Exploiting user preferences to reduce bandwidth requirements for VoD services with client caching

Felipe Alex Pinto | Evelio Martín García Fernández | Carlos Marcelo Pedroso

Department of Electrical Engineering, Federal University of Parana, Curitiba, Brazil

Correspondence
Carlos Marcelo Pedroso, Department of Electrical Engineering, Federal University of Parana, Curitiba, Brazil.
Email: pedroso@eletrica.ufpr.br

Abstract
Internet traffic is already dominated by video streaming applications. The quality of transmitted videos and the number of content available online is expected to increase in the upcoming years, which will further pressure the available network infrastructures. In this scenario, Video-on-Demand services require that distribution mechanisms improve the efficiency of video transmission, which impacts the network performance and system scalability. The efficiency of distribution is strongly related to the operating costs of the providers of such services. The most efficient methods available for Video-on-Demand distribution use strategies that combine multicast transmission and the storage capacity of the user’s equipment. This paper presents a new method for Video-on-Demand distribution that explores the client storage capabilities by modelling the users’ preferences with a Hidden Markov Model. The efficiency of the proposed method is demonstrated through computational simulations of different scenarios, including a sample of real users’ activity. Our results indicate that the proposed scheme substantially outperforms the current Video-on-Demand distribution mechanisms in terms of network bandwidth consumption, significantly reducing operation costs by improving the system scalability.

1 | INTRODUCTION

Broadband access technologies have evolved over the years. Today, the availability of high-speed networks changed the way of creating, consuming, and sharing digital media. Video-on-Demand (VoD) has become increasingly popular, making video streaming the most prominent application on the Internet. It is estimated that in 2021 video traffic will correspond to 82% of total Internet traffic [1]. The VoD service improves users interactivity, allowing the users to watch the videos they want at any time, instead of having access only to broadcast transmissions.

The bandwidth efficiency and scalability are important issues in VoD distribution. Several methods have been proposed to address these problems, such as batching [2], pyramid broadcasting [3], skycraper broadcasting [4], harmonic broadcasting [5], delayed buffering broadcast [6], hybrid broadcasting [7], multiplexed harmonic broadcasting [8], patching [9], among others [10]. Readers can find in [11] a survey of three classes of techniques for improving VoD distribution. The paper addresses methods for network load reduction, network interruption mitigation, and network load distribution. Several methods for VoD distribution, including cached delivery, peer-to-peer delivery and segment caching are also described.

Jayasundara et al. [12] proposed a method to improve scalability and efficiency in VoD distribution, called Pre-population Assisted Batching with Multicast Patching (PAB-MP). Results show that the use of PAB-MB significantly reduces the aggregated bandwidth consumption of the VoD server if compared with competing methods available at that time, mainly in high load conditions. Recently, Feng et al. [13] developed a new method for VoD delivery called Client Cache Enabled Multicast Patching (CCE-MP) which exploits the storage capacity of the users’ device. The authors show that CCE-MP overcomes the performance of other methods, including PAB-MP, in terms of resources required on the server and the network.

The methods cited above do not consider user preferences for delivering VoD services. For this reason, we present a new approach based on Hidden Markov Models (HMM) to predict which category of video is most likely to be accessed by the user. The result of this prediction is then used to exploit the allocation of video segments in the user device cache that, along with a multicast transmission strategy, reduce the bandwidth...
The use of unicast transmission for VoD traffic distribution, where a dedicated video stream is sent from the server to each client, is highly inefficient. The first attempts to increase efficiency were based on broadcast, that is, the video is sent to all clients from time to time. Then, new methods have proposed the use of prior knowledge of the video popularity and network multicast capabilities. More recently, methods that explore the storage capacity in the users’ device and the network multicast capabilities simultaneously have been proposed. In this section, we will discuss some of the main available methods for VoD distribution.

### 2.1 Batching

Suppose that the first request for a given video $v_n$ is at time $t_r$. The algorithm waits for a time $T$, called batching window, to start streaming. During this time $[t_r, t_r + T]$, the video server receives requests for $v_n$ before starting the transmission at time $[t_r + T]$. Then, the server starts the streaming using multicast to deliver $v_n$ to multiple users [14]. The requests for $v_n$ received after the end of the batching window will be answered with a new multicast stream.

This algorithm has a good performance for popular videos, for which many requests will be met with only one stream. However, this method is not efficient for non-popular videos, as it tends to become similar to the unicast strategy. Besides that, in this approach, the user may need to wait up to $T$ seconds before starting to watch the video, which may motivate the user to leave the service platform in advance.

