& + P_{i,j,l}(t)\cdot\left(1-P^{+}_{s,i,j,l}(\Delta t)\right)\left(1-P^{+}_{p,i,j,l}(\Delta t)\right)\cdot \\
& \quad \cdot\left(1-P^{-}_{s,i,j,l}(\Delta t)\right)\left(1-P^{-}_{p,i,j,l}(\Delta t)\right)
\end{aligned}
\tag{1}
$$

where the probabilities of any of the above defined events during the interval Δt are marked in a way that the superscript $+/-$ denotes whether an arrival or an end of service has happened, the first subscript denotes the service facility where the event has happened, with notation p referring to a peer and s to a server, and the remaining three subscripts denote the previous state of the system, before the event happened.

The first four lines of the expression refer to the conditions (1) to (8), accordingly, while the end of the expression refers to the condition (9) representing the joint probability that nor arrival of request, neither end of streaming will happen in the system when it is in state $\mathbf{k}(t) = (i,j,l)$.

From the definition of a Poisson process with service rate λ, the probability that exactly one event will happen within an infinitesimal interval Δt is:

$$P(k=1,\Delta t) = \lambda\Delta t e^{-\lambda\Delta t} = \lambda\Delta t + o(\Delta t) \tag{2}$$

If we use (2) to define the probabilities of the possible events in (1), divide the whole expression by Δt, make some rearrangements and let $\Delta t \to 0$, the expression for stationary state will become:

$$\begin{aligned} 0 = {} & \lambda_{s,i,j-1,l} p_{i,j-1,l} + \lambda_{s,i,j-1,l+1} p_{i,j-1,l+1} + \\ & + \lambda_{p,i-1,j,l} p_{i-1,j,l} + \lambda_{p,i-1,j,l+1} p_{i-1,j,l+1} + \\ & + \mu_{s,i,j+1,l} p_{i,j+1,l} + \mu_{s,i,j+1,l-1} p_{i,j+1,l-1} + \\ & + \mu_{p,i+1,j,l} p_{i+1,j,l} + \mu_{p,i+2,j-1,l-1} p_{i+2,j-1,l-1} - \\ & - (\lambda_{s,i,j,l} + \lambda_{p,i,j,l} + \mu_{p,i,j,l} + \mu_{s,i,j,l}) p_{i,j,l} \end{aligned} \tag{3}$$

where $p_{i,j,l}$ is the stationary probability of the state (i, j, l).

3.3 Determining the coefficients of the system of equations for the system's state probabilities

We can solve the system of equations (3) provided that we calculate the departure and arrival rates that lead to the central state.

The coefficient $\lambda_{s,i,j-1,l}$ refers to the arrival rate of requests directed to the server from the idle peers when the system is in state $(i, j-1, l)$. Its value depends on the number of peers in the waiting queue, obtained by subtracting the number of peers in the video service and failure queues from the overall number of peers in the system, and the arrival rate of each request λ. From all these candidate peers in the waiting queue, only those that cannot be served by the peers will be directed to the server. The portion of these requests is $1 - P_{p2p}(i, l)$, where $P_{p2p}(i, l)$ is the probability that a strip can be served by a peer when there are i busy channels on the peers and l failed channels. The probability of finding a peer that will stream a strip is a function of i and l because they are necessary to determine the number of potential active peers which could possibly serve the request in case they have available channels and copy of the requested strip. Eventually, the final value of the intensity of requests directed to the server when the system is in state $(i, j-1, l)$ is:

$$\lambda_{s,i,j-1,l} = (1 - P_{p2p}(i, l))(mn - i - j + 1 - l)\lambda \tag{4}$$

The coefficient $\lambda_{s,i,j-1,l+1}$ determines the arrival rate of requests directed to the server from the failed peers, when the system is in state $(i, j-1, l+1)$. Any of the $l+1$ peers in the failure queue can recover with rate μ_{off} and independently request a strip, however, only a portion $1 - P_{p2p}(i, l+1)$ of them will be directed to be served by the server. Therefore, the intensity of the probability flow from state $(i, j-1, l+1)$ will be:

$$\lambda_{s,i,j-1,l+1} = (1 - P_{p2p}(i, l+1))(l+1)\mu_{off} \tag{5}$$

The coefficient $\lambda_{p,i-1,j,l}$ determines the arrival rate of requests directed to the active peers from the idle peers when the system is in state $(i-1, j, l)$.

It is similar to (4) because the peers that make the requests are those in the waiting queue, but it differs in the portion of requests that will be directed to the peers, which is determined by the probability $P_{p2p}(i-1,l)$ that the peer is able to serve the required strip when the system is in state $(i-1,j,l)$:

$$\lambda_{p,i-1,j,l} = P_{p2p}(i-1,l)(mn-i+1-j-l)\lambda \tag{6}$$

The coefficient $\lambda_{p,i-1,j,l+1}$ determines the arrival rate of requests directed to the active peers from the failed peers when the system is in state $(i-1,j,l+1)$. Its value is obtained by multiplying the rate of recovery of the peers μ_{off} with the size of the failure queue, which is $j+1$, and the probability that the peers can serve the requested strip:

$$\lambda_{p,i-1,j,l+1} = P_{p2p}(i-1,l+1)(l+1)\mu_{\mathit{off}} \tag{7}$$

The coefficient $\mu_{s,i,j+1,l}$ determines the departure rate of finished streams on the receiving peers from the server, without failure of the peer, when the system is in state $(i,j+1,l)$. In this state, there are $j+1$ streams that could possibly end but only a portion of $1-p_{\mathit{off}}$ of them will decide not to fail. Therefore its value will be:

$$\mu_{s,i,j+1,l} = (1-p_{\mathit{off}})(j+1)\mu \tag{8}$$

The coefficient $\mu_{s,i,j+1,l-1}$ determines the departure rate of finished streams on the receiving peers from the server with failure of the receiving peer, when the system is in state $(i,j+1,l-1)$. Its value is similar to the value obtained in the previous expression, with the difference that the portion of candidates that would finish the stream with failure is equal to the probability of failure p_{off}:

$$\mu_{s,i,j+1,l-1} = p_{\mathit{off}}(j+1)\mu \tag{9}$$

The coefficient $\mu_{p,i+1,j,l}$ determines the departure rate of finished streams on the receiving peers streamed from the active peers without failure of the peer that receives the stream, when the system is in state $(i+1,j,l)$. The explanation of the expression for this coefficient is the same as (8), with the difference that the streams are originating from the active peers:

$$\mu_{p,i+1,j,l} = (1-p_{\mathit{off}})(i+1)\mu \tag{10}$$

The coefficient $\mu_{p,i+2,j-1,l-1}$ determines the departure rate of finished streams on the receiving peers streamed from the active peers with failure of the peer that receives the stream, when the system is in state $(i+2,j-1,l-1)$. Its value is obtained by multiplying the video service rate μ with the size of the failure queue $i+2$ and the probability that each of the peers will decide to fail p_{off}:

