An Upload Bandwidth Threshold for Peer-to-Peer Video-on-Demand Scalability *

Yacine Boufkhad
Paris Diderot University, LIAFA, France.
boufkhad@liafa.jussieu.fr

Fabien Mathieu
Orange Labs, Issy-les-Moulineaux, France.
fabien.mathieu@orange-ftgroup.com

Fabien de Montgolfier
Paris Diderot University, LIAFA, France.
fm@liafa.jussieu.fr

Diego Perino
Orange Labs, Issy-les-Moulineaux, France.
diego.perino@orange-ftgroup.com

Laurent Viennot
INRIA Project-Team "GANG" between INRIA and LIAFA, France.
laurent.viennot@inria.fr

Abstract

We consider the fully distributed Video-on-Demand problem, where n nodes called boxes *store a large set of videos and collaborate to serve simultaneously n videos or less between them. It is said to be* scalable *when $\Omega(n)$ videos can be distributively stored under the condition that any sequence of demands for these videos can always be satisfied. Our main result consists in establishing a threshold on the average upload bandwidth of a box, above which the system becomes scalable. We are thus interested in the normalized upload capacity $u = \frac{upload\ bandwidth}{video\ bitrate}$ of a box. The number m of distinct videos stored in the system is called its catalog size.*

We show an upload capacity threshold of 1 for scalability in a homogeneous system, where all boxes have the same upload capacity. More precisely, a system with $u < 1$ has constant catalog size $m = O(1)$ (every box must store some data of every video). On the other hand, for $u > 1$, an homogeneous system where all boxes have same upload capacity at least u admits a static allocation of $m = \Omega(n)$ videos into the boxes such that any adversarial sequence of video demands can be satisfied. Moreover, such an allocation can be obtained randomly with high probability. This result is generalized to a system of boxes that have heterogeneous upload capacities under some balancing conditions.

*Research supported by CRC "MARDI II" INRIA – Orange Labs and by ANR project "ALADDIN".

1. Introduction

The quest for scalability has yielded a tremendous amount of work in the field of distributed systems in the last decade. One of its most recent developments is the peer-to-peer model, where small capacity entities collaborate to form a system whose overall capacity grows proportionally to its size. In this paper, we address the specific problem of *fully distributed video-on-demand*: it consists in using a set of n entities, called *boxes*, with storage and networking capabilities. These boxes are used to manage a catalog of videos and to play them. When the user of a box demands a video, the playback of the video should begin after a short *start-up delay*, even though the video may be stored on other boxes. We want the system to be *doubly scalable*, with respect to the requests and catalog size. The request scalability means that the systems must be able to handle up to n simultaneous requests. The catalog scalability means that the catalog size (i.e. the number of distinct videos stored) must be $\Omega(n)$. Since the average storage capacity of a box is considered to be constant, this is the best that can be achieved.

This fully distributed problem that is considered here is mainly motivated by the existence of *set-top boxes* placed directly in user homes by Internet service providers. As these boxes may combine both storage and networking capacities, and are usually always powered on, they become an interesting target for building a low cost distributed video-on-demand system that would be an alternative to more centralized systems. In such a setting, one may expect that boxes are *homogeneous* (i.e. have identical capacities). However, we can extend our results when considering

heterogeneous capacities. The model that we propose encompasses then various architectures such as a peer-assisted server or a distributed server serving purely client boxes (i.e. with no upload capacity).

Historically, first peer-to-peer systems were devoted to collaborative storage (see, e.g., [9, 10, 19]). The academic community has then proposed numerous distributed solutions to index the contents stored in a such a system. Most prominently, one can mention the numerous *distributed hash table* proposals (see, e.g., [16, 18, 20, 21]). Extreme attention has also been paid to *content distribution*. There now exists efficient schemes for single file distribution, BitTorrent being one of the most established [7]. However, those schemes cannot be directly used for content streaming since the file is downloaded in random order, incurring a very long start-up delay [17]. Several proposals were made to cooperatively distribute a *live* stream of data (see, e.g., [4, 8, 11, 15, 23, 24]). The main difficulty is then to obtain low delays and a balanced forwarding load. Note that live streaming solutions cannot be used either for video on demand since live streaming users play the same portion of the stream, while viewers of a video may play various parts of the video stream.

More recently, the problem of collaborative *video-on-demand* streaming has been addressed. The main stream of work deals with *peer-assisted* video-on-demand, where the system relies on a server (or a server farm) for storing the whole catalog. The main problem investigated concerns the collaborative distribution of a single video [2, 6, 5, 11, 14, 12, 13, 17]. The population of users interested in the same data is often called a *swarm*, and the process of exchanging data between them is called *swarming*. All these solutions thus mainly concern swarming and rely on a server for *sourcing*, i.e. for distributing newly requested data. However, in a fully distributed system, the boxes themselves are used as sources and a compromise must be found between sourcing and swarming in bandwidth utilization.

1.1. Model

The upload bandwidth of a box is the data transfer rate from this box to the others. It is generally assumed to be the main network bottleneck, so we consider that download bandwidth is not limited in our model. This assumption is especially true for DSLs connections, which have a bandwidth ratio of at least 4 to 1 of download to upload. Let u denote the average upload bandwidth. We assume that all videos have the same average stream rate, which we normalize to 1, so $u = 1$ corresponds to an upload capacity equal to the video stream rate.

For convenience, we consider a discrete round-based model, where the time unit is the time necessary for a box to establish a connection and start data transfer. New requests may arrive within any round, and boxes may establish new connections in the following rounds. The start-up delay is the maximum number of rounds elapsed between arrival and the beginning of the playback of the video. A constant number of rounds only is allowed so that start-up delay remains constant. Furthermore, we suppose that the number of requests for a given video increases at most exponentially with time: if $f(t)$ denotes the size of a *swarm*, i.e. the population of boxes viewing the same video, then we assume $f(t+i) \leq \lceil \max\{f(t), 1\}\, \mu^i \rceil$ for some $\mu > 1$. We call μ the maximal *swarm growth*: the size of a swarm increases by a factor at most μ at each time round.

We assume that all videos have same duration T, which is a reasonable approximation if the videos are feature-length films. Therefore all videos have the same size. We suppose that the average storage capacity of a box is d videos. In addition to this storage space dedicated to the catalog, a box stores the video it is playing, as data arrives, in a cache called *playback cache*. More precisely, this cache contains all the data most recently viewed up to a video file size. If a box plays videos one after another, the cache then contains the end of the previous video and the beginning of the current one.

To allow the download of a video from multiple sources, we assume that each video is encoded into c different streams called *stripes*, whose combination gives the initial stream. For scalability reasons, c is assumed to be very low, i.e. constant or poly-logarithmic. A simple encoding into c equal rate stripes consists in splitting the video file into packets. Stripe i is then made of the packets with number equal to i modulo c. When the upload capacity u_b of box b is not a multiple of $\frac{1}{c}$, it can only upload $\lfloor u_b c \rfloor$ stripes.

Finally, we call (n, u, d)-*video system* a set of n collaborating boxes with average upload capacity u and average storage capacity d. Such a system is *homogeneous* if all boxes have same upload capacity and same storage capacity, i.e. for all b, $u_b = u$ and $d_b = d$. It is *proportionally heterogeneous* if $\frac{u_b}{d_b} = \frac{u}{d}$ for every box b. We say that an (n, u, d)-video system *achieves* catalog size m if it is possible to store m distinct videos on the boxes so that any sequence of requests of at most one video per box can be satisfied as long as the maximal swarm growth μ is respected.

