Caching video contents in IPTV systems with hierarchical architecture

Lydia Chen[1], Michela Meo[2] and Alessandra Scicchitano[1]
1. IBM Zurich Research Lab – email: {yic,als}@zurich.ibm.com
2. Politecnico di Torino, Italy – email: michela.meo@polito.it

***Abstract*— In this paper we consider IPTV systems with an hierarchical architecture. The lowest elements of the architecture are set-top boxes (STBs) at the user homes; a STB is connected to a central office (CO) which is in charge of delivering the video content to the end user. Since COs have limited storage capabilities, they may need to retrieve a particular video content that is requested by a user but temporary not stored in the local memory. Thus, COs exchange video contents with similar COs in a peer-to-peer fashion. At a higher hierarchical level, video source offices (VSOs) offer the video contents that cannot be retrieved at the CO level. Video content caching strategies at the COs and VSOs influence the system performance in terms of traffic exchanged between the network nodes. In this paper we propose two simple strategies that aim at reducing both the intra- and inter-level traffic. The strategies are studied by means of an analytical model that is validated against simulation results. Our results show that the hierarchical architecture allow good system performance even with limited overall storage capacity. The proposed strategies, in particular, can be helpful in improving system performance.**

I. Introduction

The past few years have witnessed the deployment of fully operative multi-service networks. The distinction between operators that provide telephony or data services, web browsing or real-time delay-sensitive services is vanishing. Recently, television, which traditionally was managed by separate actors in the market, is increasingly also being provided by telecom operators.

Through the deployment of optical fibers up to (or close to) the home, huge transmission capabilities are rapidly reaching a large sections of users, partially solving the problems that arise when a bandwidth-demanding service, such as TV distribution, IPTV, is carried by the same infrastructure as telephony and data. Thus, the main challenges are now related to the design of efficient architectures for the diffusion of the huge amount of information that is associated with a (possibly large) variety of video contents. One interesting aspect, which is the objective of this paper, is the organization of video contents in some network elements that can store small portions of the contents and then cooperate and exchange contents so as to implement a large memory in a distributed manner. In particular, we consider a hierarchical network architecture in which elements belonging to the same level preferentially cooperate with each other to satisfy the user requests for video contents and that resort to forwarding the requests to the higher layer only when they cannot be satisfied by their level.

Each network element stores video contents, and thus contributes to the service provisioning, according to some caching strategy. We consider a few simple caching strategies based on different degrees of cooperation between nodes and different degrees of information about content popularity. The strategies are compared in terms of the amount of intra-level (i.e., between elements of the same hierarchical level) and inter-level generated requests.

II. Caching strategies

We consider IPTV systems with an hierarchical architecture as shown in Figure 1. The terminals at the users' home are set-top boxes (STBs) that receive desired video contents from the central offices (COs). While each STB is connected to one CO only, each CO controls many STBs and the COs are interconnected, forming a highly connected network: each CO stores a part of the available video contents and video contents can be exchanged between COs. At a higher level of the network hierarchy, some video source offices (VSOs) store video contents that can be provided to the COs of the lower layer, when needed. Similarly, a video hub (VH) composed of a number of servers stores all the contents provided by the system, and it is basically a kind of back-up system since the VSOs can retrieve some video content from the VH. This hierarchical structure allows video contents to be exchanged between peers at the same hierarchical level in a kind of peer-to-peer fashion or, alternatively, to be received by an entity of the higher level, basically implementing a kind of hybrid structured-unstructured peer-to-peer video content distribution system.

When designing caching strategies for the network elements (the COs and VSOs), the main goal is to reduce the cost of video content delivery, while providing an efficient service. In particular, we consider that retrieving a content from a peer at the same hierarchical level is typically *cheaper* than retrieving a content from a higher level. For example, it is preferable that a CO receives a content from another CO rather than requesting it to the VSO associated to it; retrievals from the VSO to the VH are the most expensive ones. Thus, requests to higher levels should be minimized to reduce the cost of retrieving the video contents. Accordingly, we assume that each network element that looks for a video content first tries to retrieve it from peer entities at the same level of the architecture, and that it will issues a request to the higher level only if this peer-level attempt does not succeed.