$$\mu_{p,i+2,j-1,l-1} = p_{\mathit{off}}(i+2)\mu \tag{11}$$

In order to obtain the values of the remaining coefficients, we should examine more precisely the last member of the expression (1) which denotes the probability that neither arrival nor departure happens when the system is in state (i,j,l). The first multiplier $1 - P^{+}_{s,i,j,l}(\Delta t)$ denotes the probability that there will be no arrival for a request on the server. There are two situations when arrival can happen on the server: when an idle peer makes a request and when a failed peer makes a request immediately after the recovery. Therefore, the overall rate of arrivals will be the sum of the rates of each of these events. The final value can be obtained by adding or subtracting such values to the indexes of the functions defined in (4) and (5), so that they get the values i, j and l. The result of this operation is:

$$\begin{aligned}\lambda_{s,i,j,l} &= \lambda_{s,i,j-1+1,l} + \lambda_{s,i,j-1+1,l+1-1} = \\ &= (1 - P_{p2p}(i,l))(mn - i - j - l)\lambda + (1 - P_{p2p}(i,l))l\mu_{off}\end{aligned} \tag{12}$$

In a similar fashion, from the second multiplier $1 - P^{+}_{p,i,j,l}(\Delta t)$, the arrival rate of requests on the peers is calculated from (6) and (7):

$$\begin{aligned}\lambda_{p,i,j,l} &= \lambda_{p,i-1+1,j,l} + \lambda_{p,i-1+1,j,l+1-1} = \\ &= P_{p2p}(i,l)(mn - i - j - l)\lambda + P_{p2p}(i,l)l\mu_{off}\end{aligned} \tag{13}$$

The calculation of the coefficient $\mu_{s,i,j,l}$ comes from the third multiplier $1 - P^{-}_{s,i,j,l}(\Delta t)$ of (1),which determines the probability that there will be no end of a stream on a server within the interval Δt when the system is in state (i,j,l). The end of a stream on a server implies two possibilities: that the receiving peer will not fail and that the receiving peer will fail. The sum of the rates of these two events is the total departure rate of finished streams on the server. Like in the previous cases, it will be obtained from (8) and (9) by adjusting their indexes to values i, j and l:

$$\begin{aligned}\mu_{s,i,j,l} &= \mu_{s,i,j+1-1,l} + \mu_{s,i,j+1-1,l-1+1} = \\ &= p_{off}\, j\mu + (1 - p_{off})j\mu = j\mu\end{aligned} \tag{14}$$

In a similar way, from the fourth multiplier $1 - P^{-}_{p,i,j,l}(\Delta t)$ of (1), (10) and (11) the coefficient $\mu_{p,i,j,l}$ will be calculated as:

$$\begin{aligned}\mu_{p,i,j,l} &= \mu_{p,i+1-1,j,l} + \mu_{p,i+2-2,j-1+1,l-1+1} = \\ &= (1 - p_{off})i\mu + p_{off}\, i\mu = i\mu\end{aligned} \tag{15}$$

In the expressions for the coefficients, the only unknown variable is the probability $P_{p2p}(i,l)$ that a request for a strip will be redirected to be served by the peers when the system is in any of the set of states (i,j,l) such that $j \in [0, mn-i-l]$. The method for determining this value justifies the representation of all the peers in Fig. 2 with one representative peer that has the cumulative streaming and storage resources of all the peers. The probability $P_{p2p}(i,l)$ depends only on the number of busy channels on the peers i and the number of failed channels l, which define the number of available channels in the system. For the given set of states such that $j \in [0, mn-i-l]$, the number of available channels is $kn-i-l$. These available channels can be distributed on various ranges of peers. The minimum number of peers that can have that many available channels is $\lceil (kn-i-l)/k \rceil$, which is the case when almost each of these peers has k channels available. The maximum number of peers that can have that number of available channels is $\min(n, kn-i-l)$. In the last case, we use the minimum of the two values because if the number of available channels is lower than the number of peers, then the result would be distributing one available channel on each of the peers. In the opposite case, the maximum number of peers that can have that number of available channels will be n, where each of these peers will have at least one available channel. Having the minimum and maximum values, the range of peers that can have $kn-i-l$ available channels can be defined as:

$$R = \left[\left\lceil \frac{kn-i-l}{k} \right\rceil, \min(n, kn-i-l) \right] \tag{16}$$

Using the values of this range and the approach from [15], the probability that a request for a strip will be served by a peer when the system has i busy channels and l failed channels will be determined as:

$$P_{p2p}(i,l) = \sum_{v=1}^{c} P(v) \sum_{t \in R} w(t, kn-i-l) \left(1 - \left(1 - s\frac{P_x(v)}{m} \right)^t \right) \tag{17}$$

where $w(t,z)$ is the probability that z available channels will be distributed on t peers from the range R, $P_x(v)$ is the probability that a video with rank v will be stored in the peers and $P(v)$ is the probability that a video with rank v will be requested.

Substituting the coefficients (4)-(15) in (3) will give the final equation for the probability of each state in the system:

$$\begin{aligned}
0 = &(1 - P_{p2p}(i,l))(mn-i-j+1-l)\lambda p_{i,j-1,l} + \\
&+ (1 - P_{p2p}(i,l+1))(l+1)\mu_{\mathit{off}} p_{i,j-1,l+1} + \\
&+ P_{p2p}(i-1,l)(mn-i+1-j-l)\lambda p_{i-1,j,l} + \\
&+ P_{p2p}(i-1,l+1)(l+1)\mu_{\mathit{off}} p_{i-1,j,l+1} + \\
&+ (1-p_{\mathit{off}})(j+1)\mu p_{i,j+1,l} + p_{\mathit{off}}(j+1)\mu p_{i,j+1,l-1} + \\
&+ (1-p_{\mathit{off}})(i+1)\mu p_{i+1,j,l} + p_{\mathit{off}}(i+2)\mu p_{i+2,j-1,l-1} - \\
&- ((mn-i-j-l)\lambda + l\mu_{\mathit{off}} + (i+j)\mu) p_{i,j,l}
\end{aligned} \tag{18}$$

If we write this equation for every state of the system, we will obtain a system of linear equations which, if solved, will give the percentage of the traffic served by the peers and the server. In order to make a clearer picture of the system of linear equations, Fig. 4 shows a diagram of the flow of probabilities. In the diagram, the probability of every state of the system is represented by an elliptic node and the flow of probability from one state to another is represented by an arrow showing its direction. The intensity of each flow is the coefficient that multiplies the probability state in equation (18), but for clarity, it is intentionally omitted in the figure. The self-loop representing the probability flow from the central state to the central state is also omitted. Because of the large size and complexity of the diagram, in the figure, it is emphasized only the probability of the central state (i, j, l) and the probabilities of the states from which the central state can be reached. The diagram contains $kn + 1$ plains, where each plain contains the probabilities of the states with the same number of busy channels on the peers. In the partial diagram of a single plane, the probabilities of the states with the same number of failed channels are placed in the same row. The probabilities of the states with the same number of busy channels on the server are placed in the same diagonal. In the partial diagram that refers to the states that have i busy channels on the peers, the first row (the furthermost row) has $mn - i + 1$ states, and each following row has one state less than the previous. The last (closest to the viewer) row of each partial diagram has only one state.