An *allocation* is the process of storing stripe replicas into boxes statically. (The only data that changes frequently in a box is the data stored in its cache). We define the *minimal chunk size* ℓ as the minimal amount of data of a given video stored in a box. When splitting videos into c stripes, we get $\ell = \frac{1}{c}$. The scalability condition for c translates into $\ell = \Omega(1)$, i.e. a video cannot be split into infinitely small pieces as n increases. (All parameters of the model are summarized in Table 1).

n	Number of boxes in the system.
m	Number of distinct videos stored in the system (catalog size).
d_b	Storage capacity of box b (in number of videos).
d	Average storage capacity of boxes.
k	Number of duplicates copies of a video with random allocation ($k \approx dn/m$)
u_b	Upload capacity of box b (in number of full video streams).
u	Average upload capacity of boxes.
c	Number of stripes per video (a video can be viewed by downloading its c stripes simultaneously).
μ	Swarm growth bound: if a swarm has size p at round t, its size is less than μp at round $t+1$.
ℓ	Minimal chunk size: a box stores at least ℓ of a given video ($\ell = 1/c$ when storing complete stripes).

Table 1. Key parameters

1.2. Related work

It has been proposed that boxes cache the last videos they have played [1, 13, 14]. However, these solutions still assume that a centralized server stores the whole catalog. Such solutions do not tackle the problem of competition between sourcing and swarming that we encounter in a fully distributed system.

To the best of our knowledge, Suh *et al.* [22] made the first attempt to investigate the possibility of a server-free video-on-demand architecture. The primary copies of the catalog are replicated on set-top boxes that are used for video-on-demand. However, the focus is mainly on sourcing: the videos are sufficiently replicated so that all requests are satisfied through these original copies. Additionally, the scalability of the catalog is not investigated at all. Indeed, the system is tailored for boxes with upload bandwidth lower than playback rate and a constant catalog size (each box stores a constant portion of each video).

Following this seminal work, we studied in a preliminary work [3] the conditions for catalog scalability under the assumption that requests concern pairwise distinct videos and that the system is homogeneous. The focus is thus still on sourcing. A distributed video-on-demand system is sketched relying on existing single video distribution algorithm for handling multiply requested videos. Sourcing and swarming are thus treated separately and the resulting bound misses the interplay between the two competing algorithms.

1.3. Our results

In this work, we address the full problem when mixing both sourcing and swarming and exhibit a tight threshold. Additionally, we consider heterogeneous systems where boxes may have different capacities one from another.

First note that $u \geq 1$ is a natural requirement if all boxes may play videos at the same time. Suppose $u < 1$. As minimal chunk size is ℓ, each box b stores data of at most $\frac{d_b}{\ell}$ videos. Set $d_{\max} = \max_b\{d_b\}$. If $m > \frac{d_{\max}}{\ell}$, then for each box b, there always exists a video v not possessed by b, i.e. b stores no data at all from v. Consider a sequence of requests where each box always plays a video it does not possess. The aggregated download rate then becomes n whereas the aggregated upload rate is $un < n$ which is not sufficient. As a consequence we must have $m \leq \frac{d_{\max}}{\ell}$. Catalog size is thus constant as long as $d_{\max} = O(1)$ and $\ell = \Omega(1)$.

In contrast, our main result states that it is indeed possible to have a linear catalog size as soon as $u > 1$. In details, we propose in Section 2.1 a random video allocation scheme where each video is split into c stripes of rate $\frac{1}{c}$. Each stripe is replicated a constant number $k = O(\log_u d')$ of times where $d' = \max\{d, u, \exp(1)\}$. Replicas and stored statically on randomly chosen boxes, yielding a catalog size of $\frac{dn}{k}$. We show that in the case of a homogeneous system the graph linking each stripe to the boxes storing it at a given time has some expander property with high probability. A min-cut max-flow argument is then used in Sections 2.2 and 2.3 to prove that any sequence of demands can always be satisfied as long as $u > 1$. Theorem 1 in Section 3 formally states that catalog size $\Omega\left(\frac{(u-1)^2 \log \frac{u+1}{2}}{u^3} \frac{1}{\mu^2} \frac{dn}{\log d'}\right)$ (linear in n since μ, d' and u are constant) can be achieved under these conditions. Finally, the result is generalized to the case of a heterogeneous system in Section 4. A solution is proposed to overcome the difficulty caused by the boxes having upload less than 1. It consists in relaying the demands of these boxes through the boxes having enough upload bandwidth.

Our approach consists in applying together maximum flow arguments and the probabilist method to show that a valid allocation of videos can be found with high probability. For that purpose we have to show that all the graphs of "who give what" encountered in the infinite sequence of requests have some expander property. This is possible through the combination of algorithmic arguments concerning restrictions on how requests are made and probabilistic arguments on how videos are allocated. This result does not yield directly a practical distributed algorithm. However, it

shows that scalable video on demand is theoretically feasible for $u > 1$. Moreover the preloading scheme we propose mimics in some manner a classical peer-to-peer balancing strategies such as encountered in Splitstream [4] or in BitTorrent [7]: nodes first download pairwise distinct chunks of data so that they can exchange their chunks afterwards. (In our setting, a chunk corresponds to one time round of a stripe). Our analysis could give new insights in such contexts.

2. Preliminaries

We now present the basic requirements for achieving a given sequence of requests by considering the graph linking each request to the boxes that possess the corresponding data. We first briefly present how videos can be randomly placed in the system when using c stripes of rate $\frac{1}{c}$ per video and k replicas per stripe.

2.1. Random allocation

Random allocation consists in storing $k \geq 1$ replicas of each stripe into k boxes chosen randomly, either independently or according to a random permutation. For the sake of simplicity, we assume that $k = dn/m$ is an integer. A *random independent allocation* consists in selecting independently for each stripe replica a box with probability proportional to its storage capacity. (The process is stopped as soon as a replica falls in a completely filled-up box). Alternatively, a *random permutation allocation* consists in copying each stripe into k boxes such that each box contains exactly dc stripe replicas. We model this through a random permutation π of the $kmc = dnc$ stripe replicas into the dnc storage slots of the n boxes together: replica i is stored in slot $\pi(i)$ (the d_1c first slots fall into the first box, the d_2c next slots into the second box, and so on). The highest catalog size is obtained for the smallest possible value of k.

We call *random allocation* the process consisting in encoding each video into c stripes and storing k replicas of each stripes randomly on boxes, either according to a random permutation, or a random independent allocation.

2.2. Connection matching

We model the problem of finding connections for downloading the video stripes at a given time as a maximum flow problem. We suppose that mc distinct stripes are stored in the system according to a random allocation as described above. Let W denote the set of boxes and consider the set Y of requested stripes at time t. We can write $Y = \{(s_1, t_1, b_1), \ldots, (s_p, t_p, b_p)\}$ with $p \leq nc$, where (s_i, t_i, b_i) corresponds to a request for stripe s_i made by box b_i at time $t_i \leq t$. We let $S(Y) = \{s_1, \ldots, s_p\}$ denote the *multiset* of all requested stripes (some stripes may be requested multiple times). A *connection matching* is a matching of requests against boxes possessing the necessary video data so that each box b has degree at most u_bc. Wiring connections according to such a matching allows to satisfy requests at round $t + 1$ as each stripe has rate $\frac{1}{c}$. Finding such a connection matching is modeled as a maximal flow computation in the following bipartite graph.