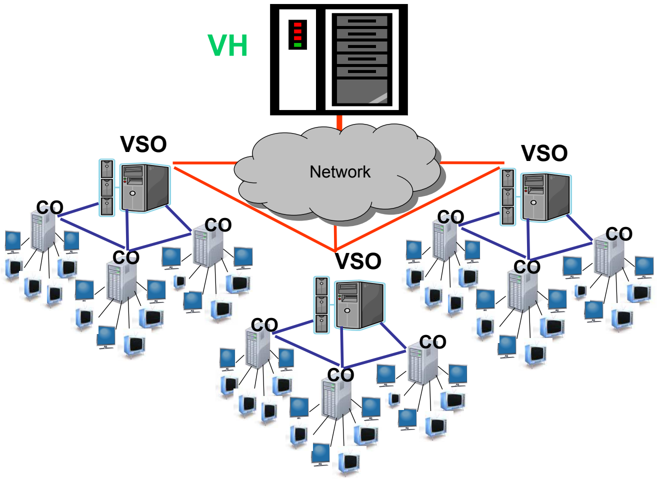


Fig. 1. Hierarchical IPTV system

A basic principle that is common to all the strategies we consider below is that the end users' requests determine and drive the retrieval of video contents. In other terms, the composition of a network-element memory is not decided in advance based on an *a-priori* knowledge of the content popularity, but, rather, content retrievals and exchanges are triggered by users' requests. Once a video content has been retrieved and delivered to the user, it is also stored in the network elements that were involved in the delivery, i.e., the CO and, in some cases, the associated VSO (if that VSO also contributed to the content retrieval). In this way, highly popular contents have more chances of being stored than contents of lesser popularity. The strategies differ in the way room is made in the (limited) memory of the network elements for a newly retrieved content.

The first caching strategy we consider is called *random*. Basically, the content that is dropped to make room in the memory for the newly retrieved video content is randomly chosen from the stored contents. We always apply this strategy at the CO level. Whenever a content is requested to a given CO and the CO does not have it in its own memory, the content is retrieved, preferentially from another CO, or, possibly, from the VSO. To store it in the CO memory, one of the video contents already contained in memory is randomly chosen and dropped. A similar mechanism can be applied to the VSOs. This policy is "selfish" in the sense that, by simply storing contents based on popularity, a CO minimizes the number of requests it generates.

Consider now the VSO level. A VSO receives requests from the COs below it. As popular contents are frequently requested by the users, it can be expected that a large number of duplicate copies of popular contents is present at the CO level and that the COs frequently and easily store or possibly exchange popular contents. In contrast, the low request rate for less popular contents increases the chance that these contents disappear from the CO level network and must be retrieved from the VSO level. This makes the request rate at the VSOs completely different from the exogenous process generated by end users; it is a combination of two mechanisms: i) popular contents are rarely requested to a VSO because they are likely present at the CO level; ii) less popular contents are in general rarely requested, but when this occurs they are probably retrieved from a VSO.

To reduce the probability that the VH is involved in content retrieval, the VSOs may adopt a *collaborative* approach, i.e., an approach in which they collaborate to reduce the probability that a request from the CO level cannot be satisfied by the VSO level. Specifically, we consider two cases, with increasing amount of signaling information:

- **Collaborative Conservative**: The VSOs aim at keeping at least one copy of each content at their level. This means that a video content is stored only if it is already not present at the VSO level and must be downloaded from the VH. Again, the content dropped to make room is randomly chosen.
- **Collaborative Aware**: To decide which content should be stored, the VSOs estimate the contents' popularity and store the less popular contents. As the VSOs do not see the user request generation process, but only the CO requests, they have to exchange some information with the COs. In particular, either the COs or the VSOs should estimate content popularity based on the history of the requests generated in a given time window. Note that once popularity has been estimated, highly popular contents are never stored at the VSO; less popular contents are stored according to the collaborative conservative approach described above.