The size of the system depends on the number of peers in the system n, the number of strips of each video m, and the number of channels of the peers k. From Fig. 4, it can be seen that the size can be obtained by summing the size of the partial diagrams of the separate plains and equals $(nk+1)\left(n^2+6m^2+2k^2-9mk)+n(18m-8k)+12\right)/12$.

3.4 Calculation of the system's performance

The final step for calculating the system's performance is to determine the state probabilities of the system in stationary state by solving the system of linear equations (18). In order to obtain unique solution, we modify the system by substituting an arbitrary equation with the condition that the sum of the state probabilities of the entire system equals 1.

The average number of busy channels on the peers can be obtained as the expected number of busy channels in the system. For that purpose, we have to calculate the probability that the system will have exactly i busy channels, which is the sum of the probabilities of all the states that lie in the plain in the diagram of probabilities shown in Fig. 4 corresponding to i busy channels:

$$P_i = \sum_{j=0}^{mn-i} \sum_{l=0}^{mn-i-j} p_{i,j,l} \tag{19}$$

If we use this value in the expression for the expected value of a random variable, we get the average number of busy channels $\overline{K}$ in the system as:

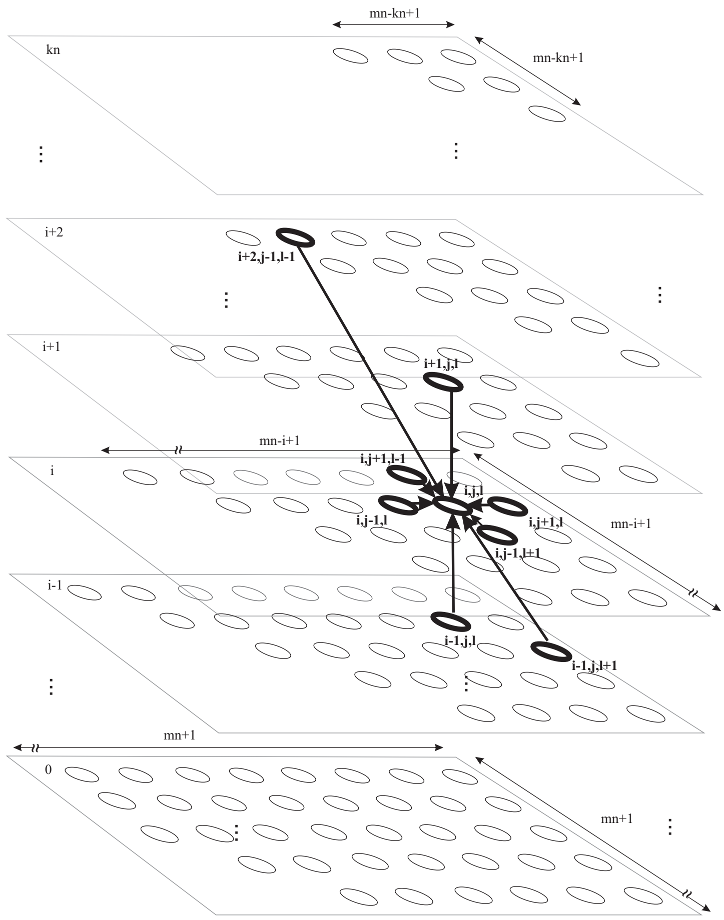


Fig. 4 Partial diagram of flow of probabilities of the system.

$$\overline{K} = E[K] = \sum_{i=1}^{kn} iP_i \tag{20}$$

We can calculate the average utilization of the streaming capacity if we divide the average number of busy channels by the total number of active peers. Therefore, we have to find the number of idle peers n_{idle} and the number of receiving peers n_{rcv}. For that purpose, we use the condition for equilibrium of the network of queues in Fig. 1 in stationary state. According to the condition for equilibrium [19], the arrival rate of customers that enter in one queue has to be equal to the departure rate of served customers that leave the queue. Since the considered network of queues is a closed system with finite population, in stationary state, it will achieve equilibrium, i.e., there will be no change of the number of customers (channels) in each service facility. The invariant number of customers in the service facilities means that there is also equilibrium in the flows of customers between adjacent service facilities, i.e., the number of customers entering a service facility in a given time interval must equal the number of customers that leave the facility in the same interval. Taking into consideration this fact, we derive the following equations:

$$\lambda n_{idle} = (1 - p_{\mathit{off}})\mu n_{rcv} \tag{21}$$

$$\mu_{\mathit{off}} n_{\mathit{off}} = p_{\mathit{off}} \mu n_{rcv} \tag{22}$$

$$\mu n_{rcv} = \lambda n_{idle} + \mu_{\mathit{off}} n_{\mathit{off}} \tag{23}$$

From the condition that in a closed network the number of peers is constant, it can be written:

$$n_{idle} + n_{rcv} + n_{\mathit{off}} = n \tag{24}$$

Using any two of the equations (21)-(23) and (24) will give the expression for the number of active peers in the system:

$$n_{act} = n_{idle} + n_{rcv} = \frac{1 - p_{\mathit{off}} + \rho}{(1 + \tau_{\mathit{off}}\mu)\,\rho + 1 - p_{\mathit{off}}} n \tag{25}$$

where $\rho = \lambda/\mu$ is the service rate of the system and $\tau_{\mathit{off}} = p_{\mathit{off}}/\mu_{\mathit{off}} = p_{\mathit{off}} T_{\mathit{off}}$ is the average off-time of the peers.