### 2.2 Patching

In this algorithm, the storage capacity of user equipment is exploited, so that it can receive the main multicast flow plus a stream called patching. Suppose a user request for video $v_n$ at time $t_r$. If there is an active main multicast stream of $v_n$ at time $t_r$, the server will start sending the missing part of the video through a patching stream [15], otherwise the server immediately initiates a main multicast stream for $v_n$.

The main problem occurs when a request arrives late in relation to the start of the main multicast stream, since the resulting patching stream would be too long. In order to handle this problem, the server should accept patchings as long as the request for video $v_n$ arrives before a time limit from the beginning of the main stream multicast [16]. If it arrives after the time limit, a new main multicast stream should serve this request. Thus, this method ensures zero delay for the user, that is, any time users make a request, they can start watching the video.

### 2.3 Pre-population assisted batching with multicast patching

This algorithm belongs to the class of methods capable of exploring the storage capacity of a user’s device to implement a cache with the initial video segments (IVS) [12]. The user request for video $v_n$ at time $t_r$ is processed as follows. If there...
is an active multicast for \( v_n \), the client joins the group and starts to receive the requested video. In this case, the user starts watching the video using information previously stored locally in the IVS, otherwise client starts a new multicast for \( v_n \).

The size of the IVS defines the batching window. For requests that arrive after this limit, the server starts a multicast patching flow in order to serve a greater number of streams. Requests that arrive after the patching window will be met with a new multicast stream. The IVSs are sent to the user’s device in times of low network and server usage. The use of the Dynamic programming-based Pre-population Lengths Optimisation (DPLO) algorithm was suggested by Jayasundara [12] to determine the size of each IVS to be stored on the user’s device. Additionally, the authors suggest that users’ devices share the locally available IVSs in a peer-to-peer configuration to increase the possibility of finding the requested video. The authors also show that the PAB-MB is more efficient than the previous patching and batching strategies.

2.4 Client cache enabled multicast patching

This approach also exploits the storage capacity of the users’ devices and was proposed by Feng et al. [13, 17]. Considering a request of video \( v_n \) at time \( t_s \), the user’s device immediately starts storing all active segments of multicast streams of \( v_n \). As in the PAB-MB, the user starts watching the video using information previously stored locally in the IVS. The beginning of the transmission of missing segments of \( v_n \) will occur as late as possible in relation to \( t_s \), so that the largest number of users can use the multicast flow of \( v_n \).

Video library consists of \( N \) videos characterised by a tuple \( \{ \text{length}, \text{bitrate}, \text{popularity} \} \). It is assumed that all videos are of equal length and bit rate, denoted by \( L \) and \( r \), respectively. The popularity distribution vector \( p = [p_0, p_1, \ldots, p_{N-1}] \) is assumed to be known \textit{a priori}. The popularity rank of a video \( v_i \) is given by \( i \), where \( p_0 \geq p_1 \geq \ldots \geq p_{N-1} \). Using the Water–Filling algorithm, the IVS for each video can be evaluated as follows:

\[
l_i = \min \left( \left( \beta - \frac{r}{\lambda_i} \right)^+, L \right), \quad i = 0, 1, \ldots, N - 1,
\]

where \( x^+ = \max(x, 0) \) and \( \beta = \frac{r}{\sum_{i=0}^{K} \lambda_i} \), where \( f_B \) is the spectral efficiency and \( \mu \) is the Lagrange multiplier. \( l_i \) and \( \beta \) can be effectively solved by the bisection method under the storage constraint [17].

Simulated results show that CCE-MP outperforms PAB-MP and other competing methods in terms of bandwidth consumption.

3 PRE-POPULATION ASSISTED BATCHING WITH PREDICTION OF USER BEHAVIOUR

The content accessed by users of VoD systems presents a predictable behaviour [18, 19], which is already explored by the recommender systems used by providers [20]. We propose to explore this predictability to perform a clever IVS allocation, improving performance as a whole. To this end, we use an HMM-based model to predict the genre of the next video that will be requested by the user, allowing the server to allocate IVS more efficiently, reducing the bandwidth usage in the network core and also resources in the servers.