In the following we provide a model for evaluating the performance of the proposed schemes. In particular, we consider three scenarios, named after the caching strategies adopted at the two levels (the CO level strategy is always random): i) *random and random*, ii) *random and collaborative conservative*, and iii) *random and collaborative aware*.

III. Model

In order to develop the model, we consider a system composed of a number V of VSOs, each of which controls C CO nodes. All the COs are identical, and similarly the VSOs. There are K different video contents in the network. A content k is requested at a CO with rate r_k, which is proportional to popularity and can, thus, be used as a metric of popularity. Here, contents are ordered according to decreasing popularity, i.e. $r_i > r_j$, if $i < j$. A VSO can store B_v contents; a CO can store B_c contents. Notation and definitions are summarized in Table I.

To analyze the system, we propose a model composed of two interacting sub-models: one for the CO level and one for the VSO level. Using an approach similar to the one proposed in [7] in the context of file sharing P2P applications, each sub-model is represented by a set of K continuous-time Markov Chains (MCs), one per each possible content. As shown in

TABLE I
NOTATION AND DEFINITIONS

K	number of contents
C	number of COs
V	number of VCOs
B_c	buffer space in a CO
B_v	buffer space in a VCO
r_k	request rate of content k at a CO
R_k	request rate of content k at a VSO
$\pi_1^c(k)$	diffusion of content k at a CO
$\pi_1^v(k)$	diffusion of content k at a VSO

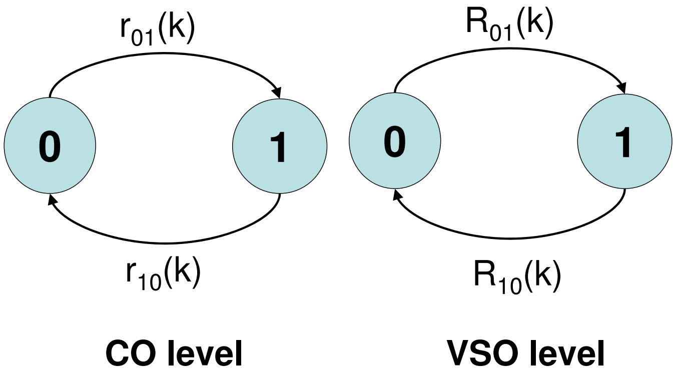


Fig. 2. Two-state MC describing the presence of content k in CO and VSO memory.

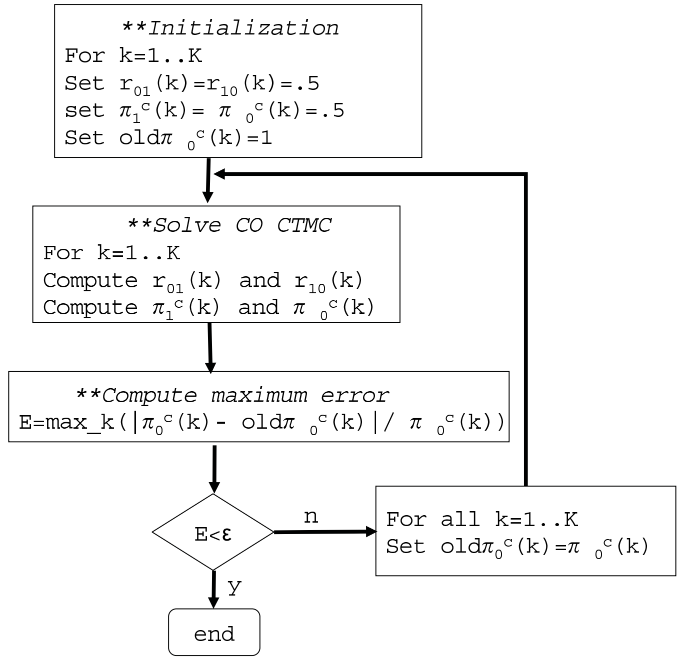


Fig. 3. Fixed-point approximation procedure for the CO level model.