Eventually, we will calculate the average utilization of the streaming capacity of the active peers η as a portion of the overall number of channels on the active peers in the system that are actually occupied with streaming strips to the other peers:

$$\eta = \frac{\overline{K}}{n_{act} k} \tag{26}$$

The portion of traffic served by the peers θ can be calculated as a ratio between the traffic served by the active clients, which expressed as number of channels is the average number of busy channels in the system $\overline{K}$, and the overall traffic in the system. The overall traffic in the system, expressed as number of strips, is the traffic received by the receiving peers, and is obtained by multiplying n_{rcv} with the number of strips per video m. Hence:

$$\theta = \frac{\overline{K}}{n_{rcv} m} = \eta \frac{k}{m} \frac{1 - p_{\mathit{off}} + \rho}{\rho} \tag{27}$$

Another parameter that will be considered in the analysis is the quantity of streaming traffic served by the server, expressed as a percentage of the maximum system's throughput, i.e., the traffic that would be generated if all the peers were receiving a stream, without failures:

$$\Phi = (1 - \theta)\frac{n_{rcv}}{n} \tag{28}$$

4 Model verification and analysis

In order to verify the mathematical model, we developed a simulation model in the discrete event simulator OMNeT++ [33], which is an extensible, modular, component-based C++ simulation library and framework for building network simulators. It offers graphical runtime environment for defining different network topologies and simulation scenarios. We also use the INET library [32], which is an open-source communication networks simulation package for the OMNeT++ simulation environment. This framework contains models of most of the wired and wireless networking protocols. In the simulations, we define a network of servers and peers organized in local communities, which request videos from the servers. The peers are served either from the servers or from the other peers from the same local community. The entire communication is simulated by exchanging messages encapsulated with headers of the different networking protocols of the TCP/IP protocol stack.

We verify the correctness of the mathematical model by comparing the results obtained from the mathematical model with the results obtained from simulations of the real-time behavior of the system for peer-assisted VoD streaming with non-cooperative peers. In the simulations, the network consists of one local community of $n = 200$ peers and one representative server with sufficient resources to serve all the peers. This server has a role of an ES and index server. It hosts a library of $c = 1500$ Standard Definition (SD) quality contents with playback rate $r = 2$ Mbps, divided into $m = 10$ strips. The streaming capacity of the peers has a variable value and is expressed as a number of channels k for streaming a single strip of a video (200 kbps). The storage capacity s is expressed as a number of strips that can be stored in a STB and it has a value $s = 100$ strips, which is equivalent of 10 videos. The videos are previously distributed in the peers according to a distribution scheme that dedicates 30% of the storage space of the STBs for uniform distribution of the popular videos and the rest of the storage for uniform distribution of the not popular contents. According to the 20-80 rule [35], the first 20% of the most requested videos in the library are considered as popular, while the rest are not popular. Although we choose this distribution scheme for the initial simulation scenarios, it can be based on arbitrary probability function. The peers in the system are homogeneous, i.e., all the peers have the same storage and streaming capacity. This case is easy to be implemented because the operator can reserve the same amount of storage and streaming resources although the peers might be heterogeneous with different streaming and storage capacities.

The requests for VoD arrive with inter-arrival time of $w = 1/\lambda = 20$ min, which is the average time a client waits to request a video after previously watching a video. The average duration of the watching session is $d = 1/\mu = 90$ min. The probability that a client will decide to turn the STB off after the end of the streaming session p_{off} is a parameter that will be varied throughout the simulations, but in most of the cases it will be $p_{off} = 0.3$. The average time that will take the clients to recover after a failure is $T_{off} = 150$ min.

The popularity of the video contents obeys the Zipf-Mandelbrot (ZM) distribution [21]. This distribution is a modification of the Zipf distribution, commonly used for modelling popularity of web pages [2], obtained by the introduction of a shifting constant q:

$$P(v) = \frac{(v+q)^{-\alpha}}{\sum_{k=1}^{c}(k+q)^{-\alpha}} \tag{29}$$

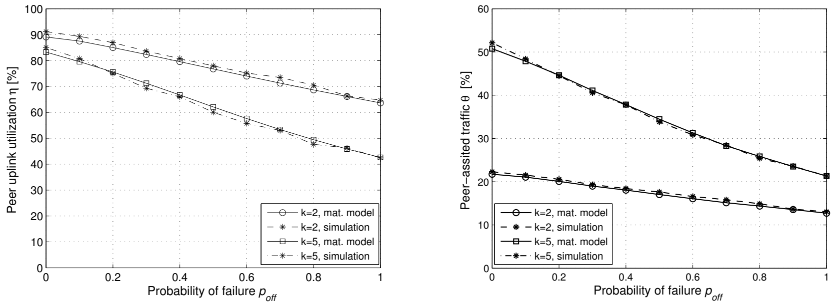
This expression gives the relative frequency of the v-th ranked video out of c videos in the library, where α is a skew factor which indicates the dispersion between the popular and not popular contents. The ZM distribution is used to better describe the specific behaviour of the clients when they request video contents: client that already watched one video will not repeat a request for the same video. This causes the steepness of the Zipf distribution curve to reduce for the popular videos, which is achieved by adding the shifting constant q. In the literature, the value of the skew factor obtained from trace data of systems with different sizes of the content library varies between 0.2 and 1 [34, 35, 8, 12, 3]. In our simulation, we use ZM distribution with skew factor $\alpha = 0.8$ and shifting constant $q = 10$ [1].

Since the mathematical model is defined for a stationary state of the system, the simulated time is long enough so that stable values of the parameters of interest are obtained.

In the first simulation scenario, we analyse the impact of the probability of failure p_{off} on the utilization of the streaming capacity η when the recovery time of the failed peers is $T_{\mathit{off}} = 150$ min for streaming capacity of $k = 2$ and 5 channels. The first conclusion from the comparison of the results obtained from the simulations and the mathematical model in Fig. 5(a), is the correctness of the mathematical model which is validated by the insignificant differences between the curves. The figure also shows that as the probability of failure p_{off} increases, the utilization of the streaming capacity falls almost linearly in both cases of streaming capacity. This behavior is quite expected because the increasing failure probability increases the number of failed peers in stationary state, which implies fewer peers that participate in the streaming. Therefore, although the peers have available streaming resources, they cannot be optimally used because the probability that a requested strip will be found somewhere on the peers reduces.

The effects of the worsened uplink utilization can be also seen in the percentage of the traffic streamed from the peers shown in Fig. 5(b). In this figure, the peer-assisted traffic falls from 50% to 20% of the overall traffic with the increasing of the failure probability p_{off} when the streaming capacity is $k = 5$. For streaming capacity of $k = 2$ strips, the uplink utilization falls more evenly, which is a result of the small participation of the peer-assisted traffic in the overall traffic. Fig. 5(b) also verifies the accuracy of the calculated results with the mathematical model compared to the results obtained from the simulations.

