Dynamic Server Configuration for Multiple Streaming in a Home Network

Laurent Lemarchand, Isaac Armah-Mensah, Jean-Philippe Babau
Univ. Bretagne Occidentale, UMR 6285, Lab-STICC, F-29200 Brest, France
Email:{laurent.lemarchand,isaac.mensah,jean-philippe.babau}@univ-brest.fr

***Abstract*—In home network, to manage network bandwidth usage, one solution is to control the server outgoing traffic with a token bucket policy. Hull-based token bucket parameter setting allows the guarantee of Quality of Service for variable bitrate video streaming. Hull is an abstraction of the bitrate, following the bandwidth requirement evolution dynamically. In the case of multi-streaming, we investigate a shifting technique to reduce the peaks impact. Postponing the streaming starting time of a video helps to decrease the maximum required bandwidth. The technique is then mixed with the hull-based reservation. Simulations show the effectiveness of the combined approaches to optimize bandwidth usage, guaranteeing the best QoS for streaming. Online utilization is also discussed.**

I. Introduction

Home networks now interconnect a lot of heterogeneous devices to offer distributed streaming services at home. A classical usage is to start a streaming application between a server (a stored video in the home's gateway) and a client (a viewer). In this instance of close devices in a closed system, the user expects the best Quality of Service (QoS), ensured by providing the necessary quantity of resource at run-time (the network bandwidth in the paper).

The home network designer is faced with two challenges. The first is to provide a mechanism to reserve enough resource for multimedia streams which are typically encoded with a Variable BitRate (VBR). Thus, the required bandwidth fluctuates during the streaming, with numerous peaks. Moreover, the solution has to be implemented on legacy systems : home networks are based on legacy technologies like TCP for transport protocols and QoS standards.

In the literature, to solve the problem of QoS guarantee for VBR transmission, the research proposals are based on a complex and accurate model of VBR [1]. From these models, resource reservation may be done with a QoS-oriented negotiation between the clients and the server like in [2]. These approaches, based on statistical model and QoS control loops, are dedicated to open, dynamic and adaptive systems following a best-effort paradigm, see iDASH [3] for Video on Demand (VoD).

But, in the specific context of closed systems, since we have a full knowledge of contents and architecture, we argue that we can control resource usage precisely using a centralized architecture and an accurate knowledge of bitrate to transmit [4]. In this approach, the network admission is classically solved by limiting the outgoing traffic on server devices with a token bucket that shapes the outgoing traffic.

The problem relies then on the tocken bucket configuration. One may set the token bucket parameters to a maximal value of bitrate [5] to guarantee the best QoS. But this solution results in an over-reservation. The reservation is based on the worst bitrate value, generally a non representative peak. Token bucket parameters may be set to an average value to optimize the link usage but it may increase the network delay [6]. A trade-off is proposed in [7], with an adapted joint rate and admission control.

Following this idea, we proposed an original approach in [8] based on an optimal hull (a worst case required bitrate) of VBR. Optimality regards average resource usage and implementation constraints. Hull is used to compute token bucket parameters to allocate network bandwidth dynamically, following as precisely as possible bitrate evolution and ensuring the best QoS (minimum delay and data loss).

The hull-based approach works for giving priority to a video stream transiting within the home network. But, if a new (or a set of new) movie(s) are to be started simultaneously triggering competition for the resource, priority cannot be used to decide between streams for admission. In case of multiple streams, a first idea is to produce a hull for the added VBR (the current streaming video and the new one). But multiple streams induce a peak problem, not solved by the hull definition. Figure 1 illustrates this issue: if videos A and B streaming start at the same time, a peak appears at date 3. So before hull definition, one needs to reduce peaks. The solution we investigate consists in delaying some videos. By delaying B 1 second, the peak decreases by 20 units.

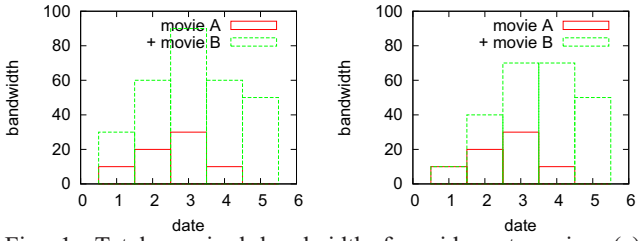

Fig. 1: Total required bandwidth for video streaming (a) without and (b) with a start delay for the second video

In the context of home network, the current streaming cannot be delayed (user requirements), the problem is then to find the best delay for others videos, with the objective of reducing peaks. The delay is limited by the maximum

978-0-7695-5249-1/14 $31.00 © 2014 IEEE
DOI 10.1109/EUC.2014.15

IEEE computer society

Authorized licensed use limited to: University of Cape Coast. Downloaded on December 02,2020 at 14:47:50 UTC from IEEE Xplore. Restrictions apply.

acceptable delay to start a new video streaming.