Fig. 2, the MC of an individual content k contains two states, state 0 and state 1, denoting the presence of content k in the considered network element (either a CO or a VSO). State 0 means that the content is not stored in the network element memory, and state 1 that the local memory contains it. The transition rate from state 0 to state 1 in the MC for content k is denoted by $r_{01}(k)$ at the CO level and $R_{01}(k)$ at the VSO level; similarly, the transition rate from state 1 to state 0 for content k is denoted by $r_{10}(k)$ and $R_{10}(k)$ at the CO and VSO level, respectively.

With known request rates $r_{01}(k)$ and $r_{10}(k)$, the probability that a CO has content k in its memory, $\pi_1^c(k)$, is given by the steady-state solution of the MC of a CO.

$$\begin{aligned}\pi_0^c(k) &= \frac{r_{10}(k)}{r_{01}(k)+r_{10}(k)},\\ \pi_1^c(k) &= 1-\pi_0^c(k) = \frac{r_{01}(k)}{r_{01}(k)+r_{10}(k)}.\end{aligned} \tag{1}$$

When content k is requested at a CO and there is no copy of it in the CO network, the request is forwarded to the corresponding VSO. The request rate of content k at a VSO is the product of the total request rate of content k at the CO level, Cr_k, and the probability that k is not stored in the CO network, i.e., no CO has it, $(\pi_0^c(k))^C$:

$$R_k = Cr_k(\pi_0^c(k))^C. \tag{2}$$

The probability that a VSO has content k in its memory can be derived in the same way as for the CO level. The diffusion of content k is the steady-state solution of the VSO MC with request rates $R_{01}(k)$ and $R_{10}(k)$,

$$\begin{aligned}\pi_0^v(k) &= \frac{R_{10}(k)}{R_{01}(k)+R_{10}(k)},\\ \pi_1^v(k) &= 1-\pi_0^v(k) = \frac{R_{01}(k)}{R_{01}(k)+R_{10}(k)}.\end{aligned} \tag{3}$$

As the transition rates of a given MC depend on the other MCs, we use a Fixed Point Approximation (FPA) procedure to solve the K MCs. Moreover, the transition rates depend on the downloading and discarding policies. Thus, we provide three derivations corresponding to the considered strategies: *random and random*, *random and collaborative conservative*, and *random and collaborative aware*.

A. Random and Random

According to the random-random policy, whenever a content k is requested at a CO, it is stored in the memory, and, to make room for it, a randomly chosen content is discarded. Therefore, the transition rate from state 0 to state 1 is simply equal to the content request rate,

$$r_{01}(k) = r_k. \tag{4}$$

A content k is randomly chosen to be discarded from the memory when a new content j is downloaded; the choice occurs with probability $1/B_c$. The download rate of content j is given by the product of the probability of not having j, namely $\pi_0^c(j)$, and its request rate r_j,

$$r_{10}(k) = \frac{1}{B_c}\sum_{j=1,j\neq k}^{K} r_j\pi_0^c(j) \tag{5}$$

As we can observe from (5), the dynamics of the MC associated with k depend on the steady-state of the other chains.

As previously mentioned, we solve the model by means of a FPA procedure, as illustrated in Fig. 3. We first give an initial guess for $r_{01}(k)$ and $r_{10}(k)$ for all the K MCs and obtain the corresponding $\pi_1^c(k)$ from the MC steady-state solution (1). The system solution is obtained by iterating this computation until convergence to a desired precision ϵ has been reached.