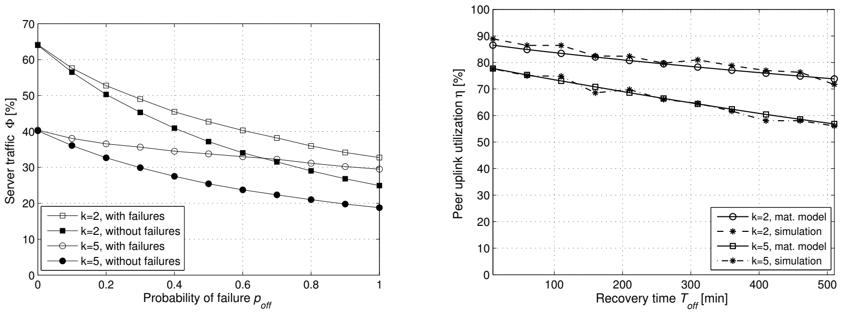
To show that the failures of the peers introduce more reduction of the peer-assisted traffic than the reduction that would cause only the reduced number



(a) (b)

Fig. 5 Dependence of (a) the uplink utilization η and (b) the peer-assisted traffic θ on the probability of failure of the peers p_{off} for streaming capacity $k = 2$ and 5 channels and recovery time $T_{off} = 150$ min.

of active peers, Fig. 6(a) shows a comparison of the overall server traffic in the case when the system has constant size n and there are failures of the peers and when there are no failures, but the size of the system n_{act} changes so that it has the same number of peers as there would be active peers when there are failures. For every value of the failure probability p_{off}, the size of the system $n = n_{act}$ is calculated according to (25). The purpose of this simulation scenario is to show the amount of additional traffic that is requested from the server for compensating the failure of the peers for uninterrupted streaming. Since there is different amount of overall traffic in the two considered cases, we choose the maximum streaming capacity of a system with $n = 200$ peers as a reference point for comparison. Therefore, we present the amount of the server traffic Φ as a percentage of the maximum traffic that could be generated in the system, i.e. the traffic that would be generated when all the n clients would simultaneously receive a stream. Fig. 6(a) shows that in the case when there are no failures, as expected, the reduction of the size of the system causes reduction of the requested traffic from the server Φ. However, when there are failures in the system with the same number of active clients as when there are no failures, there is smaller reduction of the server traffic. The reduction of the traffic comes from the reduced size due to the failures, but the more even slope of the curve comes from the additional traffic generated for compensating the strips that are interrupted with the failure of the peers. The additional traffic is actually the difference between the curve obtained for a system with failures and without failures. The value of the failure probability p_{off} has a great impact on the amount of this additional traffic. As it increases, there are more peers that fail and thus there are more streams that have to be compensated. Fig. 6(a) also shows that the additional traffic increases with the streaming capacity of the peers. There is more additional traffic when there are $k = 5$ channels, rather than when there are $k = 2$ channels, which is explained by the fact that the higher capacity implies higher number of outgoing streams from the active peers, and thus, the failure of one peer requires more compensating streams from the server. We intentionally omitted the comparison of the simulated values in the figure since the percentage of traffic served by the server Φ is calculated from the peer-assisted traffic θ, which was proven to be accurately computed in the Fig. 5(b).



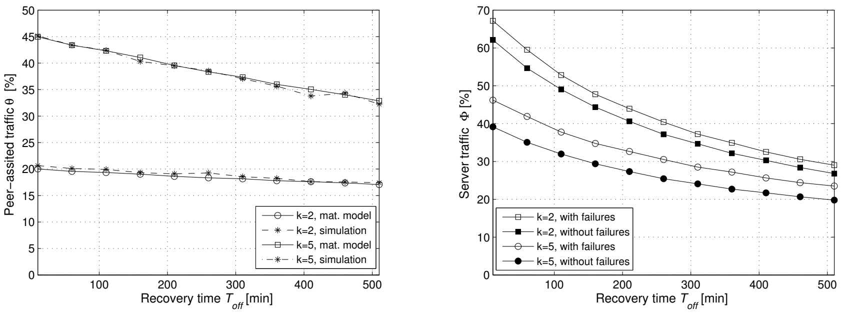
(a) (b)

Fig. 6 (a) Comparison of the requested traffic from the server Φ of a system with failures and a system without failures with the same number of active peers as the system with failures for recovery time $T_{off} = 150$ min and streaming capacity $k = 2$ and 5 channels and (b) Dependence of the peer uplink utilization η on the recovery time of the peers T_{off} for streaming capacity $k = 2$ and 5 channels and failure probability $p_{off} = 0.3$.

In the next simulation scenario we show the influence of the recovery time of the failed peers T_{off} on the system's performance. For that purpose, we set the probability of failure to value $p_{off} = 0.3$ and vary the recovery time T_{off} in the range from 10 to 510 min. The results presented in Fig. 6(b), show that the uplink utilization has a linear dependence on the recovery time T_{off}. The longer it takes a failed peer to recover, the less active peers are there in the system that can serve the receiving peers. Comparing the results obtained for the two different streaming capacities ($k = 2$ and 5 channels), it can be concluded that, although insignificantly, the influence of the duration of the failure state is more emphasized for higher streaming capacities.

The influence of the recovery time T_{off} on the percentage of the traffic served by the peers θ is shown in Fig. 7(a). According to the figure, a system with peers with $k = 5$ channels will be more affected by the longer recovery time of the failed peers than a system with $k = 2$ channels. Comparing the influence of the probability of failure p_{off} and the recovery time T_{off}, we can conclude that the former has more important role in the performance of the system. The failure probability p_{off} is more important because the traffic that has to be compensated is generated only in the moment of failure, which depends on p_{off}, while the recovery time T_{off} only determines how long there will be the reduced number of active clients in the system.

As in the previous analysis, these figures also show that the results from the mathematical model mostly overlap with the results obtained with simulation. For clarity, in the future analysis, this comparison will be intentionally omitted.

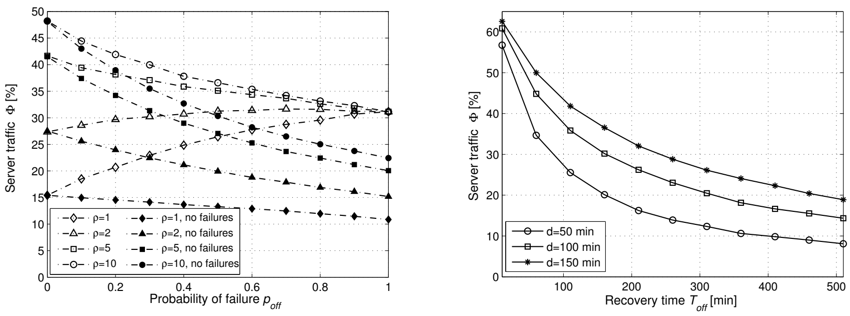


(a) (b)

Fig. 7 (a) Dependence of the server traffic Φ on the recovery time T_{off} for streaming capacity $k = 2$ and 5 channels and failure probability $p_{off} = 0.3$ (b) Comparison of the server traffic Φ of a system with failures and a system without failures with the same number of active peers as the system with failures for failure probability $p_{off} = 0.3$ and streaming capacity $k = 2$ and 5 channels.

