Budapesti Műszaki és Gazdaságtudományi Egyetem

Távközlési és Médiainformatikai Tanszék

Caching és peer-to-peer technikák alkalmazása video-on-demand szolgáltatások optimalizálására

Caching and Peer-to-Peer Techniques to Optimize

Video-on-Demand Services

Tanulmány

Vida Rolland

Készült a Pro Progressio Alapítvány megbízásából

Budapest, 2011.

Abstract

Using some forms of decentralized content distribution is appealing to build scalable access networks, especially when considering the potentially enormous amount of data generated by spreading high quality on-demand video services. There is, unfortunately, little information available about which form of content delivery method would yield maximum benefit and how it should be configured to achieve that, given a certain cost structure and request distribution.

In this technical report we investigate the performance of local caching and P2P delivery solutions, applied separately or in combination, in a video on demand service network. Our analysis is based on simulations and theoretical interpretation of the results. We show that the efficiency of the different methods largely depends on the relative cost of the different transport links and the caching servers, and we provide means to find the optimal method to use in case of a given cost structure of an ISP. We also derive general rules for configuring each of the methods in case the video popularity distribution is known.

1. Introduction

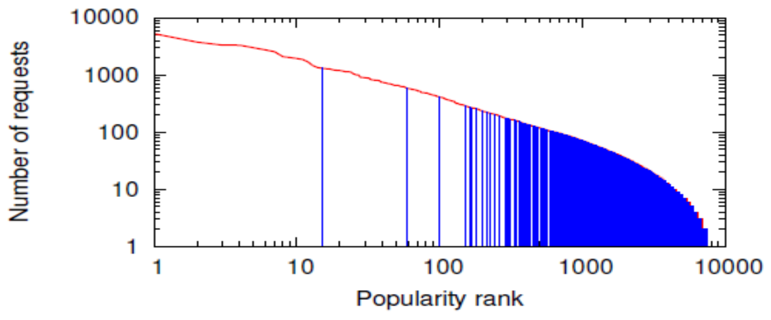
Decentralized content distribution systems are the key factors in building scalable access network infrastructures, especially when it comes to the delivery of the enormous amount of data generated by high quality Video on Demand (VoD) services. Such an architecture benefits the Internet Service Providers (ISP) by reducing link loads and latency, leading thus to an improved service performance.

A typical solution for content distribution is based on the use of caches to replicate contents near the clients [1]. The clients can be redirected to these alternative content sources through various methods, like DNS-based or HTTP-based redirection or URL rewriting, and usually no modifications are required to the generic client applications.

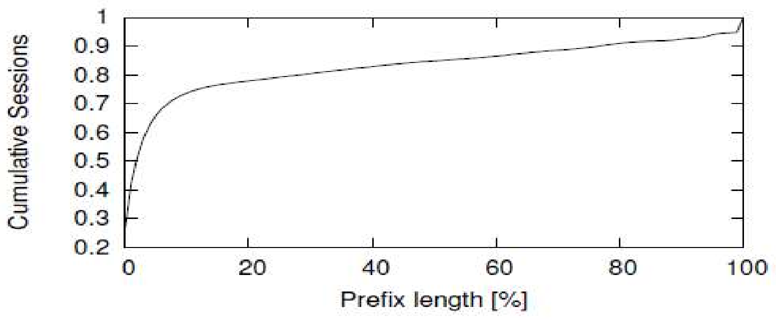
Several different flavours of caching solutions have been already deployed [2]. One of the biggest web content caching services is Akamai, which deploys its replica servers worldwide and uses DNS-based redirection to serve the web object request from the nearest cache [3].

Caching is an efficient solution for data locality problems, but peer-to-peer (P2P) data distribution schemes also offer excellent performance for offloading data sources [4]. These schemes let the clients share the data they have already downloaded; thus, the required network capacity at the content server does not increase linearly with the number of clients. P2P systems can even function without any central content server, and they do not require such a strict management as in case of caching. Peers don't have to be always connected, but join the network on a voluntary basis; thus, the fault tolerance of P2P solutions is usually higher.

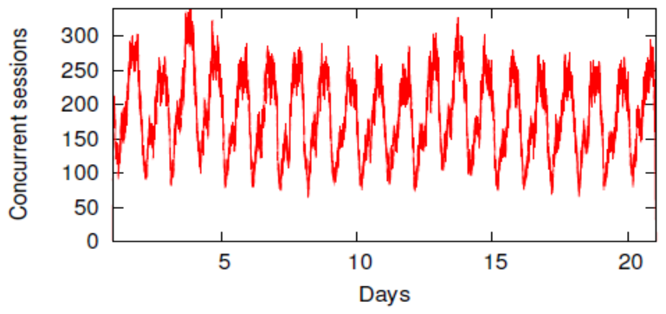
Throughout the years, several different P2P file sharing networks have emerged, and their user base is constantly increasing, despite the numerous legal threats. There were some attempts to build commercial, P2P-assisted dynamic video streaming systems over the Internet [5], but many of them had to be closed due to reasons such as the insufficient number of users, or bad quality.



(1a) video popularity distribution -- the vertical lines indicate videos that were never viewed entirely



(1b) prefix distribution CDF



(1c) number of concurrent video streaming sessions over time

Figure 1. Input dataset generated with MediSyn

The two approaches might even be combined in order to unify the desirable properties of caching and P2P content delivery in a single network. Such mixed systems are supposed to be more efficient than simple P2P and more fault tolerant than caching, but they require more maintenance than the separate systems [6]. An important question in case of such a combined architecture is how to decide whether a request for an object should be served by a cache or by the P2P network?