For the VSO level, the derivations of $R_{01}(k)$ and $R_{10}(k)$ under the *random* policy are basically the same as for $r_{01}(k)$ and $r_{10}(k)$ at the CO level, except for content k request rate at a VSO, R_k, for which we use (2). The transition rate from state 1 to state 0 at a VSO needs to be scaled by the VSO buffer space, B_v, as shown in the following:

$$\begin{aligned} R_{01}(k) &= R_k, \\ R_{10}(k) &= \frac{1}{B_v} \sum_{j=1, j\neq k}^{K} R_j \pi_0^v(j). \end{aligned} \tag{6}$$

We, again, apply the FPA computation procedure to obtain $\pi_1^v(k)$ at the VSO level with $R_{01}(k)$ and $R_{10}(k)$. Note that FPAs for CO and VSO can be applied to all content management policies considered in this paper, as long as transition rates are provided by the policy models.

B. Random and Collaborative Conservative

The random and collaborative conservative approach consists in keeping one copy of each content at the VSO level. For content k MC, the transition rate from state 0 to state 1, $R_{01}(k)$, is given by the joint probability that the node requests k, R_k, and there is no copy in the other $V-1$ VSOs, i.e., $(\pi_0^v(k))^{V-1}$:

$$R_{01}(k) = R_k(\pi_0^v(k))^{V-1}. \tag{7}$$

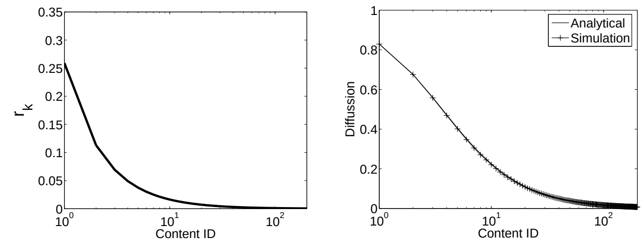
A content k is discarded from a node if it is chosen to be discarded to make room for a newly downloaded content. Therefore, the transition rate from state 1 to state 0 is given by summing the rates at which other contents are requested, downloaded and stored. Once the content is requested at a VSO node at rate R_j, it will only be downloaded from the VH and stored with probability $(\pi_0^v(j))^{V-1}$ because there is no copy of content j available at the VSO nodes:

$$R_{10}(k) = \frac{1}{B_v} \sum_{j=1, j\neq k}^{K} R_j(\pi_0^v(j))^{V}. \tag{8}$$

C. Random and Collaborative Aware

Assuming that contents popularity is estimated by using some measurement and signaling activities between nodes, the random and collaborative aware approach aims at keeping the least T^* popular/requested contents. According to our notation, these contents are indexed from 1 to T^*. As contents from 1 to T^* are not stored, the MCs are not needed. For contents with index from T^* to K, the derivation of the transition rate from state 0 to state 1 remains the same as in (7):

$$R_{01}(k) = \begin{cases} 0 & \text{for } 1 \le k < T^* \\ R_k(\pi_0^v(k))^{(V-1)} & \text{for } T^* \le k \le K \end{cases} \tag{9}$$



(a) Request rate (b) Diffusion

Fig. 4. Request rate at the CO level (a). Diffusion at the CO level (b).