The paper proposes a video shifting algorithm to find the best video starting time : when a user asks for a contents, how much to postpone the begining of his video streaming in order to minimize the peak bandwidth request during the streaming, taking into account the other running streaming process requirements during the period. The paper also investigates mixing the reservation algorithm proposed in [8] with the shifting mechanism. Last, performances are measured in order to check if the best shifting and hull computation can be made online, when a new video is ready to start, within an acceptable delay for the end-user.

The paper is organized as follows. Section 2 details related works. Background and formalization of the problem is described in section 3. Section 4 presents the shifting algorithms. The way of mixing them with the reservation policy proposed in [8] is explained in section 5. Section 6 is devoted to simulation results.

II. Related works

There is a large number of works about VBR Video traffic models, see [9] for more details. All these works propose to develop statistical predictive models of VBR behavior. On the other hand, if one can compute the VBR itself, deterministic services can offer guarantee for delay and loss rate. It results into a low network utilization [10] because it needs to reserve the bandwidth according to the peak rate. To avoid this problem of deterministic approaches, we propose a new original approach, sharing common ideas with some statistical predictive approaches, even if some are relatively old research works.

In [11], the authors propose a simplified hull of the empirical envelope [12] to integrate implementation constraints. And to provide deterministic QoS, [13] proposes a bounding version of such characterization. We share with these works the idea of a simplified view of the VBR to integrate implementation constraints and to provide deterministic QoS.

Because of time-independence, the problem of these approaches is that the required bandwidth may be very high to ensure no delay, considering peak rate. RED-VBR [14] and RCBR [15] propose flexible re-negotiation services to reduce the impact of burstiness of VBR video. The idea is to segment the video into sequences and to reserve adequate resource for each segment. RCBR mechanism [16] proposes a dynamic bandwidth allocation strategy to support VBR video traffic. More recent work [17] proposes another accurate real-time bandwidth reservation and on-line prediction. Following these works we want to adapt online the network reservation, following VBR evolution, in order to limit peaks impact.

All these works concentrate on predicting the required quantity of resource (here the network bandwidth). On the other hand, the Matrix framework [18] proposes an abstraction of device state (offered quantity of resource) through a timing evolution of discrete QoS levels. This abstraction acts as a hull of offered quantity of resource, a QoS level is then the minimal value of available quantity of resource. We share with this work the idea of making an abstraction of VBR through a timing evolution of discrete required bandwidth (worst bit-rate) levels

These approaches are developed for individual streams. In case of different videos, various smoothing algorithms [19] have been proposed for statistical multiplexing. The idea is to reduce cumulative peaks impact either by renegotiating bandwidth allocation [20], or by removing or simply delaying some frames [21]. In this work, we propose both to renegotiate bandwidth allocation and to delay some videos diffusion to both reduce the impact of cumulative peaks, and limit the problem of over-reservation induced by deterministic services.

We introduce some notations and previous work before presenting our contribution.

III. Background and Models

a) Hull based approach for single stream: We propose an original approach in [8] based on an optimal hull (worst case required bitrate) of VBR. Optimality regards average resource usage and implementation constraints. Hull is used to compute token bucket parameters of the streaming server to allocate network bandwidth dynamically, following as precisely as possible bitrate evolution and ensuring the best QoS (minimum delay and data loss). In [8], the network bandwidth required for a video streaming is represented as a sequence of values $b = \{b_1, ..., b_n\}$ called bitrate. b_i is the bandwidth required during time slot i and is expressed in *e.g* kbits/s. Such bitrate is computed from the stream file by extracting the size of the frames. A *hull* is another sequence of values $h = \{h_1, ..., h_n\}$ such that $\forall i \in [1..n], b_i \leq h_i$. It represents a network bandwidth reservation sequence. Since reservation is always higher than the requested bandwidth, QoS is ensured. Based on graph theory, the optimization algorithm minimizes the overcost (bandwith over-reservation) defined as $\sum_{i=1}^{n}(h_i - b_i)$. Obviously, if $b_i = h_i$, overcost is null. However, this is not realistic since network equipments cannot be reconfigurated as often as bitrate levels vary. The optimization algorithm models this constraint by taking into account two parameters : a reconfiguration frequency P (minimal number of consecutive identical h_i values) and/or a maximal number of reconfigurations M (number of times where $h_{i+1} \neq h_i$). Overcost is minimized according to the P and M selected values. A movie bitrate series, and a possible reservation hull are depicted on figure 2. The optimized cost corresponds to the red area.