In this report we analyze a true VoD application, where the clients are served individually and instantly, using local caching and P2P to serve the most popular contents. In this case IP multicast is not practical, because the sessions start at arbitrary times, and cannot thus be aggregated into multicast groups. Although the data transfer is streaming-like, i.e., the clients download the video during playback, the content is not live. Thus, most sessions are able to access the already cached data, stored at the network caches or other clients via P2P. This offloading is not just beneficial, but outright necessary, as the unicast delivery of HD videos consumes enormous amounts of network resources.

To analyze the efficiency of the caching and P2P schemes, and their interactions in the combined VoD system, we develop a simple cost model, which yields easily interpretable results. We assume that the main cost factor in the network is proportional to the used bandwidth, and investigate the total cost of a caching, a P2P and a combined system to find out which one is the most beneficial.

The rest of the report is organized as follows. In section II. we describe the main components of the VoD system: the users, the servers, the caches, and the P2P network. In section III. the simulation results are presented and analyzed, and the conclusions are drawn in section IV. with some outlook on our future work.

2. System Architecture

In this section we define a combined VoD architecture for our simulation study. It is similar in concepts to the one presented in [6]. There are two types of additional content sources offloading the central servers: network nodes acting as caches, and other clients already storing that content, acting as peers. %In this section we introduce the working mechanisms and properties of the three different content delivery methods.

2.1. User Behavior

The performance of a VoD system depends on how well the statistical properties of the user behavior are exploited. Therefore, we have to start the description of our VoD system with the analysis of the video requests.

Similarly to most natural item selection schemes, the contents of a video library are not accessed with the same frequency, but tend to obey Zipf's law [7], i.e., the access frequency is inversely proportional to the popularity rank of the items. This ordered popularity distribution is a straight line on a log-log scale. However, real-life data does not always fit this curve properly; therefore, modified versions of the Zipf-distribution have been developed: Zipf-Mandelbrot law [8], K-transformation [9], and recently the stretched exponential distribution [10].

We generated the video requests of the clients with the MediSyn media server workload generator tool [9]. This tool generates synthetic video request lists that obey all statistic properties its creators measured on a real VoD system. The MediSyn tool generates a Zipf-like long-term popularity distribution with the K-transformation its authors developed to approximate their measurement results. We kept their choice of K=12, but changed the Zipf exponent to α =0.7 to match the popularity distribution of [11], which was a measurement on a public VoD system, as opposed to MediSyn, which is based on a corporate media server.

We have set MediSyn to introduce 50 videos each day, and cut off the beginning of the process until 14,000 videos were accumulated in the system. Our simulations lasted for 20 days; thus, in the end 15,000 videos were available. As the popularities of the videos decrease with time, only around 8,400 different videos were requested during the simulated 20 days; the most popular video was requested 5,237 times. The resulting popularity distribution is shown in Fig. 1a.

In a usual VoD system the videos have different lengths and bitrates, but we chose to make them uniform in order to simplify the interpretation of the results in terms of video popularities. Instead of the MediSyn defaults, we set all videos to be 90 minutes long, with a bitrate of 16 Mbps (assumed to be full HD). Thus, the size of each video file is 10,800 MB, and the content library is 162 TB large. All other parameters (file introduction process, diurnal request intensity pattern, request inter-arrival time) were kept at their default values.

An interesting phenomenon of video streaming systems is the distribution of the incomplete sessions, shown in Fig.1b. Clients usually don't watch the whole movie, but rather browse the catalog; thus, a significant portion of the sessions access only the initial segments of a file, called *prefix*. In the simulations we kept the default prefix settings of MediSyn: only 5% of the sessions were completed, while the mean and the median of the prefix were at 18% and 3% respectively.

The activity of the clients, i.e. the number of parallel video download sessions is also an important property of the system. While the authors of MediSyn don't reveal the number of users, in [11] the service is said to have around 150,000 subscribers in the examined region. Their figures indicate that there are 10,000 video requests per hour in the busy period, and the mean of the prefix distribution is around 10 minutes; thus, there should be around 10,000/6 parallel sessions on average. The maximal client activity in that system is thus around 1.1%. In our simulations we have 10,000 clients, and, as Fig. 1c shows, the maximum activity is around 300 parallel sessions (3%).

2.2. Long Tail Servers

The central content servers hold all the videos that are available for the clients to download and view, but they have two major limitations. First, they are quite far from the clients, which makes them expensive in terms of network usage. Additionally, they might have limited disk I/O capacity, and network bandwidth; thus, they have to be offloaded by other content sources. In the simulations we have one server node, possibly representing the aggregate traffic of more.

The consequence of the Zipf-like popularity distribution of the video requests is that a significant portion of the requests target a relatively small number of contents; thus, replicating the most popular contents near the clients can vastly reduce the load on the central servers, at a price of only a small additional storage. Ideally, the servers serve the least popular videos only, which make up the tail of the popularity distribution; this explains the name Long Tail Server (LTS).

2.3. Local Caching

In our system architecture the cache is a content storage unit attached to a node in the operator's network. It is much closer to the clients than the LTS; thus, serving some of the videos from the cache results in lower network resource usage on the links above the cache.