3.1 Hidden Markov models

In a Markov chain, each state corresponds to an observable event [21]. The use of Markov chains to synchronise real-time streaming was proposed by [22]. In HMM, the current state of the system is not directly observable. These models have wide application, such as speech recognition, DNA sequence, facial expression recognition, and others. Figure 1 exemplifies a three-state HMM, where \( E_k \), with \( k = [0, 1, 2] \), represents the hidden states, \( p_{kj} \), \( j = [0, 1, 2] \), is the value for the transition probabilities from state \( E_k \) to \( E_j \), and vector \( w_{k,*} = [w_{k0}, w_{k1}, \ldots, w_{k(M-1)}] \) represents the probabilities of observation, also called emission probabilities, of observable event \( m \in \{0, 1, \ldots, M - 1\} \) in a given state \( E_k \). The number of hidden states is given by \( K \), and the number of different video categories is given by \( M \). The initial probability of each state is given by vector \( U \).

3.2 User behaviour forecast

In this subsection, we describe how to use of an HMM to predict the genre of content that will be accessed by the user. Each video \( v_n \), \( n \in \{0, 1, 2, \ldots, N - 1\} \) in the library is denoted by a tuple \( \{ \text{length}, \text{bitrate}, \text{popularity}, \text{category} \} \). As in CCE-MP and PAB-MB, we also assume that all videos are of equal length and bit rate, given, respectively, by \( L \) and \( r \). Additionally, we consider that each video has a unique associated genre from a finite set \( S = \{s_0, s_1, \ldots, s_{M-1}\} \) of categories and that the video server has the ability to track the history of videos accessed by the user. Thus, associated to each state \( E_k \) of a model with \( K \) states, there is an emission probability vector \( w_{k,*} = [w_{k0}, w_{k1}, \ldots, w_{k(M-1)}] \),
where $\sum_{m=0}^{M-1} w_{m} = 1$, which stores the probabilities of occurrence of each video category in that state.

Let $P$ be the $K \times K$ state transition probability matrix and $W$ be the corresponding $K \times M$ emission probability matrix of the HMM. $P$ and $W$ can be obtained from the historical data of video requests using the Baum–Welch algorithm [23]. The resulting probability transition matrix is then used to evaluate the stationary distribution of the Markov Chain given by vector $\pi = [\pi_0, \pi_1, \ldots, \pi_{K-1}]$.

The most probable state for a given user can be evaluated from the observation of the sequence of categories requested using the Viterbi or the forward algorithm [23]. The video server can perform the training periodically to reflect changes in user behaviour.

### 3.3 IVS allocation

From the trained HMM, the Viterbi algorithm can indicate the most likely current state of the user given a sequence of video categories requested by the user. The corresponding vector $w_{k,*}$ can be used to define the cache utilisation in the user equipment. The partitioning of the storage capacity of the user’s device will be proportional to the emission probability for each video category, depending on the most likely state of the user.

Let $C$ be the memory space that can be used in the user’s equipment to store IVSs, so for a given $w_{k,*}$ the capacity $C$ will be allocated among the categories according to the emission probability of HMM for the current user state. That is, for a category $s_m$, the allocated space for the IVSs of this category is given by $c_{km} = C \cdot w_{km}$. Figure 2 illustrates the process for a 3-state HMM with $M = 5$. The server keeps a historical record on all videos requested by users. The Baum–Welch and Viterbi algorithms run in the server, which will decide on the occupation of the available cache area in users’ devices. We use the Water–Filling algorithm to set the length of the IVS for videos of a given category according to Equation (1). The video’s popularity is evaluated by considering the video category and the current state of the user in HMM.

### 4 PERFORMANCE EVALUATION

User behaviour plays a key role in the allocation of IVS. To study the effects of user behaviour on the performance of PAB-UBP, we propose three scenarios:

- **Polarised**: Users have a high probability of continue watching the same category of content, as in the phenomenon of binge watching [24]. This behaviour is described in an HMM model with a transition probability matrix $P$ close to the identity matrix, that is, since the user is in state $E_k$ he/she tends to remain in the same state and the probability vector $w_{k,*}$ is highly polarised, that is, the user has preference for only one category.

- **Random**: Users have no preference for a specific video category and request them randomly. The transition probability matrix follows approximately an uniform distribution. similarly, the vector $w_{k,*}$ has equal probability of issuing between the categories.

- **Sampled Data**: We collected the access history of 50 VoD users totalling 3597 videos watched and more than 40,000 requests in an average range of 2.5 years.