b) Handling multiple streams: Let a set of videos $\{B^i\}$ to be diffused at the same time and their associated bitrate series $\{B_1^f, B_2^f, ..., B_{nf}^f\}$. $\{B_i^f\}$ represents the bandwidth requirements of video $\{B^f\}$ for the period i. Let d_f be the delay for video f and Δ its maximum. Let the video 1 be the current streaming and cannot be delayed ($d_1 = 0$). It represents the video (or the set of videos) already started, with the amount of data that have to be transmitted related to this (these) running streaming(s). One goal is then to compute the starting times d_f for ($f \in [2..m]$), for the not yet started streaming(s), such

Authorized licensed use limited to: University of Cape Coast. Downloaded on December 02,2020 at 14:47:50 UTC from IEEE Xplore. Restrictions apply.

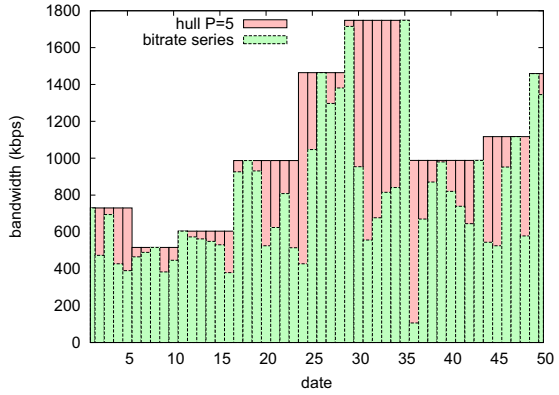

Fig. 2: A bitrate series and the optimal reservation hull according to constraint $P = 5$ (minimal time before reconfiguring the bandwith reservation level)

that $d_f \leq \Delta$ and

$$max_t \sum_{f=1}^{m} B^f_{d_f+t} \quad (1)$$

is minimum. Times t cover the whole time period of streaming.

The optimization objective defined by eq. 1, by limiting the peak, tries to limit the maximum reservation locally. As in [8], we also keep the global objective of reducing over-cost reservation induced by hull definition.

IV. Shifting Algorithms

The problem of optimizing starting times for many videos can be solved approximately or exactly. The former can be realized by processing the videos one by one. The latter corresponds to a search tree traversal.

c) Heuristic algorithm: In this case, the starting time of one video is optimized considering fixed starting times for previous videos.

Let's consider the case of a two videos set $\{B^a, B^b\}$. B^a starts at 0, and the first algorithm simply computes the peak value for of all the allowed times in interval $[0..\Delta]$ for B^b. The time leading to the minimal cumulative bandwidth peak is selected. Complexity is $O(\Delta.n)$, with n as the video bitrate size. A $O(\Delta.n.m)$ complexity algorithm can be derived for m videos (algorithm 1). At each step k, B^a corresponds to the cumulative bitrate series of videos 1 to k, according to their starting times, and B^b is the $(k+1)^{th}$ video, B^{k+1}. Notice that the solution obtained is exact if $m = 2$ (remind that first video B^1 starts at time 0 and cannot be delayed). It computes an approximate solution if $m > 2$.

d) Exact algorithm: On the contrary, the optimal solution for a set of more than two videos is found by enumerating all of the starting times combinations in the $[0..\Delta]$ interval for videos 2 to m. It corresponds to the traversal of a search tree (see figure 3). Each tree level except the root represents a video which can be postponed. A node has Δ children, corresponding to the possible shifts. A path from root to a leaf corresponds to a set of starting times for the set of videos. The exact algorithm finds the optimal solution by exploring the whole search tree. Its complexity is $O(\Delta^m)$.

Algorithm 1: Heuristic computation of starting times

```
Data: Δ, B = {B^1, B^2, ..., B^m}
Result: d = {d_1, d_2, ..., d_m}
begin
    d_1 ← 0
    for f ∈ [2..m] do
        d_f ← ∞
        for d ∈ [0..Δ] do
            obj ← max_t Σ_{f=1}^{m} B^f_{d+t}
            if obj < d_f then
                d_f ← obj
            end
        end
    end
end
```

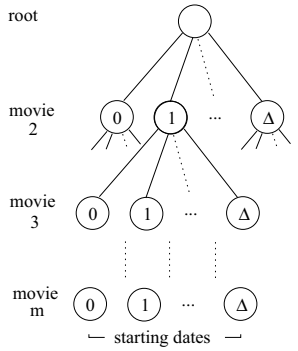

Fig. 3: Exact computation of starting times : the search tree explorated for checking all starting times combinations. Each combination is a path from root to a leaf

V. Tests on shifting

In this section, we investigate both the run-time and the quality of the two algorithms. The minimal peak is computed by the exact algorithm, providing a bound on the quality of the heuristic. The algorithms have been implemented in C, run-time is measured onto an Intel i7, 2.8 GHz system with 8GB of memory. Test videos are Psych movie series episodes of approximately 45 mn each, encoded with H264/avc1, 23.97 fps of size 720 x 402. Bitrates used are sampled per second, leading to approximately 2700 bitrate series. Peaks are around 5000-6000 kbps.