The efficiency of caching depends on the content selection mechanism: the limited cache capacity should be devoted to the items that are most likely to be needed in the near future. To accomplish this, the usual approach is to constantly monitor the popularity of the videos, and choose the most popular ones among them to be served by the caches. Predicting the popularity changes was also proposed in the literature [12], but we omitted it here, because we wanted to keep the system as simple as possible.

Our caching scheme is quite simple. Video popularities are measured by registering the corresponding requests, while aging the data with the Exponentially Weighted Moving Average algorithm to follow the changes in popularity. If the popularity of a video is over a predefined caching threshold T_c, that is, if the number of requests for it divided by the total number of requests is higher than T_c, then the clients requesting that video get redirected to a cache.

The cache downloads the requested, but yet uncached segments from the LTS, and removes other segments using the Least Recently Used (LRU) algorithm, in case it is full. Note that the redirection does not take into account the availability of the video in the cache, as the caches download contents on demand.

There are several existing solutions for pre-caching the contents to decrease the delay between the user pressing the play button and the video starting [13], but we intend to measure network utilization only.

The optimal T_c can be determined analytically in our case, because the intuitive thinking of ``the most popular contents should be put into the cache" is correct; thus, the optimal T_c is the popularity of the least popular content that still fits into the cache. In this calculation the full size of the videos should be considered without the prefix effect, because the caches store what the clients requested, and, as Fig. 1a shows, the really popular videos have at least one uninterrupted session. This reasoning will be refined a bit in section 3.1. If T_c is set to a value above the optimal threshold, the cache will not be fully utilized, while in case of a lower value some less popular videos will periodically replace each other.

2.4. Peer-to-Peer Exchange among the Clients

Several VoD systems utilizing P2P have been proposed in the literature; some of them offload central servers [14], others are completely decentralized [15]. Many different P2P data distribution schemes have been developed, but BitTorrent is the one that is adapted for VoD systems in most papers [16], because it is the most wide-spread solution.

The BitTorrent protocol [17] is a widely known file sharing scheme enjoying great popularity nowadays. It splits the contents into segments that the clients can exchange among themselves. Peers having all the segments are called seeders, while the others are leechers. The peers find each other via a central node, called the tracker, which registers the seeders and the leechers for the available contents.

Ordinary BitTorrent clients select segments and peers using a tit-for-tat scheme, the best known solution for the repeated prisoner dilemma. This basically consists of two rules: the rarest segments are downloaded first, and the willingness to upload to a peer depends on the previously experienced upload willingness of that peer. It is clear that the tit-for-tat scheme is not very suitable for video streaming, because the playback requires more or less in-order segment arrival. This can be accomplished either by enforcing in-order segment download [18], or by confining the rarest first scheme into a restricted time window [19], or both.

In our VoD system we included a variant of the BitTorrent scheme with in-order download confined into a window to minimize the amount of extra content downloads in case the video download session is interrupted. It would also be suitable for VCR operations (fast forward or rewind).

The clients use two sliding windows to control the downloads. The *fallback window* is immediately ahead of the playback position; not yet retrieved segments from this window are downloaded either from the server, or a cache, if available. The size of this window should be at least

$$\text{Size(fallback)} = 1 + \left\lceil \frac{video\ bitrate}{downlink\ bandwidth} \right\rceil, \tag{1}$$

or else the initiated downloads do not finish before the playback position arrives, and a buffer underrun occurs. In our simulations the downlink bandwidth of the clients is twice the video bitrate; thus, the fallback window was set to 2 segments.

After the fallback window there is a gap, followed by the *P2P window*: the client tries to download segments of the P2P window from other clients. The gap is important, because the downloads are only started in the P2P window, but they might take an arbitrary time depending on the peer uplink speeds. If a segment was not found at any peer or the download hasn't finished in time, the fallback window makes sure that the segment will be downloaded from a reliable source (the LTS or the cache) before the playback position arrives. The size of the gap should thus be

$$\text{Size(gap)} = 1 + \left\lceil \frac{video\ bitrate}{uplink\ bandwidth} \right\rceil, \tag{2}$$

to avoid fallbacks. Naturally, the uplink bandwidth of the clients is not constant, and bottlenecks might also occur elsewhere; thus, the size of the gap should be dynamically adjusted

We assume that the tracker is controlled by the network provider, to which we added the possibility to turn off the P2P swarm for selected contents by sending an empty peer list to the clients, who thus loose the connection to their previously available peers, if any. This is a modification introduced by us; in such a case real BitTorrent clients would keep previously known peers. The reason for this is to verify assumptions of previous papers, which suggested that P2P should be reserved for the most popular contents only (above a predefined P2P threshold T_p).

2.5. Combined Caching and P2P

As we have seen in the previous two subsections, caching and P2P can both offload central video servers, but the way they do it differs greatly. The clients download video segments from caches the same way as they would from a central server: get the segments in order, from the beginning. On the other hand, P2P needs to look ahead for segments, because the clients have significantly slower uplink speeds, and they are rather unreliable. Thus, a combined system needs to prefer P2P download, and only use server/cache as a fallback for segments needed shortly.

In our system both caching and P2P can be restricted to contents that are more popular than predefined thresholds $T_c$$ and T_p respectively. We examined two possible variations for a combined VoD system:

- *Strict:* as proposed in [6], there are three distinct popularity regions (server only, cache only, P2P only), separated by two fixed thresholds (the cache threshold T_c and the P2P threshold T_p); in our implementation, if the two thresholds are equal, caching takes priority, and P2P gets turned off.