The additional traffic requested from the server for compensating the streams of the failed peers is shown in Fig. 7(b) by comparing the curves of the server traffic in a system with failures with a constant size of the network and the server traffic in a system without failures and size of the network equal to the number of active peers of the system with failures. The maximum value of additional traffic is achieved for the smallest value of the recovery time $T_{off} = 10$ min, and then, it slightly reduces, which can be seen from the curves which are almost parallel with each other and have a tendency to join for very high values of the recovery time T_{off}. This figure justifies the earlier explanation that the additional traffic mainly depends on the failure probability p_{off} which determines the number of compensating streams from the failed peers, while the recovery time T_{off} only determines the general reduction of the traffic because of the reduced number of peers.

Fig. 8(a) shows the dependence of the server traffic on the failure probability p_{off} and the service rate $\rho = \lambda/\mu$. In order to obtain different values of the service rate ρ, we set the average duration of the streaming sessions to value $d = 100$ min and the inter-arrival time to values $w = 10$, 20, 50 and 100 min. Thus, the service rate obtains values $\rho = \lambda/\mu = d/w = 10$, 5, 2 and 1, respectively. Each value of the service rate is a separate simulation and is presented with a separate curve. The streaming capacity of the peers is $k = 5$ channels and the recovery time is $T_{off} = 150$ min. The server traffic is presented as a portion of the maximum throughput of the system. When there are no failures in the system ($p_{off} = 0$) the server is most loaded for the highest service rate and slowly reduces as the number of failures increases. The server load decreases because the recovery time T_{off} is longer than the duration of the streaming session, which means that in equilibrium there will be more failed peers than streaming peers. The server traffic for compensating the interrupted streams increases but not considerably compared to the other cases because the reduced number of streaming peers implies only a moderate number of strips that have to be compensated. This can be seen from the curve obtained for the case when there are no failures, but there is the same number of peers as the number of active peers. On the contrary, for low service rates ($\rho = 1$), the server traffic increases as the failure probability p_{off} increases. The reason for this is that, initially, there is a large number of idle peers. As the probability of failure increases, the number of receiving peers that fail increases and therefore the traffic that has to be compensated by the server increases. Thus, although the system has a tendency to reduce the server traffic with the reduction of the size of the community in the cases with no failures of the peers, for low service rates, the traffic is considerably increased for compensating the streams from the failed peers. This means that in the cases with low service rate, instead of alleviating the servers, the increased number of failing peers adds more load on the servers.



(a) (b)

Fig. 8 (a) Dependence of the server traffic Φ on the probability of failure p_{off} and the system's service rate ρ for $k = 5$ channels and recovery time $T_{off} = 150$ min, (b) Dependence of the server traffic Φ in the joining point on the recovery time T_{off} for duration of the session $d = 50$, 100 and 150 min.

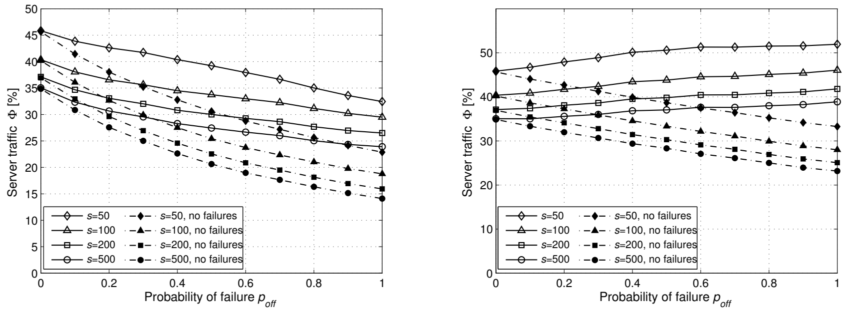
Another important observation from Fig. 8(a) is that, as the probability of failure p_{off} increases, all the curves converge to one point which is reached for $p_{off} = 1$. No matter what the service rate ρ is, for the case when all the peers fail with certainty, the server has to serve the same amount of traffic. This phenomenon can be justified with the fact that, when $p_{off} = 1$, the number of idle peers is $n_{idle} = 0$, i.e., a peer is either streaming or it is failed. The number of streaming and failed peers is independent on the inter-arrival rate, which can also be seen if $p_{off} = 1$ is substituted in (25). From the two different behaviors of the system, i.e., the tendency of the server traffic to decrease for some values of the service rate and to increase for others, it is natural to expect that for some value of the service rate ρ_x the server traffic will be constant, independent on the failure probability. The value of this service rate ρ_x is an important parameter for the system since for any rate higher than ρ_x the server traffic will decrease with the increment of the probability, and for the lower service rates the traffic will increase. Since for this specific case $\Phi(p_{off} = 0) = \Phi(p_{off} = 1)$, the value of ρ_x can be found from the curve of the dependence of the server traffic Φ on the service rate ρ obtained for $p_{off} = 0$ by locating the point which has value equal to $\Phi(p_{off} = 1)$. The value of the service rate in the point is the required critical value of the service rate ρ_x.

For a given service rate, the position of the joint point will depend on the value of the recovery time T_{off}. As the value of T_{off} decreases, the point moves upwards. This can be shown in Fig. 8(b), which gives the dependence of the server traffic in the joining point on the recovery time T_{off} for value of the duration of the video sessions $d = 50$, 100 and 150 min. If Fig. 8(b) is taken into consideration, the dependence in Fig. 8(a) for different values of T_{off} can be visualized by joining the initial points for $p_{off} = 0$ with a point obtained for a specific value of T_{off}.

Fig. 9(a) gives the dependence of the server traffic Φ on the peers' storage capacity s for streaming capacity $k = 5$ channels and recovery time $T_{off} = 150$ min. The server traffic has highest values for small storage capacities because, although the peers have available channels for streaming, they are not fully used as a result of the small number of strips they host. Therefore, all the requests for the strips that are not stored in the peers have to be streamed by the server. As the storage capacity increases, the availability of the contents increases, and the server traffic reduces. The figure shows that for storage capacities higher than $s = 100$ strips, the curves that describe the dependence of the server traffic on the failure probability have the same shape and are parallel to each other, which implies that the system reacts in nearly the same manner to the failures in all the cases of storage capacity, but with a different scale.

The comparison of the server traffic Φ with failures with the traffic generated in a network with the same size of active peers, but with no failures, shows that the additional traffic requested from the servers for compensating the interrupted streams is almost the same in all the cases for different storage capacities s and same probability of failure p_{off}. This can be seen from the equal distance between the curves obtained for a case with failures and those obtained for a case without failures for all considered values of the storage capacity s.

Fig. 9(b) shows the same dependence as in Fig. 9(a) with the difference that the recovery time is $T_{off} = 50$ min. The shorter recovery time T_{off} requires more additional traffic from the server, which causes the overall traffic requested from the server Φ to increase with the increment of the failure probability p_{off}, although there is a smaller number of receiving peers. In the same way as in the previous case, the additional traffic requested from the servers has almost the same value for all the cases of streaming capacities for a given value of the failure probability p_{off}.