We define G as the bipartite graph from Y to W where each request x is linked to each box possessing data necessary for x at $t+1$. More precisely, each $x = (s_i, t_i, b_i) \in Y$ corresponds to a request for stripe s_i made at time t_i. It thus requires data at position $t - t_i$ in the stripe. In addition to boxes storing the stripe according to the random allocation, this data is also possessed by the boxes b that have requested (s_j, t_j, b_j) with $s_j = s_i$, $b_j = b$ and $t - T \leq t_j < t_i$ since each box caches the last video data played in a time window of length T. (We assume that the system has fulfilled all requests up to time t). A connection matching is indeed a subset of links $C \subseteq E(G)$ inducing a sub-graph where each stripe request has degree 1 and each box b has degree at most u_bc.

2.3. Maximum flow feasibility

We can characterize the existence of a connection matching as follows. Let $B(x)$ denote the neighbors of a request $x \in Y$ in G, i.e. the set of boxes possessing data for x at time t. More generally, for a subset $X \subseteq Y$ of requests, let $B(X) = \cup_{x \in X} B(x)$ denote the set of boxes possessing data for any request $x \in X$. For a set $E \subseteq W$ of boxes, let $U_E = \sum_{b \in E} u_b$ denote its overall capacity. We can then state the following lemma (which is a simple generalisation of Hall's theorem).

Lemma 1 (Min-cut max-flow) *A connection matching for satisfying requests at the next time round exists iff for all $X \subseteq Y$, $U_{B(X)} \geq \frac{|X|}{c}$ where $U_{B(X)} = \sum_{b \in B(X)} u_b$.*

In the homogeneous case, this states that G must be a $\frac{1}{uc}$-expander, i.e. for all $X \subseteq Y$, $|B(X)| \geq \frac{1}{uc}|X|$.

Proof. The condition is clearly necessary as the sum of the degrees of the boxes in $B(X)$ in a connection matching is at least $|X|$, and the overall upload capacity of $B(X)$ must be at least $\frac{|X|}{c}$. On the other hand, consider the flow network obtained by adding a source node a and a sink node z as follows. An edge with capacity u_b is added from a to each box node $b \in W$. An edge of capacity $\frac{1}{c}$ is added from each box b to each request $x \in Y$ such that $b \in B(x)$. An edge with capacity $\frac{1}{c}$ is added from each request in Y to z. Considering the cut $A = \{a\} \cup W \cup Y$, $Z = \{z\}$, we see that the maximal flow is at most $\frac{|Y|}{c}$. We show that it

is indeed equal to this value, i.e. there exists a connection matching. Consider a cut A, Z with $a \in A$ and $z \in Z$. Let $X \subseteq Z$ denote the set of all request nodes x such that $x \in Z$ and $B(x) \cap A = \emptyset$ and assume $U_{B(X)} \geq \frac{|X|}{c}$. For all request $x \in A$, edge xz crosses the cut. For all request $x \in Z \setminus X$, there exist $b \in B(x) \cap A$ and edge bx crosses the cut. For each box $b \in B(X)$, edge ab crosses the cut and its weight is u_b. The capacity of the cut is thus at least $\frac{|Y-X|}{c} + \sum_{b \in B(X)} u_b \geq \frac{|Y|}{c}$. The classical min-cut max-flow theorem allows to conclude. □

We call *request obstruction* a subset X of requests such that $U_{B(X)} < \frac{|X|}{c}$. We are indeed interested in the multiset $M(X)$ of stripes requested in X. We extend this notion to any multiset of stripes: we call *obstruction* a multiset σ of stripes such that there exists a sequence of video demands that has been satisfied up to time t and where a subset X of requests at time t satisfies $M(X) = \sigma$ and $U_{B(X)} < \frac{|X|}{c}$. Clearly, Lemma 1 implies that any sequence of demands can always be satisfied iff there exists no obstruction.

We can then bound the probability that a given random allocation can be defeated as follows. We denote by N_k the random variable defined as the number of obstructions (among all possible subsets of at most nc stripes) in a permutation allocation chosen uniformly at random from the set A_k of all possible random allocations for a given k (and a given type of allocation: permutation or independent). Let $\mathcal{O}$ be the set of multisets of stripes with cardinality at most nc. For some allocation a and a multiset of stripes σ, we denote by $I(a, \sigma)$ the indicator variable that is equal to 1 if σ is an obstruction and 0 otherwise. Using the first moment method, we can bound $P(N_k > 0)$ (the probability that a random allocation admits at least one obstruction):

$$\begin{aligned} P(N_k > 0) &\leq E(N_k) \\ &= \frac{\sum_{a \in A_k} \sum_{\sigma \in \mathcal{O}} I(a, \sigma)}{|A_k|} \\ &= \sum_{\sigma \in \mathcal{O}} \frac{\sum_{a \in A_k} I(a, \sigma)}{|A_k|} = \sum_{\sigma \in \mathcal{O}} P(\sigma) \quad (1) \end{aligned}$$

where $P(\sigma)$ is the probability for some multiset of stripes σ to be an obstruction in a randomly chosen allocation. As we shall see, for sufficiently high values of u and k, the expectation of the number of obstructions is bounded by $O\left(\frac{1}{n^\lambda}\right)$ for some positive λ and then with high probability the number of obstructions in a randomly chosen allocation is zero.

3. Homogeneous Video Systems

We can now state our main theorem in the homogeneous case. We extend it to the proportionally heterogeneous case in the next section.

Theorem 1 *Given $u > 1$, consider a homogeneous (n, u, d)-video system. With high probability, a random permutation allocation with $c > \frac{2\mu^2-1}{u-1}$ and $k \geq 5\nu^{-1} \frac{\log d'}{\log u'}$ for $\nu = \frac{1}{c+2\mu^2-1} - \frac{1}{uc}$, $u' = \frac{1}{c}\lfloor uc \rfloor$ and $d' = \max\{d, u, \exp(1)\}$ allows to successfully satisfy any sequence of requests with maximal swarm growth μ. As a consequence, the system can achieve catalog size $\Omega\left(\frac{(u-1)^2 \log \frac{u+1}{2}}{u^3} \frac{1}{\mu^2} \frac{dn}{\log d'}\right)$.*

The result holds also for a random independent allocation with same bounds for c and k (in both cases, we rely on the bound given in Lemma 3 below). However, in the random independent case, box storage loads may be unbalanced. To avoid to exceed the capacity of any box with high probability, we have to additionally require $c = \Omega(\log n)$. For large n, we have $u' \geq \frac{u}{2}$, $\nu^{-1} \sim \frac{uc}{u-1}$ and $k = O\left(\frac{u}{u-1} \frac{\log d'}{\log \frac{u}{2}} \log n\right)$ is then sufficient to obtain catalog size $\Omega\left(\frac{(u-1)\log \frac{u}{2}}{u} \frac{d}{\log d'} \frac{n}{\log n}\right)$.