- *Independent*: caching and P2P can be turned on and off taking into account their respective popularity thresholds, but they can also be both enabled in a certain popularity region.

Although we define here a P2P popularity threshold, in case of a realistic cost model it is not advantageous to disable P2P for any content.

Caching and P2P affect each other in both schemes; thus, it is not trivial to tell their optimal settings, let alone guess their efficiency compared to the simple caching or P2P systems.

We wanted to examine if separating three distinct popularity phases (server only, cache only, P2P only) is advantageous or not, and if it is, what is the correct order? There may be several other variations for a combined system, but we only examined one: the *Independent* scheme enables caching and P2P solely based on their respective popularity thresholds.

We assume that the caches and the P2P system are controlled by the network operator. The operator might be the content provider as well, but most likely the content provider is a separate organization, which might have to rent the cache capacities and the P2P distribution network. In all cases, it is important to tune the system to optimal performance, which means the smallest storage and the lowest network bandwidth requirements simultaneously, which contradict of course.

2.6. Cost Model

The total cost of a VoD system includes several components, such as the price of the equipment, and the operational expenses. Defining a complete cost model is a very difficult if not impossible task, since the cost structure is operator dependent (e.g., some may lease transport links, some others may have their own networking infrastructure) and also region dependent (significant changes of leased Ethernet lines in mature markets vs. other countries may be seen in the price offerings from 0 and 0).

Our approach is to evaluate the cost gains of a certain scheme assuming a given relation between the different cost components. We also separate the bandwidth-dependent part from the bandwidth-independent part, the latter representing a constant correction factor in our analysis. Our cost model for evaluating the simulation results consists thus of the following three components, corresponding to the three content sources.

We approximate the cost of the LTS as being proportional to its outgoing network bandwidth, which is a good approach for the bandwidth-dependent cost of the servers. Note that the bandwidth-dependent operating costs are usually not directly proportional to the used bandwidth, but in our case the changes are sufficiently small for a linearized model. The LTS cost factor is thus: $CF_{LTS}=BW_{LTS.}$

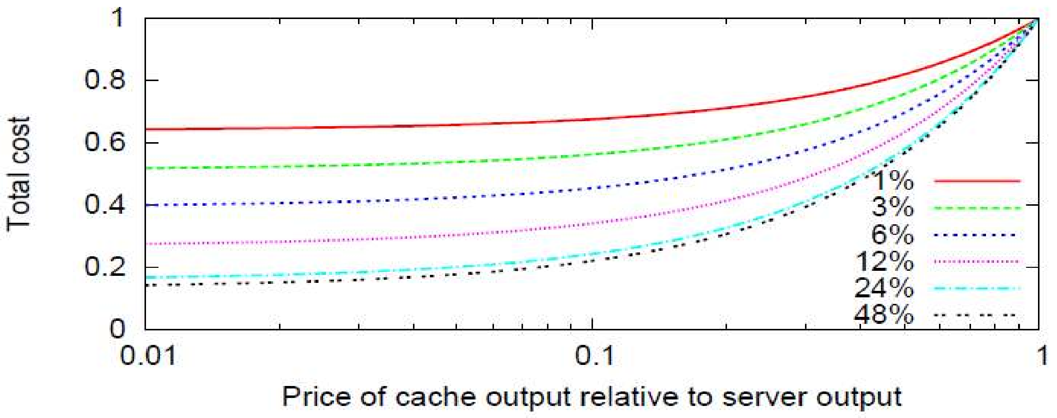


Figure 2. System cost with different cache sizes (relative to the content library)

Similarly, we assume that the cost of the cache is directly proportional to its outgoing bandwidth, because the primary cost factor of a storage unit is its performance in IO operations. The cache cost factor is thus $CF_{cache} = BW_{cache}$.

The cost factor of the P2P distribution is proportional to the available uplink capacity of the clients, because its efficiency depends on it. The uplink channel might even be allocated for the VoD service by the network operator, and pricing advantages or similar incentives might be used for keeping the client equipments on (probably Set Top Boxes (STB) supplied by the network operator). The P2P cost is thus

$$CF_{P2P} = \sum_{i \in clients} BW_{uplink_i}$$

If the P2P distribution was limited with a popularity threshold, the allocated uplink capacities would not be utilized; thus, the only tunable parameter of the P2P system is the allocation amount.

The total cost of the system is the weighted sum of the three cost factors:

$$TC = P_1CF_{LTS} + P_2CF_{cache} + P_3CF_{P2P}, \qquad (3)$$

where the P_i values are the unit prices of the cost components. As we have no precise information on these prices, and they might very well differ from ISP to ISP, we will treat them as variable parameters in our analysis.

All three cost factors will be normalized relative to the outgoing server bandwidth in the server-only case. The unit prices will also be given relative to P_1. Beyond the generality of the results, the main advantage of this normalization is that the cost gain of specific setups can be directly read from the plots, and the disadvantageous ones (with $TC > 1$) are easy to identify.

3. Simulation Results

We conducted several simulations to examine the behavior of the VoD system and to find the optimal values of its control parameters. We wanted to simulate a large number of users over a reasonably long time period, for which packet-level simulations would have been too slow. Thus, we decided to simulate the transfer of the videos on segment-level. As we found no suitable event-driven data transfer simulator available, we developed our own from scratch. In this section we describe our simulation setup, which consists of the statistical behavior patterns of the simulated user base, the network topology, and some implementation details, and present our findings on each system variant.