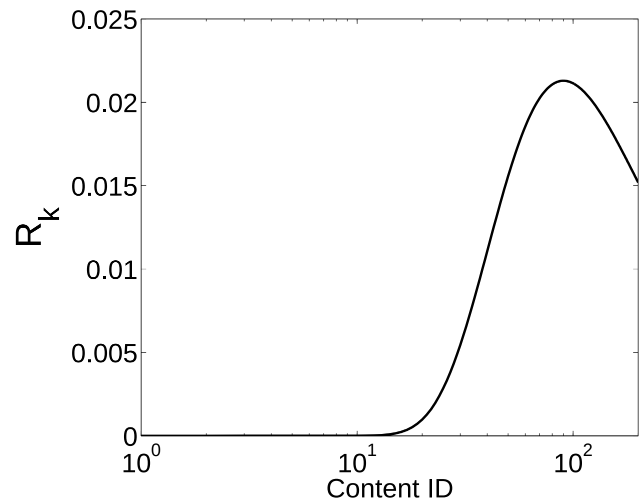


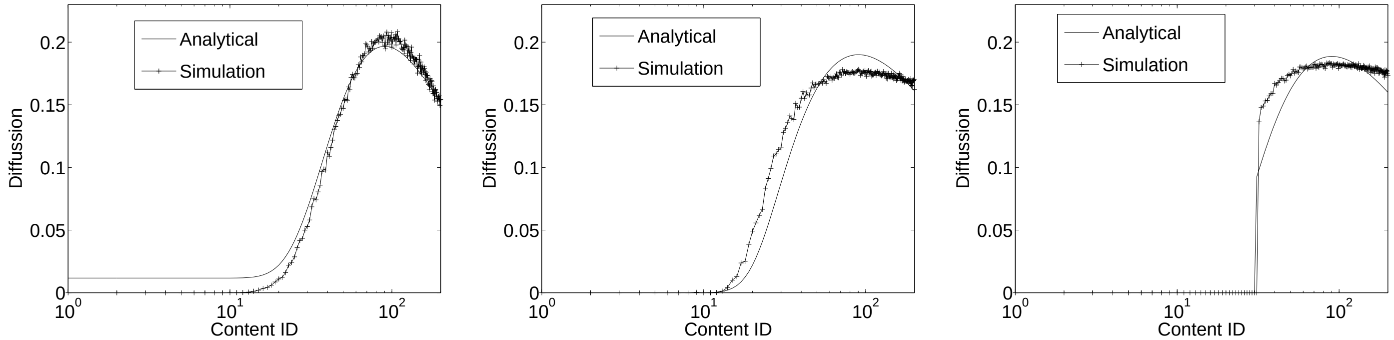
Fig. 5. Content request rate at the VSO level.

Similarly, the derivation of $R_{10}(k)$ is the same as in (8), due to the same discarding policy:

$$R_{10}(k) = \begin{cases} 0 & \text{for } 1 \le k < T^* \\ \dfrac{1}{B_v} \displaystyle\sum_{j=1, j\neq k}^{K} R_j(\pi_0^v(j))^{V} & \text{for } T^* \le k \le K \end{cases}. \tag{10}$$

IV. Experiments

To validate the models and evaluate the proposed policies, we developed a simulator in C environment. The simulator jointly models the two hierarchical levels of the architecture. Similar to what is done with the model, we simulated the random policy at the CO level and three different policies at the VSO level. Results obtained by simulation and the analytical model are then compared. In the simulated scenario, $C = 50$ COs participate in the P2P community under every VSO and $V = 5$ VSOs are considered. The CO node buffer capacity, B_c, is equal to 10 contents. The request rates of the $K = 200$ possible contents at every CO are distributed according to a Zipf's distribution [6] with parameter 1.2, i.e., the popularity of content k translates into a request rate $r_k = k^{-1.2}$. The distribution is depicted in Fig. 4 (a). Every VSO node has a buffer space, B_v, equal to 30. The request rate for every VSO, shown in Fig. 5, depends on the CO level. Note that, since at the CO level we consider only the random



(a) Random (b) Collaborative Conservative (c) Collaborative Aware

Fig. 6. Content diffusion at the VSO level under the three considered policies.

policy, the request rate shown in the figure is the same for the three considered scenarios.