Relying on Section 2, we use a scheme where boxes store and upload full stripes. We assume in the sequel a choice of c satisfying $c > \frac{2\mu^2-1}{u-1}$, or equivalently $u > 1 + \frac{2\mu^2-1}{c}$. As a box can upload only $\lfloor uc \rfloor$ stripes, its effective upload capacity is $u' = \frac{\lfloor uc \rfloor}{c} \geq u - \frac{1}{c} > 1 + \frac{2(\mu^2-1)}{c}$. The proof mainly relies on two arguments. First, a request strategy is proposed to cope with highly demanded videos. Second, a randomized argument bounds the probability that a request for various videos cannot be satisfied by the boxes storing them according to the random allocation scheme.

Indeed, we use the following preloading strategy for requesting stripes. Consider a box b where the user demands a video v during the interval $[t-1, t[$. A *preloading* request (s, t, b) for one stripe s of v is first issued at time t. Then $c-1$ *postponed* requests are made for the $c-1$ remaining stripes of v at time $t+1$. The start-up delay for playing a video is thus 3 time rounds. (Note that the downloading of a video now lasts during one more round time. For the sake of simplicity, we assume that this additional time round is counted in the duration T). By convention, we say that the box *enters* the swarm of v at time t (when it begins to download some data). To balance preloading requests, we use a counter for each video v to give successive numbers to boxes entering the swarm of v. The pth box then preloads stripe number p modulo c so that all stripes of a video are equally preloaded. We will see that this strategy allows to manage a large swarm growth μ as long as the number of stripes is sufficiently large ($c > \frac{2\mu^2-1}{u-1}$).

On the other hand, we rely on Equation 1 that consists in bounding $P(\sigma)$ for every multiset σ of size at most nc. We obtain an upper bound of $P(\sigma)$ that depends only on the number of stripes in σ and the number of pairwise distinct stripes among them. For that purpose, we estimate the num-

ber of boxes that can serve the requests made during a time interval $[t-T,t]$ thanks to the following lemma.

Lemma 2 *At time t, consider any subset X of stripe requests made in $[t-T,t]$. Let $i=|X|$ denote the size of X and let i_1 denote the number of pairwise distinct stripes requested in X. Then the set $B(X)$ of boxes that can serve requests in X satisfies $|B(X)| \geq \frac{i-(c+2\mu^2-1)i_1}{c+2(\mu^2-1)}$.*

Proof. We consider the requests in X video by video. Suppose that some video v has some stripes requested in X. Let $X(v) \subseteq X$ denote the requests for stripes of v. Let $i(v)$ denote the number of requests in $X(v)$ and $i_1(v)$ denote the number of distinct stripes requested in $X(v)$. (We thus have $i=\sum_v i(v)$ and $i_1=\sum_v i_1(v)$). Let t_v denote the latest time a request of $X(v)$ was made. We define $X^{pre}_{t'}(v)$ (resp. $X^{post}_{t'}(v)$) as the preloading (resp. postponed) requests in $X(v)$ made at some time t'. We mainly consider $X^{pre}_{t_v-1}(v)$, $X^{post}_{t_v}(v)$, and $X^{pre}_{t_v}(v)$. On the other hand, the definition of t_v implies $X^{post}_{t_v}(v) \cup X^{pre}_{t_v}(v) \neq \emptyset$. Let $X'(v)=X(v)-X^{pre}_{t_v-1}(v)-X^{post}_{t_v}(v)-X^{pre}_{t_v}(v)$ denote the remaining requests in $X(v)$.

Let $x_0(v)$, $x_1(v)$ and $x_2(v)$ the number of boxes that have entered the swarm of v before t_v-1, at t_v-1 and at t_v respectively. To derive a lower bound on $|B(X_{t_v}(v)^{post} \cup X^{pre}_{t_v}(v))|$, we estimate separately the contribution to this bound of boxes that entered the swarm at time t_v-1 and before t_v-1.

Note that our preloading strategy implies that requests in $X^{pre}_{t_v-1}(v) \cup X^{post}_{t_v}(v)$ (resp. $X^{pre}_{t_v}(v)$) are made by boxes that entered the swarm of v at t_v-1 (resp. t_v). As each box makes at most one preload request, we have $x_1(v) \geq |X^{pre}_{t_v-1}(v)|$ and $x_2(v) \geq |X^{pre}_{t_v}(v)|$. Let $i_2(v)$ denote the number of distinct stripes requested in $X^{post}_{t_v}(v)$. (We have $i_2(v) \leq i_1(v)$). There are thus at least $\frac{|X^{post}_{t_v}(v)|}{i_2(v)}$ boxes making these requests as a box requests a given stripe only once. We thus have $x_1(v) \geq \frac{|X^{post}_{t_v}(v)|}{i_2(v)}$. Thanks to our preloading strategy, $\left\lfloor \frac{x_1(v)}{c} \right\rfloor \geq \left\lfloor \frac{|X^{post}_{t_v}(v)|}{c \cdot i_2(v)} \right\rfloor \geq \frac{|X^{post}_{t_v}(v)|}{c \cdot i_2(v)} - 1$ boxes preload any stripe requested in $X^{post}_{t_v}(v)$. As boxes preloading distinct stripes are distinct, we deduce the contribution of boxes entering the swarm at t_v-1 (1): $|B(X^{post}_{t_v}(v))| \geq i_2(v)\left(\frac{|X^{post}_{t_v}(v)|}{c \cdot i_2(v)} - 1\right) \geq \frac{|X^{post}_{t_v}(v)|}{c} - i_2(v) \geq \frac{|X^{post}_{t_v}(v)|}{c} - i_1(v)$.

On the other hand, requests in $X^{pre}_{t_v}$ can be served by boxes that entered the swarm of v before t_v-1. The number $x_0(v)$ of such boxes is at least $\frac{|X'(v)|}{i_1(v)}$. This gives the following bound (2): $|B(X^{post}_{t_v}(v) \cup X^{pre}_{t_v}(v))| \geq \frac{|X'(v)|}{i_1(v)}$. Moreover, the bound μ on swarm growth implies $x_0(v)+x_1(v)+x_2(v) \leq \lceil \mu^2 x_0(v) \rceil \leq \mu^2 x_0(v)+1$. We thus get an other bound (3): $|B(X^{post}_{t_v} \cup X^{pre}_{t_v})| \geq x_0(v) \geq \frac{x_1(v)+x_2(v)-1}{\mu^2-1} \geq \frac{|X^{pre}_{t_v-1}|+|X^{pre}_{t_v}|-1}{\mu^2-1}$. Note that the boxes considered in (1) are distinct from those considered in (2) and (3) since they entered the swarm later. The boxes previously considered in (1) and (2) are currently in the swarm of v. The boxes considered in (3) have entered the swarm of v at t_v-T or after. (Note that they effectively still have the data at position $t-t_v$ in their cache as $t_v-T+(t-t_v) \geq t-T$.)