The bandwidth consumption was evaluated using computational simulations, comparing Patching [16], CCE-MP [17], and PAB-UBP. The system performance was studied using different request rates, different trends in video popularity, and different number of hidden states of the Markov chain. The requests that arrive at the video server can be characterised by a Poisson process with request rate $\lambda$ [25]. On the other hand, the popularity of videos is commonly characterised by the Zipf distribution [26], whereby the popularity of a video $i$, ranked from the most popular to the least, stored in a library containing $N$ videos is given by $p_i = (1/\alpha^i)/(\sum_{j=1}^{N} 1/\alpha^j)$, where $\alpha$ is the exponent characterising the distribution.

The simulation considers that the requested video will be entirely watched by the user and that all videos have the same length and transmission rate. These assumptions are also made by the authors of Patching, PAB-MP and CCE-MP, which simplifies the analysis without the loss of generality. In addition, it was considered that the popularity of videos of the same category follows the Zipf distribution with parameter $\alpha$ and the initial state of the HMM was chosen at random. Table 2 shows all the parameters used in the simulations.

In a first experiment, the consumption of bandwidth was considered as a function of the request rate $\lambda$ received by the server. In this test, the parameter $\alpha$ of the Zipf distribution was kept constant at 0.8, which is a typical value reported in the literature.

Figure 3(a) shows the performance of Patching, CCE-MP, and PAB-UBP in terms of average bandwidth consumption. For PAB-UBP, cases of Random and Polarised user
behaviour were considered. It is observed there is no significant difference between the methods for low requests rates. However, as the request rate increases, PAB-UBP outperforms the others regarding Polarised behaviour, since the proposed method present a better hit rate of IVS stored in users’ devices. For Random behaviour, it is possible to notice that the CCE-MP is more efficient. However, the behaviour of real users is much closer to the polarised profile, as illustrated in the next section, which tends to favour PAB-UBP.

In a second experiment, the bandwidth consumption was evaluated as a function of the shape parameter of the Zipf distribution, $\alpha$. Figure 3(b) shows that, when considering the random scenario, CCE-MP only performs similarly to the proposed method for lower values of $\alpha$ (special case where the video popularity follows an uniform distribution). Considering the polarised scenario, there is a saving of 21% in bandwidth consumption.

Some studies show that Zipf’s $\alpha$ parameter varies between 0.5 and 1.0 [12, 27]. Within this range, PAB-UBP has the best performance among all competing methods. For $\alpha > 1.2$, CCE-MP becomes more efficient than PAB-UBP in the Polarised scenario, because for $\alpha > 1.2$ only a small number of videos are requested by users and few videos represent almost 100% of the videos effectively transmitted.

4.1 | Sample data

Sample data from 50 VoD users were used to discover the parameters of the HMM, which was later used in simulations. Each video was manually classified into one of the following 16 categories: Drama, Comedy, Action, Adventure, Animation, Romantic, Sci-Fi, Romance Comedy, Comedy Drama, Action Comedy, Suspense, Thriller, Fantasy, Serie, Documentary and Other. From these data, the HMM was trained using the Baum–Welch algorithm to obtain the transition probability matrix and the probabilities of emission [28]. In this paper, an HMM with 3, 4, 5, and 10 states was considered. The resulting $P$ and $W$ matrices for the 3-state HMM are given, respectively, by

$$P = \begin{bmatrix} 0.98 & 0.00 & 0.02 \\ 0.26 & 0.27 & 0.47 \\ 0.00 & 0.91 & 0.09 \end{bmatrix}, \quad U = \begin{bmatrix} 1/3 & 1/3 & 1/3 \end{bmatrix},$$

$$W = \begin{bmatrix} 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.11 & 0.08 & 0.11 & 0.03 & 0.06 & 0.04 & 0.04 & 0.06 & 0.04 & 0.02 & 0.05 & 0.04 & 0.03 & 0.15 & 0.07 & 0.06 \\ 0.14 & 0.08 & 0.10 & 0.04 & 0.07 & 0.05 & 0.04 & 0.04 & 0.04 & 0.02 & 0.06 & 0.05 & 0.02 & 0.14 & 0.06 & 0.05 \end{bmatrix}.$$  

A reasonable polarisation is observed in the user behaviour, which favors the efficiency of the proposed method. For instance, it can be seen from $P$ that, if the process is in state $E_i$, there is 98% of probability to remain in the same state in the next request. In this state, the emission probability vector $w_{i,*}$ (first row of $W$) indicates that the user is 97% likely to request only one video category. This behaviour is closer to the polarised profile, which favors PAB-UBP, as it was already shown in previous section.