A. Model Validation

Content diffusion at a CO is shown in Fig. 4(b). We can observe that analytical results are very accurate and match simulation results pretty well. The shape of the diffusion distribution for the random policy is very similar to the request distribution, due to the fact that random policy is driven by the request rate, i.e., by popularity: by being more often requested, highly popular files are also more likely stored in the CO memory. We also computed the amount of traffic generated at the COs, that is the number of content downloads that are requested in the time unit; then, we evaluate the fraction of these downloads that correspond to exchanges between COs and the fraction of those that correspond to requests forwarded to the VSO: around 88% of the total traffic is between CO peers under the random policy, the remaining 12% of CO requests are directed to the VSO level and, as Fig. 5 depicts, most of the requests refers to little popular contents.

At the VSO level, the simulation and analytical results are summarized in Fig. 6 with respect to the three policies. Overall, the analytical derivation and simulation results are well validated. As already noticed, the *random* policy results in a content diffusion, whose shape is "similar" to the one of content request distribution. At the VSO level, the VSO buffer under the collaborative conservative is shared by all files, whereas the collaborative aware policy tries to use the VSO buffer only to store the most frequently requested files. Therefore, the content diffusion under the collaborative aware policy has a steeper shape than under the other policies, see Fig. 6(c). Overall, the total VSO traffic is thus lower in the collaborative policies than in random policy as shown in Fig. 7.

We summarize the data about the total VSO traffic in Table II. In addition to generating a higher total traffic, the random policy also generates a higher percentage of traffic to the VH (i.e., the central video hub that acts as a back-up server) than the two collaborative policies. Of all polices, the collaborative aware policy generates the lowest total traffic at VSO level and the lowest traffic to the VH.

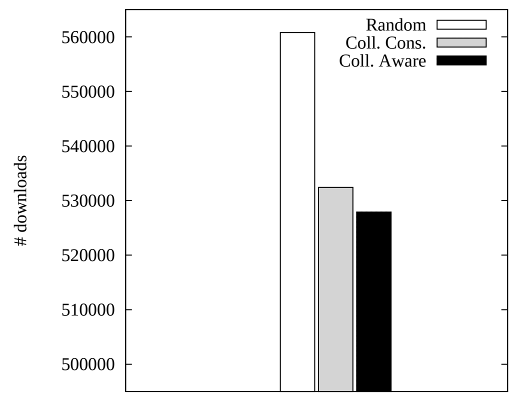


Fig. 7. Total traffic generated at the VSO level with respect to policies. $B_v = 30$

TABLE II

DISTRIBUTION OF VSO TRAFFIC UNDER THE THREE POLICIES. $B_v = 30$

Random		Collaborative Conservative		Collaborative Aware	
To VSOs	To VH	To VSOs	To VH	To VSOs	To VH
53.9 %	46.1 %	81.5 %	18.5 %	84.7 %	15.3 %

B. Further Results

Here, we change the VSO buffer size from 30 to 35. The total VSO traffic generated is expected to be lower in all policies when the buffer space is larger. Figure 8 shows that the proposed collaborative policies can achieve an even higher traffic reduction than the random policy, when the buffer is larger. Moreover, the percentage of traffic to the VH is reduced drastically ($< 5\%$) in both collaborative policies; whereas the random policy has still around 40% of the total traffic going to the VH, as reported by Table III. We can observe that the proposed collaborative conservative and collaborative aware policies have an even better performance with larger buffer dimension. For both the considered values of the buffer space, collaborative aware policy not only generates the least traffic but also efficiently directs most of traffic towards the peer level. It is to be noticed, however, that the collaborative aware policy requires higher signaling information exchange than the other policies, due to the need for estimating video content

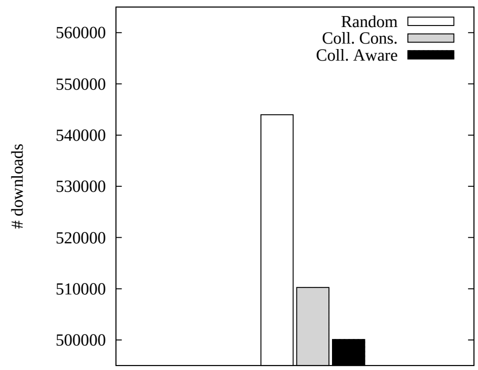


Fig. 8. Total traffic generated at VSO under three policies. $B_v = 35$

TABLE III
DISTRIBUTION OF VSO TRAFFIC UNDER THE THREE POLICIES. $B_v = 35$

Random		Collaborative Conservative		Collaborative Aware	
To VSO	To VH	To VSO	To VH	To VSO	To VH
60.1 %	39.9 %	95.7 %	4.3 %	98.2 %	1.8 %

popularity.