The network topology in our simulations consists of one video server (possibly representing the aggregate traffic of several servers), 10,000 clients, and a node in between, which acts as a cache. The cache and the clients have fixed storage capacities, but only the latter ones have limited outgoing bandwidth; the cache is assumed to be able to serve all active clients simultaneously (which are only 3% of the nodes in our simulations, as previously specified). The clients have enough storage for one complete video, but in most installments the uplink of the clients is significantly slower than their downlink, and we wanted to examine the effect of this limitation on the efficiency of the P2P distribution scheme.}

The T_c popularity threshold will be given in percentage of the total amount of video requests, cache sizes in percentage of the whole video library, and bandwidths relative to the video bitrate. The costs will be normalized, so that the outgoing traffic of the server is 1 in the server only case, and the unit-price of the server link is 1. A setup is thus advantageous, if the corresponding total cost is less than 1.

3.1. Caching only

In this case the P2P distribution is turned off; thus, the two cost factors are only the outgoing traffic of the server, and that of the cache. Fig. 2 shows how the size of the cache affects the total cost of the system, when the cache runs with an optimal T_c. The curves intersect at 1 (i.e., if the cache cost equals the LTE cost), and the total cost is never higher than 1 (i.e., using caches always decreases the costs), because the total amount of traffic does not depend on the cache size, and the bandwidth-independent cost factors are not included. We omitted them because we have no precise and up-to-date information on storage costs (relative to network bandwidth costs). Adding that cost to the total would shift all curves upwards with different constant.

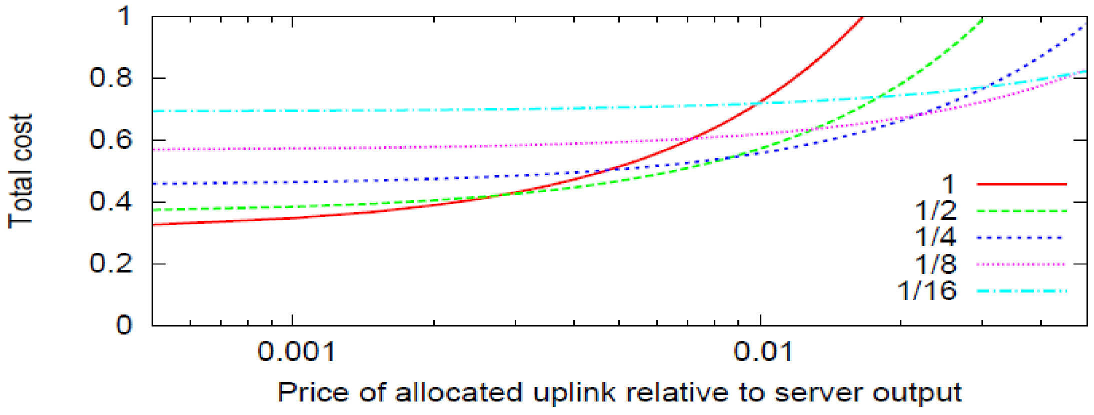


Figure 3. System cost at different uplink speeds relative to the video bitrate

Initially we used the method described in section 2.3 to set the T_c, which proved to be slightly inaccurate due to the continuous change of the video popularities, and the introduction of new videos each day. This might be due to the MediSyn tool, which generates the same popularity distribution for each day.

The real optimum is slightly lower than the predicted, which probably means that the LRU algorithm successfully eliminated the less popular segments, keeping space for the popular parts of more videos. The interesting threshold range is below 1%, but the exact values of the thresholds are not important, as they highly depend on the popularity distribution. The values are only given to show how the optimal settings change with the cost structure.

The accuracy of the calculation further decreased as the cache size was increased, because there is a limited subset of videos requested within a day, and most of the new videos are not introduced in the busy period; thus, the optimal T_c turned out to be 0% for the 12% and larger caches. The largest reasonable cache would be 8,400/15,000=56% in our case, because it can hold every video that is requested at least once.

The caches offload the servers, but their cost function are different. As Fig. 2 shows, the cost gain is not proportional to the cache size, and the optimal cache size highly depends on the cost structure, unless the cache cost is very low. It is also worth noting that the total cost can be much higher than 1.

3.2. P2P only

The cost of the P2P content distribution is comparable to the caching scheme. The gradual shifting of the optimal settings in function of the price of the uplink is shown in Fig. 3 for different uplink capacities. In the simulations all clients had the same uplink capacity, but in a real installment this is not necessarily true. The higher the allocated client uplink is, the more cost gain can be achieved, but only if the uplinks are sufficiently cheap. Fig. 3 only shows the lines until TC = 1, but it is apparent that installing the P2P system might increase the total cost.

Due to the low activity of the users, the allocated client uplink is much larger than the total traffic of the VoD system; thus, the price range shown in Fig. 3 is twenty times cheaper than the one in Fig. 2. The maximal gain shown in Fig. 3 is lower than what is achievable with caching: a cache of 12% is more effective than any of the shown P2P configurations.

It is not visible in Fig, 3, but the P2P scheme transfers slightly more data than what is actually needed; the P2P download window is far from the playback position, so segments are downloaded in advance even if the user stops the video before those segments get played. In a cost model that includes the usage of client downlink, this extra download would penalize the lower uplink speeds, as the P2P window should be set even further away from the playback position.