We can now consider all videos that are requested in X. The set of boxes considered in bounds (1) and (2) are disjoint since these boxes request distinct videos. The boxes considered in bound (3) may concern at most two videos as a box enters at most two swarms during a period T (when one video is played after another). We thus get the following lower bound: $|B(X)| \geq \sum_v \frac{|X^{post}_{t_v}|}{c} - i_1(v) + \max\left\{\frac{|X'(v)|}{i_1(v)}, \frac{|X^{pre}_{t_v-1}|+|X^{pre}_{t_v}|-1}{2(\mu^2-1)}\right\}$. Using $\max(A,B) \geq \frac{i_1(v)A+2(\mu^2-1)B}{i_1(v)+2(\mu^2-1)}$ for any A,B, we obtain: $|B(X)| \geq \sum_v \frac{|X^{post}_{t_v}|}{c} - i_1(v) + \frac{|X'(v)|+|X^{pre}_{t_v-1}|+|X^{pre}_{t_v}|-1}{i_1(v)+2(\mu^2-1)}$. Using $\frac{1}{c} \geq \frac{1}{c+2(\mu^2-1)}$ and $\frac{1}{i_1(v)+2(\mu^2-1)} \geq \frac{1}{c+2(\mu^2-1)}$, we get: $|B(X)| \geq \sum_v \frac{|X^{post}_{t_v}|}{c+2(\mu^2-1)} - i_1(v) + \frac{|X'(v)|+|X^{pre}_{t_v-1}|+|X^{pre}_{t_v}|-1}{c+2(\mu^2-1)}$. As $i(v)=|X(v)|=|X'(v)|+|X^{pre}_{t_v-1}|+|X^{post}_{t_v}|+|X^{pre}_{t_v}|$, we get $|B(X)| \geq \sum_v \frac{i(v)-(c+2(\mu^2-1))i_1(v)-1}{c+2(\mu^2-1)} \geq \frac{i-(c+2\mu^2-1)i_1}{c+2(\mu^2-1)}$. □

The following lemma bounds from above the probability that a set of pairwise distinct stripes are allocated to the same set of p boxes in a random permutation allocation. It is also trivially satisfied if stripe replicas are placed according to a random independent allocation rather than a random permutation.

Lemma 3 *Consider a random permutation allocation of $kmc=dnc$ stripe replicas into the dnc memory slots of n boxes. The probability that ki given replicas fall into p given boxes with $dpc \geq ki$ is less than $\left(\frac{p}{n}\right)^{ki}$.*

Proof. Drawing uniformly at random a permutation of the $kmc=dnc$ stripe replicas amounts to choose uniformly at random a slot for the first replica, then a slot for the second among the remaining slots and so on. The ki replicas are ordered. Let E_a denotes the event that the a^{th} replica falls into one of the pdc slots of the p boxes. $P(\cap_{a \leq ki} E_a) = P(E_1).P(E_2|E_1)...P(E_a|E_1 \cap E_2 ... \cap E_{a-1})... = \frac{pdc}{ndc}.\frac{pdc-1}{ndc-1}... \frac{pdc-a+1}{ndc-a+1}... \leq \left(\frac{p}{n}\right)^{ki}$ (since $\frac{pdc-i}{ndc-i} \leq \frac{pdc}{ndc}$ for $p \leq n$). □

We can now bound the probability that a multiset of at most nc stripes is an obstruction.

Lemma 4 *Let σ be a multiset of stripes of size $i \leq nc$. Let i_1 be the number of pairwise distinct stripes in σ. The probability $P(\sigma)$ of σ to be an obstruction is at most $P(\sigma) \leq \left(\frac{u'nce}{i}\right)^i \left(\frac{i}{u'cn}\right)^{ki_1}$. In addition, $P(\sigma) = 0$ when $i_1 \leq \nu i$.*

Note that the assumption on c implies $uc > c + 2\mu^2 - 1$. We thus have $0 < \nu < 1$ as $\nu = \frac{1}{c+2\mu^2-1} - \frac{1}{uc}$.

Proof. Consider a sequence of video demands where $\sigma = M(X)$ for some subset X of requests at time t. We then consider two cases :

1. Either $i_1 \leq i\nu$, implying $i - (c + 2\mu^2 - 1)i_1 \geq i\frac{c+2\mu^2-1}{uc} \geq i\frac{(c+2\mu^2-1)-1}{uc-1}$ as $uc > c + 2\mu^2 - 1$. As a box uploads full stripes, its effective upload capacity is $u' = \frac{1}{c}\lfloor uc \rfloor \geq \frac{1}{c}(uc - 1)$. According to Lemma 2, we thus have $U_{B(X)} \geq \frac{i}{c}\frac{(c+2\mu^2-1)-1}{c+2(\mu^2-1)} = \frac{i}{c}$. X cannot be an obstruction set in this case and then $P(\sigma) = 0$.

2. or $i_1 > \nu i$. Any set E of boxes with $U_E \leq \frac{i}{c}$ satisfies $|E| \leq \frac{i}{u'c}$ and is included in some subset of size $\left\lfloor \frac{i}{u'c} \right\rfloor$. The number of such sets is at most $\binom{n}{\lfloor i/(u'c) \rfloor} \leq \binom{u'nc}{i}$. We use this very coarse bound so that the lemma will remain valid in the heterogeneous case. By considering all such sets and using Lemma 3 for the replicas of the i_1 distinct stripes of s, we obtain $P(U_{B(X)} < \frac{i}{c}) \leq \binom{nu'c}{i}\left(\frac{i/(u'c)}{n}\right)^{ki_1} \leq \left(\frac{u'nce}{i}\right)^i \left(\frac{i}{u'nc}\right)^{ki_1}$. The last inequality is obtained by using the standard upper bound of the binomial coefficient $\binom{b}{a} \leq \left(\frac{be}{a}\right)^a$.

□

Proof.[of Theorem 1] From Equation (1), and since the probability that a multiset σ is an obstruction depends only on $i = |\sigma|$ the total number of stripes and i_1 the number of distinct ones, and using Lemma 4, one can write $P(N_k > 0) \leq \sum_{i=1}^{nc} \sum_{i_1=\lceil \nu i \rceil}^{\min\{i,mc\}} M(i, i_1)\left(\frac{u'nce}{i}\right)^i \left(\frac{i}{u'nc}\right)^{ki_1}$ where $M(i, i_1)$ is the number of multisets of cardinality i taken from sets of stripes of cardinality i_1 and $\nu = \frac{1}{c+2\mu^2-1} - \frac{1}{uc}$ as in Lemma 4. Note that i_1 is bounded by the catalog size: $i_1 \leq mc$.

We have $M(i, i_1) = \binom{mc}{i_1}\binom{i-1}{i_1-1} \leq \binom{nc+mc}{i}2^i \leq \left(\frac{4d'nce}{i}\right)^i$ where $d' = \max\{d, u, e\} \geq 1$ since $nc + mc \leq nc + \frac{dnc}{k} \leq 2d'nc$ as $k \geq 1$ (we use $\binom{b}{a} \leq \binom{b+c}{a+c}$ for $c \geq 0$ and $\binom{b}{a} \leq 2^b$). Notice also that $\left(\frac{i}{u'nc}\right)^{ki_1} \leq \left(\frac{i}{u'nc}\right)^{\nu ki}$. The probability $P(N_k > 0)$ is then at most $\sum_{i=1}^{nc}(1 - \nu)i\phi(i)$ where $\phi(i) = \left(\frac{i}{u'nc}\right)^{\kappa i}\delta^i$ with $\kappa = \nu k - 2$ and $\delta = \frac{4d'e^2}{u'}$. We finally get $P(N_k > 0) \leq (1 - \nu)nc\sum_{i=1}^{nc}\phi(i)$.