V. Related Studies

Several caching policies based on the access frequencies and size of contents have been proposed to improve the quality of IPTV networks of various architecture. A framework for managing IPTV service deployment is proposed in [2] to improve the subscribers' Quality of Experience (QoE). Sofman et al. [3] partition cache memory among different IPTV services to increase the overall effectiveness of caching. They proposed an iterative method to find the optimal number of cached titles for each service given the cache memory size and throughput limitation, so that the overall cache hit rate is maximized. Simsarian and Duelk [5] show that the amount of bandwidth in the MAN depends on the location of the video servers and cached video content and develop a model of the IPTV network to determine the optimum location of the cached video content. In [4], the authors present a caching algorithm considering an architecture for time-shifted television (tsTV). The algorithm uses sliding caching windows based on content popularity and/or distance metrics for both stand-alone and co-operative mode. Ghandeharizadeh and Shayandeh [1] propose cooperative caching policies requiring high synchronization overhead among peers in wireless home networks, where there are only few devices. Our proposed policies exploit popularity of contents, and rely on the cooperation among peers with the minimum synchronization overhead.

VI. Conclusion

In this paper, we provided analytical and simulation models to investigate inter-level and intra-level content dynamics in an hierarchical IPTV network. Our results showed that a hierarchical IPTV network can provide efficient content storage capabilities with only a limited amount of memory that is distributed in the network nodes. Moreover, since caching policies are essential to well utilize the network and storage resources, two collaborative types of caching policy were proposed to efficiently store contents. Collaborative caching policies improved content retrieval rate with respect to the simple random policy. In particular, the collaborative aware policy has the best performance at the cost of some additional signaling information needed to estimate video content popularity, that is at the basis of the policy.

References

[1] S. Ghandeharizadeh, S. Shayandeh, ”Cooperative Caching Techniques For Continuous Media in Wireless Home Networks”, in Proc. of the 1st International Conference on Ambient Media and Systems 2008, pp. 1-8.

[2] D. Agrawal, M. S. Beigi, C. Bisdikian, K. Lee, ”Planning and Managing the IPTV Service Deployment”, 10th IFIP/IEEE International Symposium on Integrated Network Management, pp. 353-362, 2007.

[3] L. Sofman, B. Krogfoss, A. Agrawal, ”Optimal Cache Partitioning in IPTV Network”, in Proc. of Communications and Networking Simulation Symposium (CNS 2008),2008 ,pp. 79-84, .

[4] T. Wauters, W. Van de Meerssche, F. De Turck, Bart Dhoedt, P. Demeester, T. Van Caenegem, E. Six, ”Co-operative Proxy Caching Algorithms for Time-Shifted IPTV Services”, in Proc. of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO-SEAA'06), pp. 379-386, 2006.

[5] J. E. Simsarian, M. Duelk, ”IPTV Bandwidth Demands in Metropolitan Area Networks”, in Proc. of 15th IEEE Workshop on Local Metropolitan Area Networks (LANMAN 2007), pp. 31-36, 2007.

[6] G.K.Zipf, ”Psycho-Biology of Languages ”, MIT Press,1965.

[7] M. Meo and F. Milan, “QoS Content Management for P2P File-sharing Applications,” *Int. Journal of Future Generation Computing Systems*, Elsevier, Vol. 24, No. 3, March 2008, pp: 213 - 221.