A. Run-time

Run-time depends on the value of Δ (which defines the size of the search space), and both the number and the length of

Authorized licensed use limited to: University of Cape Coast. Downloaded on December 02,2020 at 14:47:50 UTC from IEEE Xplore. Restrictions apply.

the videos. The impact of the former is tested first.

Figure 4 depicts run-time in milliseconds for both exact and heuristic algorithms, for computing the starting times of 1 to 3 new movies. Run-time is given according to Δ values (from 0 to 100 s). The curves reflect the complexity of algorithms: $O(\Delta.m)$ for the heuristic, $O(m^{\Delta})$ for the exact algorithm, if m is the number of movies, of a given fixed length. For 1 or 2 new movies, the exponential factor has not too much impact on the exact algorithm run-time, but it explodes with the 3^{rd} movie. Heuristic algorithm is 1000 times faster than exact one for a Δ of 100. Regarding the actual time of computation, the heuristic algorithm returns in less than 3 ms, which makes it applicable online.

If one remembers that heuristic algorithm provides the exact solution if only one video is to be added, this algorithm is obviously to be used in this single addition case. Furthermore, the heuristic behaves efficiently for very large values of Δ, even if users won't accept a delay of more than a few seconds.

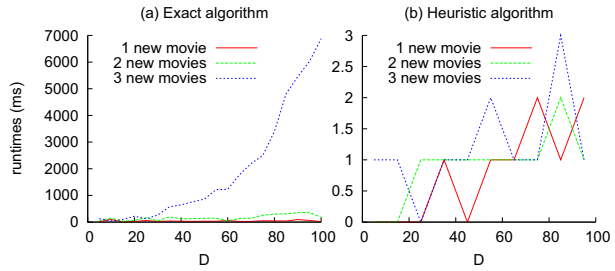

Fig. 4: Run-times in milliseconds for up to 4 movies according to Δ

Concerning the number of videos, the exact algorithm run-time gets worse rapidly with the increase in number of videos. For example, with $\Delta = 10$ and 7 videos, the exact algorithm runs within 10 s, while the heuristic is still at the millisecond.

The second video parameter that influences the run-time is the lengths of the videos, corresponding to the number of times to be compared. Figure 5 shows the evolution of the run-time of the exact algorithm for various video lengths, with $\Delta = 5$.

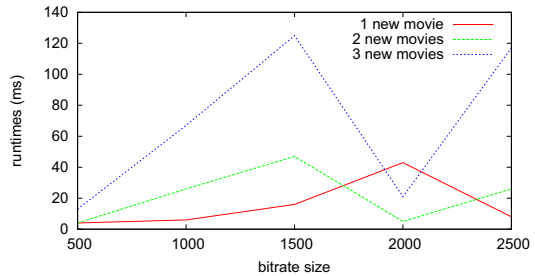

Fig. 5: Run-times of exact algorithm for up to 4 movies of variable lengths, with $\Delta = 5$

Theoretically speaking, the shortest the bitrate lengths, the better are the run-times, since the peak search computation time is linear in term of video length at each step. However, if a good solution is found early, the peak computation is aborted early in subsequent configurations evaluations. This explains the irregularities for the 2000 and 2500 points in figure 5. Notice that heuristic run-time is about 3 ms for $\Delta = 5$ and 4 2500-length movies.

B. Quality of solutions

The goal of the shifting is to minimize the peak value for combined videos reservation. The figure 6 shows comparative results for both the exact and heuristic algorithms. 1 to 3 new movies are considered, with values of Δ in the range $[2..100]$. The results obtained can be compared to the case when no shifting policy is applied, e.g if all of the movies start at the same time.

	optimal algorithm			heuristic		
	# new movies			# new movies		
Δ	1	2	3	1	2	3
2	8482	8872	9539	8482	8872	9539
3	7365	7725	8233	7365	7725	8233
4	7365	7725	8233	7365	7725	8233
5	7166	7526	8010	7166	7526	8141
10	6915	7297	7873	6915	7340	7913
20	6915	7297	7873	6915	7340	7913
50	6915	7292	7821	6915	7340	7913
100	6683	6973	7667	6683	6973	7750