In this section we also analyze whether it is advantageous to reserve the capacity of the P2P swarm to the more popular contents. Fig. 3 shows the case, when the occupied uplink bandwidth is the cost factor, and the uplinks have 1/2 capacity. The system cost decreases, if we lower the T_p value; thus, it is not advantageous to turn off the P2P network for any content, unless the client uplink is nearly as expensive as the server link. The turning point is around 0.87, and the maximum yield is comparable to a 6% cache.

Due to the prefix phenomenon the clients download slightly more data with the P2P scheme than in the server only or caching case, because the P2P window is far from the playback position. This is the reason why the P2P system increases the cost if the client uplinks are more expensive than 0.87.

The number of parallel downloads can be increased by widening the P2P window, but in our experience it only increases the superfluous download without any real benefit. Theoretically it might occur that a segment is only found at already occupied peers, where a P2P window larger than 1 segment would allow retries, but it is a rare event, as the client activity is maximum 3%; thus, at most 6% of the clients upload something simultaneously in case of a 1/2 uplink.

Fig. 3 shows results for 1/2 uplink, with the P2P window placed at 5 segments away from the playback position to avoid fallbacks. With a slower uplink the server output would increase, due to the less efficient P2P and the extra download would also increase considerably because of the increased gap.

The clients would also need to find more peers to keep the download speed in par with the playback speed, which is possible, since their activity is very low. We omitted showing our results for client cache sizes other than one full video, because only the numeric values differ, but the conclusions remain the same.

3.3 Combined

The combined system has two parameters to tune: the Tc cache threshold, and the allocated client uplink. We chose the 6% cache for analyzing the combined system, because the P2P distribution is able to achieve about the same gain with sufficient uplink capacity. Fig. 4 shows the total cost and the optimal settings for the interesting price ranges (which is again lower for the P2P uplink allocation than for the cache output).

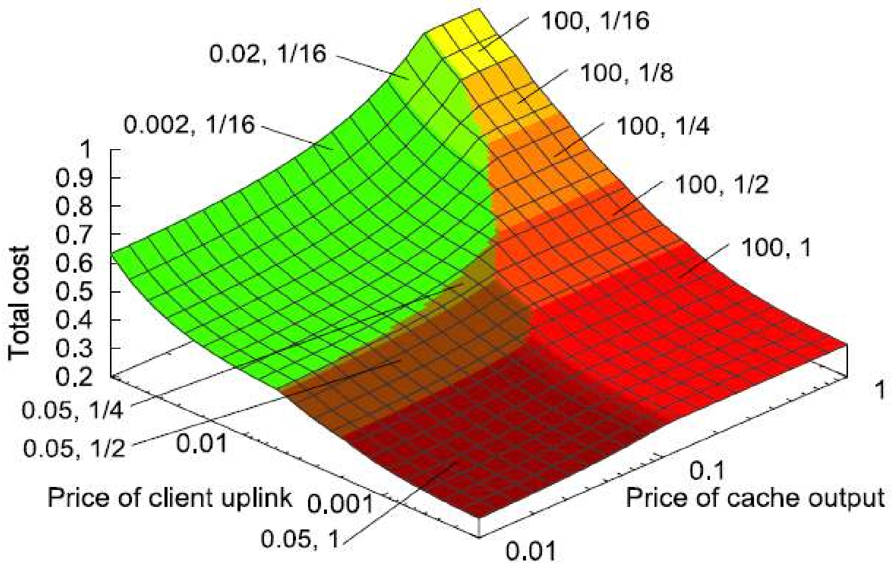
The surfaces shown in the figures correspond to the achievable minimum total cost, and the coloring shows the optimal settings: the red component is proportional to the Tc (more red means higher Tc and less cache usage), and the green component is proportional to the allocated uplink speed (more green means slower uplink). As in the cache only case, the exact values of the thresholds are not important, they are only given to show how the optimal settings change with the pricing.

Comparing the achievable gain in Fig. 4a and Fig. 4c, we can find that the Independent scheme has slightly lower cost at the same price combinations. In the left corner, where the uplink is expensive and the cache is cheap, the total cost is 0.63 and 0.57 with the *Strict* and *Independent* schemes respectively. In the right corner, where the uplink is cheap and the cache is expensive, the cost is 0.31 for both schemes. In the bottom corner, where both schemes are cheap, the cost is 0.26 and 0.21 respectively. With this cache size a total cost of 0.4 is achievable, and the P2P only scheme can reach 0.31 with the fastest uplink.

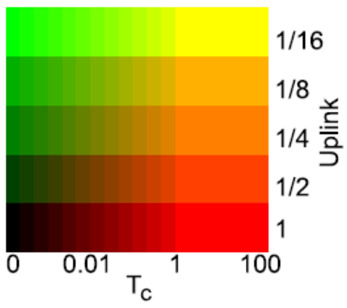
In the right corner the *Strict* scheme reverted to the P2P only scheme by turning off the expensive caching completely. In the Independent scheme; however, the efficiency of the P2P distribution is not affected by Tc; thus, the additional caching simply offloads the LTS without crippling the P2P.

In the left corner both schemes perform worse than the cache only would, because the expensive P2P delivery couldn't be turned off completely. Thus, the 10% extra gain of the combined system is not significant.