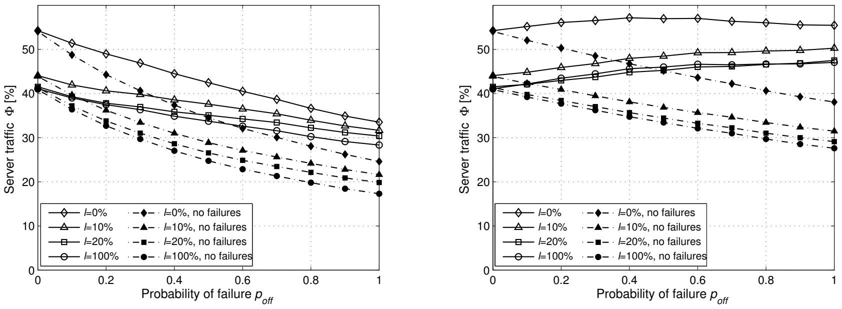


(a) (b)

Fig. 9 Dependence of server traffic Φ on the probability of failure p_{off} and the peers' storage capacity s for $k = 5$ for recovery time (a) $T_{off} = 150$ min and (b) $T_{off} = 50$ min.

The next simulation scenario shows the dependence of the server traffic Φ on the distribution of the contents in the peers obtained by changing the percentage of the storage capacity dedicated to the popular contents l for storage capacity $s = 100$ strips and recovery time $T_{off} = 150$ min. The distribution of both the popular and not popular contents is uniform, but with a different portion of the storage space dedicated for each of the two groups of contents. Fig. 10(a) shows that the server is most loaded when the peers store only the not popular contents, i.e., when the portion of the storage space dedicated to the popular contents is $l = 0\%$. Dedicating only $l = 10\%$ of the storage to the popular contents will significantly reduce the traffic originating from the server. Furthermore, the figure shows that dedicating more space to the popular contents will not remarkably contribute to the reduction of the server traffic because the curve obtained for $l = 20\%$ slightly differs from the one obtained for complete dedication of the storage to the popular contents ($l = 100\%$). As far as the additional traffic for compensating the interrupted strips is concerned, it can be concluded that the key factor is the failure probability p_{off}, while the distribution has only minor effect.

The similar dependence of the server traffic Φ on the portion of the storage space dedicated to the popular contents l can be observed in Fig. 10(b), where the recovery time is $T_{off} = 50$ min. The difference is that instead of the tendency to decrease, the server traffic increases as the probability of failure p_{off} increases.



(a) (b)

Fig. 10 Dependence of server traffic Φ on the probability of failure p_{off} and the portion of storage dedicated to popular contents l for streaming capacity $k = 5$ channels and recovery time (a) $T_{off} = 150$ min and (b) $T_{off} = 50$ min.

5 Conclusion

In this chapter, we proposed a system for peer-assisted VoD streaming in managed networks with non-cooperative peers and developed a stochastic mathematical model of the system, which apart from a large set of variables, includes the failure probability of the peers and their recovery time. The simulations showed that the mathematical model is an accurate representative of the proposed system since the results obtained with the calculations are identical to the simulation results. Throughout the chapter, the mathematical model was used as a tool to conduct various analyses on how the failures influence on the system performance. The results showed that the duration of the streaming sessions, the inter-arrival time, the failure probability and the recovery time have a crucial role in the overall performance. One of the findings with a significant importance is the fact that by increasing the failure probability, the traffic streamed by the servers converges to one point for any value of the inter-arrival rate of the request. However, in the cases when the service rate of the system is higher than some threshold service rate, although there is additional traffic to compensate the interrupted streams, the overall traffic of the server decreases with the increasing failure probability of the peers. In the opposite case, however, the failures of the peers have tendency to increase the overall streaming traffic from the servers. This fact could be a serious issue for the VoD provider because the servers are loaded with more traffic than it is planned for conditions with cooperative peers.

The storage capacity of the peers is also important for the amount of served traffic in the cases of failures, but this parameter only determines the initial value of the dependence curve, which follows the same shape as in the other cases of storage capacity. The same conclusion can be taken for the portion of storage space dedicated to the popular contents since it has a minor influence on the shape of the curve of the dependence on the server traffic for small values, but for the rest of the values it only determines the elevation of the curve.

The mathematical model proved to be a powerful and precise tool for estimating the behavior of the system for various parameters depending on the level of cooperativeness of the clients. The model could be used for planning the networks and predicting their performances for a large variety of parameters, which could save precious time and resources of the providers of VoD service in the privately managed networks.

The model opens a wide area of future work towards more realistic modelling of the managed networks, better utilization of the last-mile links and higher QoS. In that direction, we are encouraged to develop a heterogeneous model with peers that have different streaming and storage capacities. Our future work will tend to increase the reduction of the traffic in the core of the network by utilization of various incentive methods. Following the new trends of streaming technologies, the heterogeneity of the system could be extended to the very streaming process by implementation of Dynamic Adaptive Streaming over HTTP (DASH) [27]. This extension infers higher granularity of the streaming channels, storage of various copies of the strips for different streaming rates and introduction of probability that a client will switch from one streaming quality to other depending on the current state of the network. The price for these extensions will be the considerably increased complexity of the stochastic model and computation resources needed for obtaining the results.

References

1. Borst S, Gupta V, Walid A (2009) Self-organizing Algorithms for Cache Cooperation in Content Distribution Networks. Bell Labs Technical Journal, 14(3):113–125
2. Breslau L, Cao P, Fan L, Phillips G, Shenker S (1999) Web Caching and Zipf-like Distributions: Evidence and Implications. In: Proceedings of IEEE INFOCOM, vol 1, pp 126–134
3. Brosh E, Agastya C, Morales J (2009) Serving Niche Video-on-Demand Content in a Managed P2P Environment. Architecture pp 1–17
4. Cha M, Rodriguez P, Moon S, Crowcroft J (2008) On Next-generation Telco-managed P2P TV architectures. In: Proceedings of IPTPS, pp 1–6
5. Chen Y, Huang Y, Jana R (2007) When is P2P Technology Beneficial for IPTV Services. In: Proceedings of ACM NOSSDAV
6. Chen YF, Huang Y, Jana R, Jiang H, Rabinovich M, Rahe J, Wei B, Xiao Z (2009) Towards Capacity and Profit Optimization of Video-on-demand Services in a Peer-assisted IPTV Platform. Multimedia Systems 15(1):19–32
7. Chen YF, Jana R, Stern D, Wei B, Yang M, Sun H, Dyaberi J (2010) Zebroid: Using IPTV Data to Support STB-Assisted VoD Content Delivery. Multimedia Systems, 16(3):199–214