Fig. 6: Results comparison for exact and heuristic algorithms. Minimal bandwith peaks (in kbps) for 1, 2 and 3 added movies when maximum allowed shifting Δ varies for 2 to 100. Without any shifting strategy (simultaneous starting time for all movies), peak is of 9225 (resp 10030, 10741) kbps for 1 added movie (resp. 2, 3)

First, adding shifting is efficient: if no shifting is considered, peak values are more than 8 % higher, even for low values of Δ. For example, if a single movie is added, a delay of $\Delta = 2$ units (e.g. 2 seconds) leads to a peak value of 8482 kbps instead of 9225 kbps if the second movie starts immediatly. Second, the shifting results show that the heuristic is efficient. Tests with $\Delta \geq 5$ are the only configurations where the peak can be higher with the heuristic, but still less than 1.6% more than optimal value provided by the exact algorithm. Highest values of Δ considered here are theorical : postponing the starting time of a movie streaming more than a few seconds will not be acceptable for end-users. Furthermore, as shown in figure 4 (a), the exact computation of starting times is time consuming when the number of movies increases.

In terms of peak values, larger allowed shifts can lead to better results. For 1 added movie, increasing Δ allows peak to decrease from 8482 to 6683 kbps (21% gain). For 3 movies, it reduces from 9539 to 7750 kbps (18% gain).

We now evaluate the impact of shifting on dynamic reservation based on a hull.

Authorized licensed use limited to: University of Cape Coast. Downloaded on December 02,2020 at 14:47:50 UTC from IEEE Xplore. Restrictions apply.

VI. Reservation hull optimization

The network cannot be reconfigured as often as requirements vary with a VBR encoded video. [8] proposes an algorithm for optimizing the bandwidth allocation (see section I). Remind that a series of configurations called *hull* is computed, defining bandwidth allocation levels, that must cover the video bitrate series. The hull definition is constrained by the number of different allowed configurations (M value) and the minimal period between 2 reconfigurations (P value). The algorithm minimizes the cost of bandwidth reservation ($\frac{\text{allocated bandwith}}{\text{required bandwidth}}$). In the following, it is noted as a percentage, e.g a ratio of 1.38 is denoted as 38%.

The hull algorithm has been designed for a single bitrate, but it is applicable to a set of bitrates by considering the cumulative values of the series.

In this section, we investigate how hull-based reservation and shifting mechanism interact together. There are two approaches for the mixing process, depicted in figure 7.

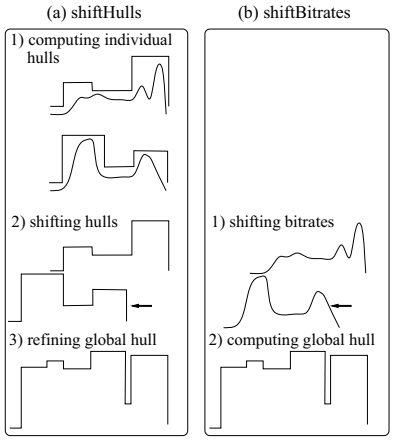

Fig. 7: How to mix hull reservation with shifting for peak optimization procedure

(a) *shiftHulls* hulls are first computed individually for each stream. The shifting algorithm is then applied on the hull shapes for finding the best starting times, leading to a *minimal peak for the hulls*. Due to the shifts, reconfigurations induced by each hull may be non compatible. Thus, the hull computation is to be repeated on the merged shifted hulls.
(b) *shiftBitrates* shifting algorithm is applied directly on the bitrate series, then hull computation is made on the merged shifted bitrates.

As an example, consider series `A = 5 8 7 9 9 8` and `B = 3 2 2 2 5 6`. With *shiftHulls*, with P=2, we obtain hulls `hA = 8 8 8 9 9 9` and `hB = 3 3 2 2 6 6`. Shifting with $\Delta = 3$ leads to a shift of 2 for `hB` and a peak value of 12. The resulting cumulative values series is `hA+hB= 8 8 8 12 12 11 2 6 6`, with period P non respected. The hull must be recomputed. The result is `hAB = 8 8 8 12 12 12 6 6 6` of cost 78.
With *shiftBitrates*, for the same P and Δ values, `B` is also shifted of 2, leading to a peak of 11 for the cumulative series `A+B = 5 8 10 11 11 10 5 6`. It leads to a final hull `hAB = 8 8 11 11 11 11 6 6` of cost 81.

First approach seems more complex but can leads to better results. The run-times of the overall process and the quality of the resulting hulls were investigated for the two considered approaches (a) and (b). They are discussed next.