It is easy to check that as a function of i the term $\phi(i)$ of the sum decreases from $\phi(1)$, reaches a minimum at $\phi(i^\star)$ with $i^\star = \frac{u'nc}{e\delta^{1/\kappa}}$ then increases to $\phi(nc)$. Using this fact, we bound $P(N_k > 0)$ by considering separately the sum for $i \leq i^\star$ and $i > i^\star$ and by replacing each term with the maximum term on its side. On one hand, we have $(1-\nu')nc\sum_{i=u'c}^{\lfloor i^\star \rfloor}\phi(i) \leq (1-\nu')nc.nc.\phi(1) = O(\frac{1}{n^{\kappa-2}})$. On the other hand, we have $(1-\nu)nc\sum_{\lfloor i^\star \rfloor+1}^{nc}\phi(i) \leq (1-\nu)nc.nc.\phi(nc) = O(n^2(u'^{-\kappa}\delta)^{nc})$ as $\phi(nc) = \left(\frac{\delta}{u'^\kappa}\right)^{nc}$. Finally, we get $P(N_k > 0) = O(\frac{1}{n^{\kappa-2}}) + O(n^2(u'^{-\kappa}\delta)^{nc})$. The first term is $O(\frac{1}{n})$ for $\kappa \geq 3$, i.e. $k \geq 5\nu^{-1}$. The second term vanishes exponentially for $u'^{-\kappa}\delta < 1$, i.e. $\kappa > \log_{u'}\delta$ or equivalently $k > \nu^{-1}\left(2 + \log_{u'}\frac{4d'e^2}{u'}\right) = \nu^{-1}\log_{u'}(4e^2d'u')$ on the number of replicas per stripe.

Finally, the probability that an obstruction exists is $O(\frac{1}{n})$ for $k \geq \nu^{-1}\max\left\{5, \log_{u'}(e^4d'u')\right\}$. In particular, $k \geq 5\nu^{-1}\log_{u'}d'$ is sufficient since $\log_{u'}d' \geq \max\{1, \log_{u'}e\}$. For $k \geq \left\lceil 5\nu^{-1}\log_{u'}d' \right\rceil$, the system thus achieves catalog size $m = \Omega\left(\frac{\nu dn}{\log_{u'}d'}\right)$ with high probability.

For $c = \left\lceil 2 \cdot \frac{2\mu^2-1}{u-1} \right\rceil$, we have $c \geq \frac{2}{u-1}$ as $\mu \geq 1$, and $u' \geq u - \frac{1}{c} \geq \frac{u+1}{2}$. Using $(u-1)c \geq 2(2\mu^2 - 1)$ and $c < (2\mu^2 - 1)(\frac{2}{u-1} + 1)$, we then obtain $\nu^{-1} \leq 8\mu^2\frac{u^3}{(u-1)^2}$, $k = O\left(\frac{\mu^2u^3}{(u-1)^2}\frac{\log d'}{\log\frac{u+1}{2}}\right)$ and $m = \Omega\left(\frac{(u-1)^2\log\frac{u+1}{2}}{u^3\mu^2}\frac{dn}{\log d'}\right)$. □

4. Balanced Heterogeneous Video Systems

The main difficulty in an heterogeneous system occurs when many boxes with upload less than 1 play the same video since their capacity is not sufficient to replicate the data between themselves. Let $u^* > 1$ be an *upload threshold* under which a box is considered to have deficient upload, i.e. less than u^*. We introduce the *upload deficit* with respect to u^* as the quantity $\Delta(u^*) = \sum_{b|u_b<u^*} u^* - u_b$ which is the overall bandwidth missing to *poor* boxes, i.e. boxes with capacity less than u^*. A box b is said to be *rich* when $u_b \geq u^*$. In this section, we are interested in heterogeneous systems satisfying:

$$u > 1 + \frac{\Delta(1)}{n}$$

One can see that $u \geq 1 + \frac{\Delta(1)}{n}$ is an intuitive lower bound for scalability by considering the following request scenario. Suppose that all rich boxes watch a video they do not possess and poor boxes start to play the same video v at maximum growth rate. Either v is widely replicated among the poor boxes, or rich boxes will have to send the data to the poor boxes that cannot upload within themselves. In

the latter case, this requires an additional overall upload of roughly $\Delta(1)$.

We say that a system can be u^*-*upload-compensated* if for any poor box b we can reserve an upload capacity $u^* - u_b + 1 - u_b$ on a rich box $r(b)$ with $u_{r(b)} \geq u^* + (u^* + 1 - 2u_b)$.Several reservations may fall in a box a as long as $u_a \geq u^* + \sum_{b|r(b)=a}(u^* + 1 - 2u_b)$. Note that this requires at least $u \geq u^* + \frac{\Delta(1)}{n}$.

Another difficulty may come from the unbalance between storage capacity and upload capacity. Indeed, it may be useless to have very high storage capacity in boxes with low upload capacity and vice versa. A system is u^*-*storage-balanced* with respect to u^* if $2 \leq \frac{d_b}{u_b}$ and $\frac{d_b}{u_b} \leq \frac{d}{u^*}$ for all b. As a particular case, a proportionally heterogeneous system, where $\frac{u_b}{d_b} = \frac{u}{d}$ for all box b, is always u^*-storage-balanced for $d \geq 2$ and $u^* \leq u$. Note that a system with $2 \leq \frac{d_b}{u_b}$ for all box b can always be considered as u^*-storage-balanced for $u^* \leq u$ by artificially reducing the storage capacity of each box to $d'_b = \tau u_b$ with $\tau = \min_b \frac{d_b}{u_b}$ at the cost of reducing the average storage capacity to τu.

We say that a video system is u^*-*balanced* if it is u^*-storage-balanced and can be u^*-upload-compensated. The maisn idea behind the requirement of compensated systems is to relay stripes for each *poor* box b (i.e. with $u_b < u^*$) through a rich box $r(b)$ with sufficient upload capacity according to the u^*-upload-compensated assumption. The strategy for making requests is then the following. A poor box b whose user demands a video during the interval $[t-1, t]$ processes as follows: at time t, it asks $r(b)$ to issue a request for its preloading stripe (selected as before), this is considered as a preloading request. At time $t+1$, $r(b)$ forwards this preloading stripe to b. This relies on the upload statically reserved on $r(b)$ and this is not considered as a request. At time $t+2$, it requests $c_b = \lfloor cu_b - 4\mu^4 \rfloor$ of the $c-1$ remaining stripes ($c_b = 0$ if $u_b \leq \frac{2\mu^4}{c}$). At time $t+3$, it asks $r(b)$ to request the $c - 1 - c_b$ remaining stripes (these are postponed requests). At time $t+3$, $r(b)$ forwards these $c - 1 - c_b$ stripes to b. (in addition to the preloading stripe). Again this relies on the upload reserved on $r(b)$ and this is not considered as requests. The strategy for a rich box a (i.e. $u_a \geq u^*$) whose user demands a video at time t remains similar except that the postponed requests are made at time $t+2$ instead of $t+1$. In both cases we say that the box enters the swarm at t (the time of the first request). From the point of view of requests, a scenario of user demands results in a sequence identical to the sequence obtained previously if we scale the time round duration by a factor of 2. For this time scale, the bound on swarm growth becomes μ^2 instead of μ.