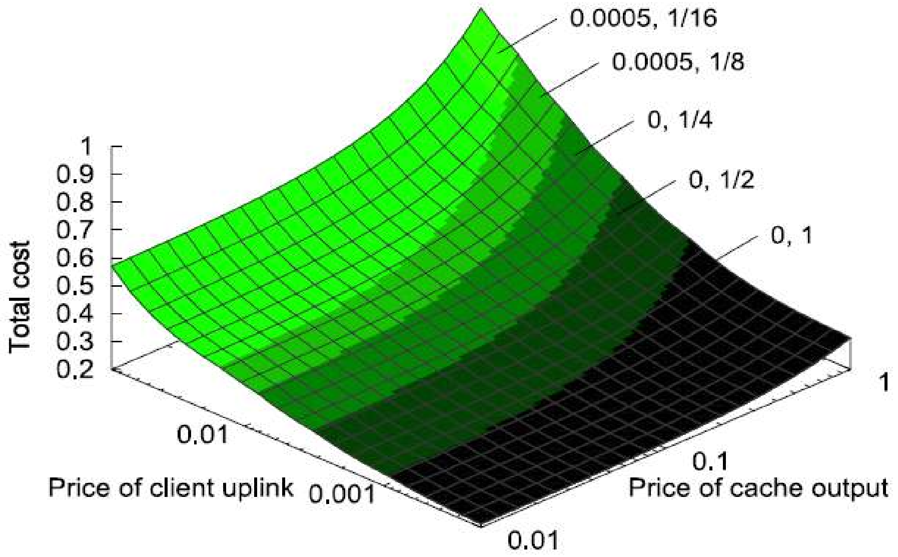
The cost difference between the two schemes is not that large, but the shapes and the colors of the areas with different optimal settings make a huge difference. In case of the *Strict* scheme the shown price range is divided into several small areas of very different colors, while in case of the *Independent* scheme there are fewer but larger areas with significantly lower color variation. This means that the Independent scheme is much easier to set up for optimal performance.



(a)



(b)



(c)

Figure 4. Total cost as a function of the price of the allocated client uplink and the cache output for the Strict (4a}) and the Independent (4c) schemes. The shown plane is the minimum of the cost planes corresponding to the different threshold and uplink settings, and the coloring, explained in 4b, reflects the optimal parameters for each price combination. The corresponding T_c, uplink' values are also given for each major area.

The 3D figures show that with both schemes there is a threshold combination that yields cost reduction for all cost structures, because neither planes exceed a total cost of 1. The size of the cache was chosen to be 6% in order to get the same 45% gain in the cache only (left corner) and P2P only (right corner) case. In the front corner, where the unit price of both offloading schemes is 0.1, the total cost of the *Strict* scheme is 0.37, and of the *Independent* scheme is 0.31.

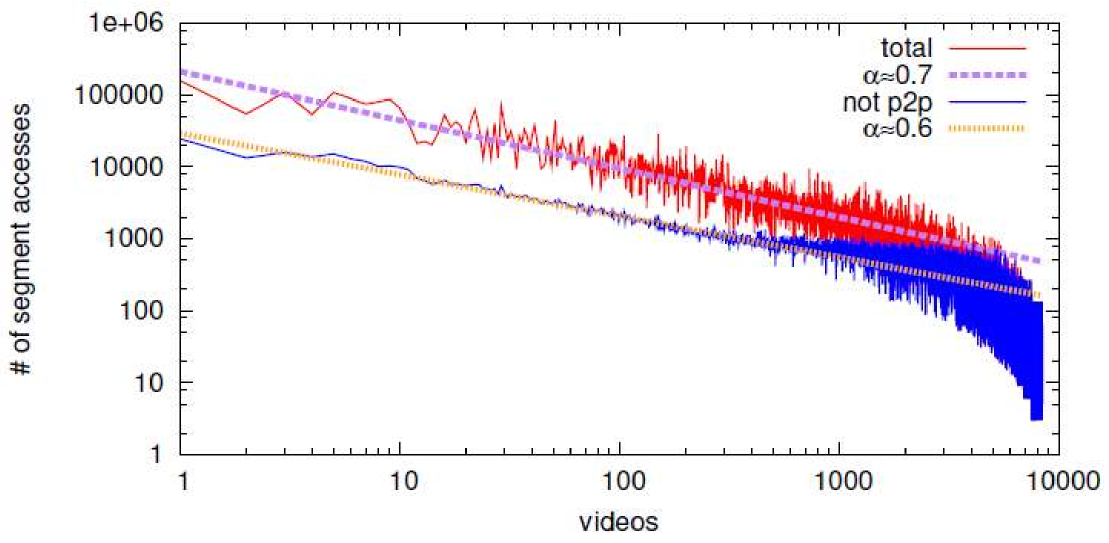


Fig. 5 Total number of segments, and the amount not served by P2P (Tp=0%, videos are ordered by request popularity): the remaining server load has different popularity steepness

The optimal Tc cache threshold becomes different with the P2P system turned on, because it changes the popularity distribution seen by the cache. Fig. 5 shows the number of segments served from each video, and the Zipf-alpha parameter (approximated with least squares method) for both the total number of segments, and the portion not served by the P2P system. As the steepness of the two popularity curves differs, Tc has to be optimized for the modified Zipf-parameter. This means that although the same number of videos are allowed to be cached in both cases, they represent a different percentage of the total number of requests.

4. Conclusions

In this report we analyzed a Video on Demand system, where caching and P2P delivery are both available to offload the central servers. Both offloading schemes have tunable parameters: they might be restricted to the most popular videos with popularity thresholds, and their efficiency depends on the cache sizes and the available uplink capacities of the clients. As expected, there is always a substantial gain with each type of distributed delivery, provided the schemes are optimally tuned. With a simple cost model we have sought the optimal parameter combinations for a wide price range.