A. Quality of solutions

Shifting aims at minimizing the peak for cumulative bandwidth of two or more videos. Hull reservation targets to limit over-cost reservation induced by the hull definition (remember that a hull is a worse bound of required bitrate depending on the P and M parameters). So in this section, we investigate:

- the influence of combined hull definition: what is the benefit of considering conjointly the videos instead of making independent hull reservations;
- the impact of shifting technique on hull-based reservation performance;
- the impact of hull parameters on shifting.

First, we evaluate the interest of making a global reservation and the influence of the shifting onto the hull. The evaluation is made for two videos for different Δ values. The over-cost is expressed as a percentage of the sum of the considered bitrates.

Hull constraints	global hull over-cost for $\Delta \in [1..20]$	over-cost for 2 distinct hulls
P = 5	21.6 – 22.7%	29.1%
P = 10	37.4 – 39.4%	49.1%
M = 1200	4.1 – 4.3%	5.0%
M = 600	11.3 – 11.8%	14.3%

Fig. 8: For various hull building constraints (P and M values), global hull quality intervals for $\Delta \in [1..20]$ for 2 videos of 2400s each. Also cost ratio if bandwidth allocation is computed separately for the 2 videos instead of being made on the cumulative bitrates of the 2 videos

The table 8 shows the over-cost values obtained when Δ varies between 1 and 20.

A first conclusion on quality is that hull computation must be made globally for minimizing the over-cost of separate reservations: over-cost is always higher if allocation is made individually for each of the videos, instead of computing a common bitrate series first. The second conclusion is on the fact that over-cost intervals are very tight: shifting the start of a video doesn't modify significantly the hull quality for these simple test cases.

To evaluate impact of hull definition on peaks, we evaluate the approaches (a) and (b) for the two different shifting techniques on a more complex case. Table 9 shows the impact onto the peak value, and on the over-cost for a set of 4 videos.

This example shows that over-cost, if linked to P and M, is relatively independent of the shifting technique and integration approach used. The difference is less than 6.67% in any case.

Authorized licensed use limited to: University of Cape Coast. Downloaded on December 02,2020 at 14:47:50 UTC from IEEE Xplore. Restrictions apply.

		exact shift		heur. shift	
hull constraints	integration approach	peak	over-cost	peak	over-cost
P5	(a)	8339	16.2%	10741	16.5%
	(b)	8010	16.1%	8141	16.8%
P10	(a)	10741	28.7%	10741	28.7%
	(b)	8010	28.6%	8141	28.9%
M1200	(a)	8077	3.6%	8094	3.5%
	(b)	8010	3.4%	8141	3.6%
M600	(a)	8077	9.5%	8293	9.6%
	(b)	8010	9.0%	8141	9.4%

Fig. 9: Final hulls cost and peak values with exact and heuristic shifting algorithm for (a) *shiftHulls* and (b) *shiftBitrates* approaches. 4 videos and $\Delta = 5$

Concerning peaks, exact method confirms its superiority, with sometimes a large gap (e.g 8339 to 10741 (+ 28.8%) for P5 with approach (a)). We also notice an impact of the integration approach for peak minimization. Except in one case (M1200 with heuristic shifting), the approach (b) *shiftBitrates* produces better peak results than (a).

With the influence of hull parameters on shifting performance, we notice that the obtained minimal peak is independent of P and M, if the integration approach (b) is chosen. For approach (a), a difference of 33 % (for P=10 and M=1200) is possible.

To conclude, shifting does not reduce nor increase significantly the over-cost for the global hull. On the other hand, P and M parameters have influence on peak minimization only if the integration approach is (a) *shiftHulls*. And *shiftHulls* is less efficient than *shiftBitrates* for the peak optimization aspect.

B. Run-time

This section is devoted to compare run-times of the two approaches (a) *shiftHulls* and (b) *shiftBitrates*. The following aspects are considered. (1) *Hulls provide a simplified view of bitrates series*. This could help reduce the run-time for the shifting algorithm in *shiftHulls* and allows to extend the applicability of the exact shifting algorithm (see fig. 5 for the impact of input video length on shifting algorithm run-time). (2) *The relative run-time cost of the algorithms* for the 2 phases is evaluated, including the cost of pre-computed hull for (a).

We consider here the shifting of 4 videos with bitrate sizes of around 2400 values (one value per second). The exact (E) and the heuristic (H) shifting methods with $\Delta = 5$ are evaluated. Table 10 shows the different results.

The evaluation shows that the main cost is due to hull computation, as compared to shifting. So (b) *shiftBirates* is always faster, since it saves the individual hulls computation run-time. The consequence is that (a) *shiftHulls* is interesting only if individual hulls are pre-computed. In this case, run-times are comparable (see the second line (a')).