Note that an upload bandwidth $(c - c_b)\frac{1}{c} < 1 - u_b + \frac{4\mu^4+1}{c}$ is statically allocated to b and cannot be used for answering requests. We will assume $c \geq \frac{10\mu^4}{u^*-1}$, implying $u^* \geq 1 + \frac{10\mu^4}{c}$, the reserved upload $u^* + 1 - 2u_b$ is clearly sufficient for this allocation. There still remains a reservation of $u^* - u_b - \frac{4\mu^4+1}{c} > 0$ since $u_b < 1 + \frac{4\mu^4+1}{c}$, $u^* > 1 + \frac{10\mu^4}{c}$ and $\mu > 1$. Additionally, we require that each stripe forwarded by $r(b)$ to b is also cached by $r(b)$. If storage capacity has to be used, the storage capacity of the box is reduced this is the reason behind the u^*-storage-balance condition that $d_{r(b)} \geq 2u_{r(b)}$ since the capacity of $r(b)$ is reduced by a factor at most 2 in total. Note that a poor box b caches all stripes whereas $r(b)$ caches only the stripes it forwards which include the preloading stripe of b.

We can now extend Theorem 1 to balanced heterogeneous systems.

Theorem 2 *For any fixed $u^* > 1$, consider a u^*-balanced (n, u, d)-video system. With high probability, a random permutation allocation with $c > \frac{4\mu^4}{u^*-1}$ and $k \geq 5\nu^{-1}\frac{\log d'}{\log u'}$ for $\nu = \frac{1}{c+2\mu^4-1} - \frac{1}{c+3\mu^4}$, $u' = \frac{c+3\mu^4}{c}$ and $d' = \max\{d, u^*, \exp(1)\}$ allows to successfully satisfy any sequence of requests with maximal swarm growth μ. For $c = \left\lceil \frac{10\mu^4}{u^*-1} \right\rceil$ and $u^* \leq 2$, it can achieve catalog size $\Omega\left(\frac{(u^*-1)^2 \log \frac{u^*+3}{4}}{\mu^4} \frac{dn}{\log d'}\right)$.*

The proof of this theorem follows the same steps as for Theorem 1. We claim that Lemma 2 can be generalized in this setting. The main arguments are the following. Remaining reserved upload capacities are pairwise disjoint and are disjoint from box capacities (a rich box a is considered to have upload capacity $u_a - \sum_{b|r(b)=a}(u^* + 1 - 2u_b) \geq u^*$). A stripe preloaded by a poor box b is cached by both b and $r(b)$. It can thus be uploaded $\left\lfloor c(u^* - u_b - \frac{4\mu^4+1}{c}) \right\rfloor + \lfloor cu_b \rfloor \geq c + 3\mu^4$ times according to the bound on c and using $\mu > 1$. Boxes considered in the bound (1) of the proof of Lemma 2 concern preloaded stripes. We can thus count an upload at least $\frac{c+3\mu^4}{c}$ for each such box with it associated rich box. The same is true for postponed stripes that are forwarded through $r(b)$. Each time we count the cache of b in $|B(X)|$ for these forwarded stripes in the proof of Lemma 2, we thus have a similar upload capacity of $c + 3\mu^4$ stripes. The only difficulty comes from postponed stripes downloaded directly by a box b. This represents at most $c_b \leq cu_b - 4\mu^4$ stripes. However, b can upload $\lfloor cu_b \rfloor \geq cu_b - 1 \geq c_b + 3\mu^4$ stripes. In any case, if $i_1(v)$ stripes of a video v are requested in X and b is counted in $|B(X)|$ in the proof of Lemma 2, b and $r(b)$ have upload capacity at least $i_1(v) + 3\mu^4$. (This is also true for rich boxes.) When we sum up box uploads instead of counting them, we obtain $U_{B(X)} \geq \sum_v \left(\frac{|X_{t_v}^{post}|}{c} - i_1(v)\right)\frac{c+3\mu^4}{c} +$

$\frac{|X'(v)|+|X^{pre}_{t_v-1}|+|X^{pre}_{t_v}|-1}{i_1(v)+2(\mu^4-1)}\frac{i_1(v)+3\mu^4}{c}$, and thus $U_{B(X)} \geq \sum_v \left(\frac{|X^{post}_{t_v}|}{c+2(\mu^4-1)} - i_1(v)\right)\frac{c+3\mu^4}{c} + \frac{|X'(v)|+|X^{pre}_{t_v-1}|+|X^{pre}_{t_v}|-1}{c+2(\mu^4-1)}\frac{c+3\mu^4}{c}$. We finally get $U_{B(X)} \geq \frac{i-i_1(c+2(\mu^4-1)+1}{c+2(\mu^4-1)}\left(1+\frac{3\mu^4}{c}\right)$.

We claim that Lemma 4 stills holds. The rest of the proof of Theorem 1 can then be immediately applied to this case.

Following the proof of Lemma 4, Case 1 comes directly from the generalization of Lemma 2: we get $U_{B(X)} \geq \frac{i}{c}$ for $i_1 \leq \nu i$ with $\nu = \frac{1}{c+2\mu^4-1} - \frac{1}{c+3\mu^4}$. Case 2 relies on the u^*-storage-balanced assumption. Consider a rich box a. Let $U^r = \sum_{b|r(b)=a} u^* + 1 - 2u_b$ denote the total upload that has been reserved on a. Some of the upload reserved for poor box b with $r(b) = a$ is statically reserved (at most $(c-c_b)\frac{1}{c} < 1 - u_b + \frac{4\mu^4+1}{c}$) and the other part can be used to answer any requests concerning allocation stripes, i.e. stored according the random allocation on a. (Note that this remaining upload cannot be used to serve requests concerning cached stripes except those of b). The statically reserved upload U^s on a is thus bounded by $U^s \leq \sum_{b|r(b)=a} 1 - u_b + \frac{4\mu^4+1}{c}$ The upload available on a for answering allocation stripes is thus $u'_a = u_a - U^s \geq u_a - U^r + \sum_{b|r(b)=a}(u^*+1-2u_b) - (1-u_b+\frac{4\mu^4+1}{c}) \geq u_a - U^r + (\sum_{b|r(b)=a} u^* - \frac{4\mu^4+1}{c}) - (\sum_{b|r(b)=a} u_b)$. As $U^r = (\sum_{b|r(b)=a} u^*+1) - 2(\sum_{b|r(b)=a} u_b)$, we obtain $u'_a \geq u_a - \frac{1}{2}U^r + (\sum_{b|r(b)=a} u^* - \frac{4\mu^4+1}{c}) - \frac{1}{2}(\sum_{b|r(b)=a} u^* + 1) \geq u_a - \frac{1}{2}U^r + \frac{1}{2}(\sum_{b|r(b)=a} u^* - 1 - \frac{8\mu^4+2}{c})$. The bound on c implies $u^* \geq 1 + \frac{10\mu^4}{c}$ and thus $u'_a \geq u_a - \frac{1}{2}U^r \geq \max\left\{u^*, \frac{u_a}{2}\right\}$. As the system is u^*-storage-balanced, we have $\frac{d_a}{u_a} \leq \frac{d}{u^*}$. By using only storage $d'_a \geq \frac{d_a}{2}$, we may still assume $\frac{d'_a}{u'_a} \leq \frac{d}{u^*}$. The overall storage is at least half used, and catalog is reduced by a factor at most 2 by this operation. On the other side, a poor box b can use all its upload capacity for allocation stripes.