We assume that, in each category, the videos’ popularity is also modelled by a Zipf distribution. For simulation, it was considered that each category had 320 videos, with a total of 5120 videos in the simulated video library. Figure 4(a) shows the bandwidth consumption as a function of the request rate. Considering a request rate of $\lambda = 8$ req/min, PAB-UBP saved

![Figure 3](image-url) Average bandwidth consumption of the polarised and random scenarios: (a) as a function of the request rate and (b) as a function of $\alpha$. 

**TABLE 2** Simulation parameters

| Parameter | Value |
|-----------|-------|
| $N$       | 100 videos |
| $M$       | 5 categories |
| $K$       | 3 states |
| $r$       | 2 Mbps |
| $L$       | 60 min |
| $C$       | 120 min |
| $f_B$     | 1 bps/Hz |
| $\lambda$ | 10 min$^{-1}$ |
59.53% of the network bandwidth when compared to CCE-MP. This margin tends to increase as the request rate increases due to the better hit rate of IVS.

Figure 4(b) shows the bandwidth consumption as a function of \( \alpha \) for \( \lambda = 10 \text{ req/min} \). At each step, the same value of \( \alpha \) was used for all 16 different video categories to generate the user requests. It is possible to observe that, compared to CCE-MP, the performance of PAB-UBP was considerably better. It can also be observed that the variation of \( \alpha \) leads to a large variation in CCE-MP performance.

To assess the impact of the number of HMM states on the final performance of the system, the training of the HMM was performed considering 3, 4, 5, and 10 states using the sampled data. Figure 5(a) shows the effects of varying the arrival rate \( \lambda \) from 1 to 10 requests per minute. Figure 5(b) presents the effects of varying \( \alpha \) from 0 to 1.1 on the average bandwidth consumption. It can be observed that using a 4-state HMM leads to better performance with lower bandwidth consumption. However, this result is not generalisable because it depends on the dataset under study. The limited number of 50 users in this study allowed a few sets of classes, which should increase if larger populations of users are considered. Results also show that the higher the \( \alpha \), the better the performance of PAB-UBP, as previously illustrated.

Figure 6 shows the bandwidth consumption as a function of the storage capacity of the user’s device. The storage capacity is denoted by \( C/(NL) \), which represents the fraction of all videos available in the library that could be stored locally. It is possible to see that PAB-UBP outperforms CCE-MP for the polarised behaviour. If a user chooses video categories randomly,
CONCLUSIONS

We proposed an HMM-based method to perform the allocation of initial video segments using users’ preferences. Through computational simulations, we showed that the proposed method outperforms the reference model (CCE-MP) in terms of bandwidth consumption, and that it can be used together with multicast transmission and storage of user’s device in order to save bandwidth in the network and improve system scalability.

It is important to note that the gain margin changes as the user’s behaviour changes, which has been demonstrated through the Polarised and Randomised scenarios. We also show that in a real system, the user’s behaviour is expected to be somewhere between these behaviours.

ACKNOWLEDGEMENT

The authors would like to thank the Academic Publishing Advisory Center (Centro de Assessoria de Publicação Acadêmica, CAPA (http://www.capa.ufpr.br)) of the Federal University of Paraná (UFPR) for assistance with English language editing.