8. Chesire M, Wolman A, Voelker GM, Levy HM (2001) Measurement and Analysis of a Streaming-Media Workload. In: Proceedings of USITS, pp 1–12
9. Cisco Systems (2013) Cisco Visual Networking Index: Forecast and Methodology, 2012-2017
10. Ciullo D, Martina V, Garetto M, Leonardi E, Torrisi G (2014) Peer-Assisted VoD Systems: An Efficient Modeling Framework. Parallel and Distributed Systems, IEEE Transactions on 25(7):1852–1863
11. Do T, Hua K, Tantaoui M (2004) P2VoD: Providing Fault Tolerant Video-on-Demand Streaming in Peer-to-Peer Environment. In: Proceedings of IEEE ICC, pp 1467–1472
12. Dyaberi JM, Kannan K, Pai VS (2010) Storage Optimization for a Peer-to-Peer Video-on-Demand Network. In: Proceedings of ACM MMSys, pp 59–70
13. Fan B, Chiu DM, Lui J (2006) Stochastic Differential Equation Approach to Model BitTorrent-like P2P Systems. In: Proceedings of IEEE ICC, vol 2, pp 915–920
14. Muñoz Gea J, Nafaa A, Malgosa-Sanahuja J, Rohmer T (2012) Design and Analysis of a Peer-Assisted VoD Provisioning System for Managed Networks. Multimedia Tools and Applications, pp 1–36
15. Gramatikov S, Jaureguizar F, Cabrera J, García N (2013) Stochastic modelling of peer-assisted VoD streaming in managed networks. Comput Netw 57(9):2058–2074
16. Guo L, Chen S, Zhang X (2006) Design and Evaluation of a Scalable and Reliable P2P Assisted Proxy for On-Demand Streaming Media Delivery. IEEE Trans Knowl Data Eng 18(5):669–682
17. Jayasundara C, Nirmalathas A, Wong E, Chan CA (2011) Localized P2P VoD Delivery Scheme with Pre-Fetching for Broadband Access Networks. In: Proceedings of IEEE GLOBECOM, pp 1–5
18. Kerpez K, Luo Y, Effenberger FJ (2010) Bandwidth Reduction via Localized Peer-to-Peer (P2P) Video. Digital Multimedia Broadcasting, pp 1–10
19. Kleinrock L (1975) Queuing Systems, vol I: Theory. Wiley Interscience
20. Korosi A, Lukovszki C, Szekely B, Csaszar A (2009) High Quality P2P Video-on-Demand with Download Bandwidth Limitation. In: Proceedings of IWQoS, pp 1–9
21. Krogfoss B, Sofman L, Agrawal A (2008) Caching Architectures and Optimization Strategies for IPTV Networks. Bell Labs Technical Journal, 13(3):13–28
22. Kumar R, Liu Y, Ross K (2007) Stochastic Fluid Theory for P2P Streaming Systems. In: Proceedings of IEEE INFOCOM, pp 919–927
23. Li B, Yin H (2007) Peer-to-Peer Live Video Streaming on the Internet: Issues, Existing Approaches, and Challenges. IEEE Communications Magazine, 45(6):pp. 94–99
24. Liu Y, Guo Y, Liang C (2008) A Survey on Peer-to-Peer Video Streaming Systems. Peer-to-Peer Networking and Applications, 1(1):pp. 18–28

25. Lu Y, Zhang A, He H, Deng Z (2005) Stochastic fluid model for p2p content distribution networks. In: Proceedings of ISADS, pp 707–712
26. Lu Y, Mol JD, Kuipers F, Mieghem PV (2008) Analytical Model for Mesh-Based P2PVoD. In: Proceedings of IEEE ISM, IEEE Computer Society, pp 364–371
27. Mueller C, Lederer S, Timmerer C, Hellwagner H (2013) Dynamic adaptive streaming over http/2.0. In: Proceedings of IEEE ICME, pp 1–6
28. Qiu D, Srikant R (2004) Modeling and Performance Analysis of BitTorrent-like Peer-to-Peer Networks. In: Proceedings of SIGCOMM, pp 367–378
29. Ramachandran KK, Sikdar B (2005) An Analytic Framework for Modeling Peer to Peer Networks. In: Proceedings of IEEE INFOCOM, IEEE, vol 3, pp 2159–2169
30. Rimac I, Elwalid A, Borst S (2008) On Server Dimensioning for Hybrid P2P Content Distribution Networks. In: Proceedings of P2P, IEEE, pp 321–330
31. Tu YC, Sun J, Hefeeda M, Prabhakar S (2005) An Analytical Study of Peer-to-Peer Media Streaming Systems. ACM Trans Multimedia Comput Commun Appl 1(4):354–376
32. Varga A (2013) INET Framework. URL http://inet.omnetpp.org, available at: http://inet.omnetpp.org
33. Varga A, Hornig R (2008) An Overview of the OMNeT++ Simulation Environment. In: Proceedings of Simutools, pp 1–10
34. Yang M, Fei Z (2003) A Model for Replica Placement in Content Distribution Networks for Multimedia Applications. In: Proceedings of IEEE ICC, vol 1, pp 557–561
35. Yu H, Zheng D, Zhao BY, Zheng W (2006) Understanding User Behavior in Large-scale Video-on-Demand Systems. ACM SIGOPS Operating Systems Review 40(4):pp. 333–344
36. Zhou Y, Fu T, Chiu DM (2011) Statistical Modeling and Analysis of P2P Replication to Support VoD Service. In: Proceedings of IEEE INFOCOM, pp 945–953
37. Zhu P, Yoshiuchi H, Yoshizawa S (2010) P2P-Based VOD Content Distribution Platform with Guaranteed Video Quality. In: Proceedings of IEEE CCNC, pp 1–5

Sasho Gramatikov received a Bachelor degree in Computer science, information technologies and automation (five-year engineering program) in 2005 and a Master degree in Computer Science and computer engineering degree (two-year MS program) in 2009, both from the University of Ss Cyril and Methodius, Skopje, Macedonia. In 2013 he received a PhD degree at the Universidad Politécnica de Madrid (UPM), Madrid, Spain. He is currently working as an Assistant Professor at the Faculty of Computer Science and Engineering in Skopje, Macedonia. His research interests are distribution and streaming of video contents in networks.

Fernando Jaureguizar received the Telecommunication Engineering degree (six-years engineering program) and the Ph.D. in Telecommunication, both from the Universidad Politécnica de Madrid (UPM), in 1987 and 1994, respectively. Since 1987 he is a member of the Image Processing Group of the UPM. In addition, since 1991 he is a member of the faculty of the E.T.S. Ingenieros de Telecomunicación at UPM, and since 1995 he is an Associate Professor of Signal Theory and Communications at the Department of Signals, Systems, and Communications. His professional interests include digital image processing, video coding, 3DTV, computer vision, and design and development of multimedia communications systems. He has been actively involved in European projects (Eureka, ACTS and IST) and national projects in Spain.

View publication stats