For each box b, we truncate its upload to a multiple of $\frac{1}{c}$. (We may again loose a negligeable fraction of the storage in this operation.) As mentionned previously, a poor box b together to $r(b)$ can upload $\left\lfloor c(u^* - \frac{2\mu^4}{c} - u_b)\right\rfloor + \lfloor cu_b \rfloor \geq c + 3\mu^4 \geq u'c$ stripes (counting only on remaining reserved upload). A rich box can upload at least $\lfloor u^* c \rfloor \geq c + 10\mu^4 - 1 \geq u'c$ stripes (without using reserved upload). The average upload thus remains at least u'. We then virtually split b into a collection of elementary sub-boxes with upload capacity $\frac{1}{c}$ and storage capacity $\frac{d_b}{u_b c} \leq \frac{d}{u'c}$. A set E of boxes with overall upload capacity $U_E \leq \frac{i}{c}$ thus contains at most i elementary sub-boxes. We thus consider the $\binom{unc}{i}$ sets of i elementary sub-boxes. Such a set E' corresponds to a storage space $D_{E'} \leq \frac{d}{u'c} i$. Lemma 3 can clearly be generalized to bound the probability that the ki_1 replicas of the stripes considered fall into these memory slots by $\left(\frac{di/(u'c)}{dn}\right)^{ki_1}$. We thus finally obtain the same bound.

Using $c = \left\lceil \frac{10\mu^4}{u^*-1} \right\rceil$ and $u^* \leq 2$, we can obtain $\nu^{-1} = O\left(\frac{\mu^4}{(u^*-1)^2}\right)$ and $u' \geq \frac{u^*+3}{4}$. This yields catalog size $\Omega\left(\frac{(u^*-1)^2 \log \frac{u^*+3}{4}}{\mu^4}\frac{dn}{\log d'}\right)$.

5. Conclusion

In this paper, we show an average upload bandwidth threshold for enabling a scalable fully distributed video-on-demand system. Under that threshold, scalable catalog cannot be achieved. Above the threshold, linear catalog size is then possible and the problem of connecting nodes to serve demands reduces to a maximum flow problem. A similar threshold is shown for heterogeneous systems. Interestingly, our bound on catalog size measures the trade-off between video quality and catalog size when the upload bandwidth is fixed: for higher video bit-rate, we obtain better quality, but the normalized upload u tends to 1 and our lower bound on catalog size tend to 0 proportionally to $(u-1)^2 \log \frac{u+1}{2} \sim (u-1)^3$.

References

[1] M. S. Allen, B. Y. Zhao, and R. Wolski. Deploying video-on-demand services on cable networks. In *Proc. of the 27th Int. Conf. on Distributed Computing Systems (ICDCS)*, pages 63–71, Washington, DC, USA, 2007. IEEE Computer Society.

[2] S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, and P. Rodriguez. Exploring VoD in P2P swarming systems. In *INFOCOM*, pages 2571–2575, 2007.

[3] Y. Boufkhad, F. Mathieu, F. de Montgolfier, D. Perino, and L. Viennot. Achievable catalog size in peer-to-peer video-on-demand systems. In *Proc. of the 7th Int. Workshop on Peer-to-Peer Systems (IPTS)*, pages 1–6, 2008.

[4] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: High-bandwidth multicast in cooperative environments. In *Proc. of the 19th ACM Symp. on Operating Systems Principles (SOSP)*, 2003.

[5] B. Cheng, X. Liu, Z. Zhang, and H. Jin. A measurement study of a Peer-to-Peer Video-on-Demand system. In *Sixth Int. Workshop on Peer-to-Peer Systems (IPTPS)*, pages 1–6, 2007.

[6] Y. R. Choe, D. L. Schuff, J. M. Dyaberi, and V. S. Pai. Improving VoD server efficiency with BitTorrent. In *MULTIMEDIA '07: Proc.of the 15th Int. Conf. on Multimedia*, pages 117–126, New York, NY, USA, 2007. ACM.

[7] B. Cohen. Incentives build robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[8] A. Gai and L. Viennot. Incentive, resilience and load balancing in multicasting through clustered de bruijn overlay network (prefixstream). In *Proc.of the 14th IEEE Int. Conf. on Networks (ICON)*, volume 2, pages 1–6. IEEE Computer Society, September 2006.

[9] P. K. Gummadi, S. Saroiu, and S. D. Gribble. A measurement study of Napster and Gnutella as examples of peer-to-peer file sharing systems. *Computer Communication Review*, 32(1):82, 2002.

[10] S. B. Handurukande, A.-M. Kermarrec, F. L. Fessant, L. Massoulié, and S. Patarin. Peer sharing behaviour in the eDonkey network, and implications for the design of server-less file sharing systems. *SIGOPS Oper. Syst. Rev.*, 40(4):359–371, 2006.

[11] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross. Insights into pplive: A measurement study of a large-scale p2p iptv system. In *In Proc. of IPTV Workshop, Int. World Wide Web Conf.*, 2006.

[12] C. Huang, J. Li, and K. W. Ross. Can internet video-on-demand be profitable? *SIGCOMM Comput. Commun. Rev.*, 37(4):133–144, 2007.

[13] Y. Huang, Z. Fu, D. Chiu, J. Lui, and C. Huang. Challenges, design and analysis of a large-scale p2p vod system. In *ACM Sigcomm 2008*, 2008.

[14] V. Janardhan and H. Schulzrinne. Peer assisted VoD for set-top box based IP network. In *Peer-to-Peer Streaming and IP-TV Workshop (P2P-TV)*, pages 1–5, 2007.

[15] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat. Bullet: High bandwidth data dissemination using an overlay mesh, 2003.

[16] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *IPTPS '01: First Int. Workshop on Peer-to-Peer Systems*, pages 53–65, London, UK, 2002. Springer-Verlag.

[17] N. Parvez, C. Williamson, A. Mahanti, and N. Carlsson. Analysis of bittorrent-like protocols for on-demand stored media streaming. In *Proc.of the 2008 ACM SIGMETRICS Int. Conf.*, pages 301–312, 2008.

[18] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *SIGCOMM*, pages 161–172, New York, NY, USA, 2001. ACM.

[19] M. Ripeanu. Peer-to-peer architecture case study: Gnutella network. In *Proc. of the 1st IEEE Int. Conf. on Peer-to-Peer (P2P 2001)*. IEEE Computer Society, 2001.

[20] A. I. T. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware*, pages 329–350, 2001.

[21] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Netw.*, 11(1):17–32, 2003.

[22] K. Suh, C. Diot, J. F. Kurose, L. Massoulié, C. Neumann, D. Towsley, and M. Varvello. Push-to-Peer Video-on-Demand system: design and evaluation. *IEEE Journal on Selected Areas in Communications, special issue on Advances in Peer-to-Peer Streaming Systems*, 25(9):1706–1716, December 2007.

[23] D. Tran, K. Hua, and T. Do. Zigzag: An efficient peer-to-peer scheme for media streaming, 2003.

[24] D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava. On peer-to-peer media streaming. In *Proc. of the 22nd Int. Conf. on Distr. Comp. Systems (ICDCS)*, pages 363–371, 2002.