ORCID

Carlos Marcelo Pedroso  https://orcid.org/0000-0002-3747-7238

REFERENCES

1. Cisco: Cisco visual networking index: forecast and methodology, 2016–2021. White Paper - Cisco Public, Technical Report (2017)
2. Aggarwal, C.C., Wolf, J.L., Yu, PS: On optimal batching policies for video-on-demand storage servers. Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems, pp. 253–258.IEEE, Piscataway, NJ (1996)
3. Viswanathan, S., Imielinski, T.: Metropolitan area video-on-demand service using pyramid broadcasting. Multimedia Syst. 4(4), 197–208 (1996)
4. Hua, K.A., Sheu, S.: Skyscraper broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems. SIGCOMM Comput. Commun. Rev. 27(4), 89–100 (1997)
5. Juhn, L.-S., Tseng, L.-M.: Harmonic broadcasting for video-on-demand service. IEEE Trans. Broadcast. 43(3), 268–271 (1997)
6. Chung, Y.: Alternate video broadcasting scheme for popular videos. IET Commun. 3(8), 1321–1332 (2009)
7. Chen, Y., Lin, C., Huang, C.: Hybrid broadcasting scheme with low waiting time and buffer requirement for video-on-demand services. IET Commun. 6(17), 2949–2956 (2012)
8. Liu, Y., Yu, S., Wang, X.: Multiplexed harmonic broadcasting scheme for efficient video-on-demand services. IET Commun. 1(1), 15–18 (2007)
9. Hua, K.A., Cai, Y., Sheu, S.: Patching: a multicast technique for true video-on-demand services. MULTIMEDIA ’98: Proceedings of the Sixth ACM International Conference on Multimedia, pp. 191–200.ACM, New York, NY (1998)
10. Choi, J., Reaz, A.S., Mukherjee, B.: A survey of user behavior in VoD service and bandwidth-saving multicast streaming schemes. IEEE Commun. Surv. Tutorials 14(1), 156–169 (2012)
11. Ma, K.J., Bartos, R., Bhatia, S.: A survey of schemes for internet-based video delivery. J. Netw. Comput. Appl. 34(5), 1572–1586 (2011)
12. Jayasundara, C., et al.: Improving scalability of VoD systems by optimal exploitation of storage and multicast. IEEE Trans. Circuits Syst. Video Technol. 24(3), 489–503 (2014)
13. Feng, H., Chen, Z., Liu, H.: Design and optimization of VoD schemes with client caching in wireless multicast networks. IEEE Trans. Veh. Technol. 67(1), 765–780 (2018)
14. Dan, A., Sitaram, D., Shahabuddin, P.: Scheduling policies for an on-demand video server with batching. MULTIMEDIA ’94: Proceedings of the Second ACM International Conference on Multimedia, pp. 15–23.ACM, New York, NY (1994)
15. Hua, K.A., Cai, Y., Sheu, S.: Patching: A multicast technique for true video-on-demand services. MULTIMEDIA ’98: Proceedings of the Sixth ACM International Conference on Multimedia, pp. 191–200.ACM, New York, NY (1998)
16. Gao, L., Towsley, D.: Threshold-based multicast for continuous media delivery. IEEE Trans. Multimedia 3(4), 405–414 (2001)
17. Feng, H., Chen, Z., Liu, H.: Minimizing bandwidth requirements for VoD services with client caching. 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–7.IEEE, Piscataway, NJ (2016)
18. Ren, P., Wang, X.: User preference and behavior pattern in push VoD systems. 2014 IEEE 5th International Conference on Software Engineering and Service Science, pp. 425–429.IEEE, Piscataway, NJ (2014)
19. Xiao, Y., Lai, J., Liu, Y.: A user participation behavior prediction model of social hotspots based on influence and Markov random field. China Commun. 14(5), 145–159 (2017)
20. Gomez.Uribe, C.A., Hunt, N.: The Netflix recommender system: Algorithms, business value, and innovation. ACM Trans. Manage. Inf. Syst. 6(4), 1–13 (2015)
21. Grinstead, C.M., Snell, J.L.: Introduction to Probability, 2nd edn. American Mathematical Society, Providence, RI (1998)
22. Wu, J., et al.: A novel transmission scheme to inter destination video synchronisation. IET Commun. 9(5), 603–612 (2015)
23. Rabiner, L., Juang, B.: An introduction to hidden Markov models. IEEE ASSP Mag. 3(1), 4–16 (1986)
24. Claes, M., et al.: Cooperative announcement-based caching for Video-on-Demand streaming. IEEE Trans. Netw. Serv. Manage. 13(2), 308–321 (2016)
25. Azgin, A., AlRegib, G., Atlunbasak, Y.: Cooperative delivery techniques to support Video-on-Demand service in IPTV networks. IEEE Trans. Multimedia 15(8), 2149–2161 (2013)
26. Xia, B., Yang, C., Cao, T.: Modeling and analysis for cache-enabled networks with dynamic traffic. IEEE Commun. Lett. 20(12), 2506–2509 (2016)

27. Yu, H., et al.: Understanding user behavior in large-scale video-on-demand systems. SIGOPS Oper. Syst. Rev. 40(4), 333–344 (2006)

28. R Core Team: A language and environment for statistical computing. R Core Team, Vienna, Austria, 2018. Access April 30, 2020, https://www.R-project.org/

How to cite this article: Pinto FA, Fernández EMG, Pedroso CM. Exploiting user preferences to reduce bandwidth requirements for VoD services with client caching. IET Commun. 2021;15:763–770. https://doi.org/10.1049/cmu2.12118