For this example, shifting and global hull computation run-times are of the same order for P5 or P10. M1200 and M600 techniques are much more costly but stay under acceptable limit of 1 sec.

hull constraints	approach	ind. hulls	shift (E)/(H)	global hull	total time (E)/(H)
P5	(a)	91	16/2	22	129/115
	(a')		16/2	22	38/24
	(b)		12/1	22	34/23
P10	(a)	96	2/0	27	125/123
	(a')		2/0	27	29/27
	(b)		8/0	27	35/27
M600	(a)	226	8/1	56	290/283
	(a')		8/1	56	64/57
	(b)		13/0	52	65/52
M1200	(a)	161	10/1	73	244/235
	(a')		10/1	73	83/74
	(b)		13/0	42	55/42

Fig. 10: Total and step run-times in milliseconds for (a) *shiftHulls* and (b) *shiftBitrates* approaches. Steps are individual hulls for (a), shifting and final global hulls for both approaches. $\Delta = 5$. For the shifting step, (E): exact and (H): heuristic

However, if the run-time cost of shifting increases due to a high number of videos and/or a high Δ value (see section V-A), heuristic algorithm (H) is to be used, especially for online computation.

VII. Simulation

In this section, we simulate the approach using NS2 [22]. In order to shape the bandwidth needs, a typical approach consists of using a Hierarchical Token Bucket (HTB), mainly characterized by its throughput rate parameter. As an example, HTB is used in Unix kernels. For the sake of simplicity, only 10 minutes long portions of the test movies couple are simulated. Simulations correspond to the streaming of 2 movies through a network, with a common bandwidth allocation. The average throughput of the two movies is 768 and 925 kb/s respectively (a total of 1693 kb/s for both movies). The maximum lateness of frames and the maximum buffered data are measured as QoS indicators. The case A is the reference case : it is realized without shifting and without hull computation. Case B corresponds to the most promising approach, *shiftBitrates*, with $P = 10$ and $\Delta = 5$ for hull constraints. The obtained peak values for cases A and B are 5519 and 4987 kbps respectively.

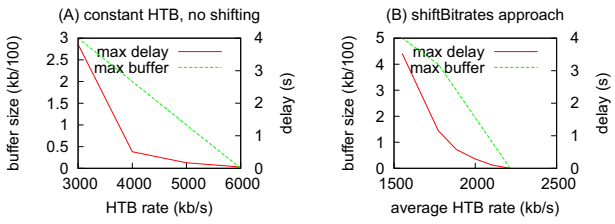

Fig. 11: Delays and Buffer size for 2 videos streaming according to HTB throughput rate. (A) without variable bandwidth allocation, (B) with the *shiftBitrates* approach with parameters $\Delta = 5$ (shifting) and $P = 10$ (hull constraints),

Authorized licensed use limited to: University of Cape Coast. Downloaded on December 02,2020 at 14:47:50 UTC from IEEE Xplore. Restrictions apply.

Results are shown in figure 11, for the bandwidth ranges (x axis) allowing to reach the QoS requirements in each case. In case A, a bandwidth allocation of 5500 kbps is needed for satisfying the 0 delay QoS requirement. On the other side, in case B, an average allocation of approximately 2215 kbps is sufficient to obtain the same QoS with the *shiftBitrates* approach, comparable with the average data throughput of 1693 kb/s. Notice that the shifting technique decreases the peak from 5519 to 4987 kbps, limiting peak impact on other network traffic. We also evaluate hull without shifting, and obtain an average allocation of 2237 kbps for a zero-delay result, confirming that shifting does not reduce hull performances (2215 kbps with shifting).

Finally, the simulation shows that mixed approach helps both on minimizing peak and reducing mean bandwidth allocation.

VIII. Conclusion

This paper proposes a way to shape bandwidth allocation for videos streaming in a home network. It consists of first shifting the start of new videos to avoid cumulative peaks. Then, reservation is based on a cumulative hull of the obtained bitrates. Overall run-times measured range from 50 to 250 ms following the approach and parameters chosen. The shaping (shifting and hull definition) can thus be computed online when starting a new video, or resuming a paused one. Simulations show that both delays and buffer consumption can be decreased using the proposed approach, e.g. with an average bandwidth of 2200 kbits/s instead of 6000 kbits/s for a 0-delay and around 0-buffer streaming. We are now working on implementing the technique as a specific reservation protocol on Linux.

References

[1] A. Alheraish, "Autoregressive video conference models," *Network Management*, vol. 14, no. 5, pp. 329–337, 2004.

[2] D. Bethanabhotla, G. Caire, and M. Neely, "Utility optimal scheduling and admission control for adaptive video streaming in small cell networks," *IEEE Int. Symposium on Information Theory*, pp. 1944–1948, 2013.