We provided a formula for approximating the optimal cache threshold T_c, if the video popularities are known, but due to the incomplete sessions the real optimum is slightly lower, i.e., caching the popular parts of less popular videos increases the efficiency. We also have seen that it is not advantageous to restrict the P2P scheme to the most popular videos, but the optimal amount of allocated uplink capacity strongly depends on the pricing.

We examined two variants of the combined system: the *Strict* scheme, which has been proposed in the literature, turns off the P2P distribution for the videos popular enough to be cached, while our *Independent* scheme handles the two subsystems independently. The latter scheme realized slightly larger cost gains, and it also turned out to be substantially easier to set up for optimal cost, because its optimal settings do not depend that much on the pricing. The achievable cost gain of the combined system over the individual ones is not much; though, and in real installments the additional bandwidth-independent cost factors might make the combined system even less profitable.

We found that the combined system can outperform the individual ones; however, we recommend checking the bandwidth-independent cost factors before deployment, as they might erode the cost advantage, which is quite fragile; one expensive component can make it more expensive than the individual systems. If one has to choose between the two individual systems, we recommend caching, as it can achieve higher cost gain than P2P, unless the uplinks are faster than the video bitrate, and upgrading its size is probably easier than adding more uplink speed.

References

[1]. G. Pallis and A. Vakali, “Insight and perspectives for content delivery networks,” *Communications of the ACM*, pp. 101–106, January 2006.

[2]. Stefan Saroiu, Krishna P. Gummadi, Richard J. Dunn, Steven D. Gribble, and Henry M. Levy, “An analysis of internet content delivery systems,” in *Proc. of Fifth Symposium on Operating Systems Design and Implementation (OSDI)*, 2002, pp. 315–327.

[3]. Akamai Technologies Inc., “How “akamaization” works,” 2000. [Online]. Available at: http://www.akamai.com/html/about/press/releases/2000/press_061300.html

[4]. Meeyong Cha, Pablo Rodriguez, Sue Moon, and John Crowcroft, “On next-generation telco-managed P2P TV architectures,” in *Proceedings of IPTPS’08*, Tampa Bay, FL, USA, February 2008.

[5]. E. Setton and B. Girod, *Peer-to-peer Video Streaming*. Springer, 2007.

[6]. Frederic Fieau, Mohamed Fouz Menai, and Nathalie Omnès, “Peer-AHNA: An adaptive network architecture for IPTV services,” in *Proceedings of ICIN2008*, Bordeaux, France, October 2008.

[7]. G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949.

[8]. Benoit Mandelbrot, *Information Theory and Psycholinguistics*. Penguin Books, 1968.

[9]. W. Tang, Y. Fu, L. Cherkasova, and A. Vahdat, “Modeling and generating realistic streaming media server workloads,” *Computer Networks*, vol. 51, no. 1, pp. 336–356, 2007.

[10]. Lei Guo, Enhua Tan, Songqing Chen, Zhen Xiao, and Xiaodong Zhang, “The stretched exponential distribution of internet media access patterns,” in *Proceedings of PODC08*, Twenty-Seventh Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, Toronto, Canada, August 2008.

[11]. Hongliang Yu, Dongdong Zheng, Ben Y. Zhao, and Weimin Zheng, “Understanding user behavior in large-scale video-on-demand systems,” in *Proceedings of EuroSys ’06*. Leuven, Belgium, ACM, April 18th - 21st , 2006, pp. 333–344.

[12]. X. Chen and X. Zhang, “A popularity-based prediction model for web prefetching,” *IEEE Computer*, vol. 36, pp. 63–70, March 2003.

[13]. James S. Gwertzman and Margo Seltzer, “The case for geographical pushcaching,” in *Proc. of HOTOS ’95*. Washington, DC, USA: IEEE Computer Society, 1995, p. 51.

[14]. T. T. Do, K. A. Hua, and M. A. Tantaoui, “P2VoD: Providing fault tolerant video-on-demand streaming in peer-to-peer environment,” in *Proc. of ICC 2004*, Orlando, FL, USA, 2004, pp. 1467–1472.

[15]. D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava, “On peer-topeer media streaming,” in *Proc. of ICDCS02*. Vienna, Austria: IEEE Computer Society, 2002, pp. 363–371.

[16]. A. Vlavianos, M. Iliofotou, and M. Faloutsos, “BiToS: Enhancing BitTorrent for supporting streaming applications,” in *Proc. of INFOCOM’06*, Barcelona, Spain, April 2006, pp. 1–6.

[17]. B. Cohen, “Incentives build robustness in BitTorrent,” in *Proc. of 1st Workshop on Economics of Peer-to-Peer Systems*, 2003.

[18]. Y. R. Choe, D. L. Schuff, J. M. Dyaberi, and V. S. Pai, “Improving VoD server efficiency with BitTorrent,” in *MULTIMEDIA ’07*. Augsburg, Germany: ACM, 2007, pp. 117–126.

[19]. P. Savolainen, N. Raatikainen, and S. Tarkoma, “Windowing BitTorrent for Video-on-Demand: Not all is lost with tit-for-tat,” in *Proc. of GLOBECOM’08*, New Orleans, LO, USA, December 2008, pp. 1–6.

[20]. OpenReach BT Access Portfolio
http://www.openreach-communications.co.uk/ethernet/our-products/access.aspx.

[21]. Bharat Sanchar Nigam Ltd.- Leased Line tarrifs
http://www.bsnl.co.in/service/leased_tariff.htm#high