[3] Y. Sánchez de la Fuente, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. De Vleeschauwer, W. Van Leekwijck, and Y. Le Louédec, "idash: Improved dynamic adaptive streaming over http using scalable video coding," *MMSys'11, ACM Conf. on Multimedia Systems*, vol. 8, pp. 257–264, 2011.

[4] M. Louvel, P. Bonhomme, J.-P. Babau, and A. Plantec, "A network resource management framework for multimedia applications distributed in heterogeneous home networks," *Int. Conf. on Advanced Information Networking and Applications (AINA)*, pp. 724–731, 2011.

[5] K. Nahrstedt and R. Steinmetz, "Resource management in networked multimedia systems," *Computer*, vol. 28, no. 5, pp. 52–63, 1995.

[6] E. Hernández and J. Vila, "A new approach to optimize bandwidth reservation for real-time video transmission with deterministic guarantees," *Real-Time Imaging*, vol. 9, pp. 11–26, 2003.

[7] R. Yu, Y. Zhang, C. Huang, and R. Gao, "Joint admission and rate control for multimedia sharing in wireless home networks," *Comput. Commun.*, vol. 33, no. 14, pp. 1632–1644, 2010.

[8] L. Lemarchand, M. Louvel, and J.-P. Babau, "VBR Video Abstraction for Home-Network Reservation," *Int. Conf. on Embedded and Multimedia Computing (EMC-12)*, vol. 181, pp. 113–122, 2012.

[9] S. Tanwir and H. Perros, "A survey of vbr video traffic models," *IEEE Commun. Surveys Tuts.*, vol. 15, pp. 1778 – 1802, 2013.

[10] J. Zhang and J. Hui, "Applying traffic smoothing techniques for quality of service control in {VBR} video transmissions," *Computer Communications*, vol. 21, no. 4, pp. 375–389, 1998.

[11] J. Liebeherr and D. Wrege, "An efficient solution to traffic characterization of vbr video in quality-of-service networks," *ACM/Springer Multimedia Systems Journals*, vol. 6, pp. 271–284, 1998.

[12] D. Wrege, E. Knightly, H. Zhang, and J. Liebeherr, "Deterministic delay bounds for vbr video in packet-switching networks: fundamental limits and practical trade-offs," *IEEE/ACM Trans. Netw.*, vol. 4, no. 3, pp. 352 –362, 1996.

[13] E. Knightly and H. Zhang, "D-bind: an accurate traffic model for providing qos guarantees to vbr traffic," *IEEE/ACM Trans. Netw.*, vol. 5, no. 2, pp. 219 –231, 1997.

[14] H. Zhang and E. Knightly, "Red-vbr: a renegotiation-based approach to support delay-sensitive vbr video," *Multimedia Systems*, vol. 5, pp. 164–176, 1997.

[15] M. Grossglauser, S. Keshav, and D. Tse, "Rcbr: a simple and efficient service for multiple time-scale traffic," *IEEE/ACM Trans. Netw.*, vol. 5, pp. 741–755, 1997.

[16] A. Adas, "Using adaptive linear prediction to support real-time vbr video under rcbr network service model," *IEEE/ACM Trans. Netw.*, vol. 6, no. 5, pp. 635 –644, 1998.

[17] S. Kang, S. Lee, Y. Won, and B. Seong, "On-line prediction of non-stationary variable-bit-rate video traffic," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1219–1237, 2010.

[18] L. Rizvanovic and G. Fohler, "The matrix - a framework for real-time resource management for video streaming in networks of heterogeneous devices," *Int. Conf. on Consumer Electronics (ICCE 2007)*, pp. 1–2, 2007.

[19] W. Feng and J. Rexford, "Performance evaluation of smoothing algorithms for transmitting prerecorded variable-bit-rate video," *IEEE Trans. Multimedia*, vol. 1, no. 3, pp. 302–312, 1999.

[20] Z. Wang, H. Xi, and G. Wei, "A relaxing bandwidth smoothing schedule for transmitting prerecorded vbr video in periodic network," *Multimedia Systems*, vol. 16, no. 2, pp. 151–168, 2010.

[21] Z. Zhang, J. Kurose, J. Salehi, and D. Towsley, "Smoothing, statistical multiplexing, and call admission control for stored video," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 6, pp. 1148–1166, 1997.

[22] S. Mccanne, S. Floyd, and K. Fall. (2007) ns2 (network simulator 2). [Online]. Available: http://www-nrg.ee.lbl.gov/ns/

Authorized licensed use limited to: University of Cape Coast. Downloaded on December 02,2020 at 14:47:50 UTC from IEEE Xplore. Restrictions apply.