Max Percentile Replication for Optimal Performance in Multi-Regional P2P VoD Systems

Yuval Rochman[§]
Tel-Aviv University
Tel-Aviv, Israel
Email: yuvalroc@gmail.com

Hanoch Levy[§]
Tel-Aviv University
Tel-Aviv, Israel
Email: hanoch@cs.tau.ac.il

Eli Brosh
Vidyo
New Jersey, USA
Email:eli@vidyo.com

***Abstract*—Peer-to-peer based (P2P) VoD systems have proven to be an effective solution for scalable video distribution. In P2P VoD, each peer contributes storage to replicate videos and assist video delivery. A fundamental question is how to optimally replicate video content across the peers so as to maximize their upload capacity. We study this question within the context of a large-scale P2P network where peers are grouped into different geographical regions, and downloading a video across regions is more expensive than within a region. Our analysis addresses the combined challenge of (1) optimizing the replica allocation (placement) with respect to an arbitrary stochastic demand distribution, and (2) finding an optimal assignment of video requests to peers. The problem addressed can model other applications including inventory problems. Our main result is that optimal replica placement in single- and multi-region environments is of max percentile nature. We derive optimal algorithms and show that they have low complexity and thus very practical. We use numerical analysis and simulation to evaluate the system performance and study its behavior. Our results can be used to provide valuable insights on the design of P2P VoD systems.**

I. Introduction

Video-on-Demand (VoD) services have experienced an explosive growth in recent years. Traditional VoD systems are based on a client-server architecture which incurs expensive provisioning costs and has limited scalability. The peer-to-peer (P2P) approach has emerged as an effective solution for scalable content distribution. It has been successfully used by file download and live video streaming systems [1], [2]. Recently, there have been various efforts to build peer-based VoD systems, for example using managed devices [3], [4]. In a P2P VoD system, peers use their storage space and upload bandwidth to replicate video content, serving it to other peers on-demand. Compared to live streaming, VoD users are less synchronous and may not have the same content to share with others. The lack of synchrony is mitigated by letting peers serve content that is different than that being currently viewed.

We focus on a managed P2P environment in which peers remain owned and under the control of the content provider, e.g., like cable set-top boxes. In such a system, video servers must still be deployed to complement missing content and guarantee quality of service. Hence, the P2P network acts as a mechanism to offload the provider's servers in the data centers.

[§] This research was supported in part by the NET-HD consortium.

The effectiveness of a P2P network is largely dependent on the number and location of video replicas at the peers [3]. A fundamental design issue in P2P VoD is to determine the right *replica placement (allocation) strategy* so as to make best use of the upload capacity of peers. While there exists various literature on P2P replica placement, it is mainly geared towards file sharing systems. For example, [5] optimized the network bandwidth usage in replication; while [6] maximized file availability. These works pay little attention to the bandwidth limitation of peers, a key concern for VoD. More recently, the placement problem has been studied for P2P VoD [7], [8]. However, these works focus on flat networks, and do not consider hierarchical network topologies often used in practice.

In this paper, we seek to derive replica placement strategies that enable content providers to maximize the use of peers' upload bandwidth, hence minimizing the cost of servicing video requests. Such strategies should account for *arbitrary* video access distributions to accommodate the diversity of usage patterns exhibited by existing VoD services such as Netflix, IPTV, and YouTube [4]. Perhaps as importantly, they should account for the provider's network topology, typically hierarchically structured. To this end, we consider a *multi-region* P2P model where peers are grouped into different regions, and serving a video across regions is more expensive than within a region. Service can be granted by dedicated servers as fall back, but at higher cost. The multi-region model is a natural fit for the network of a large content provider. For example, cable operators use video hub offices in each metropolitan area (all inter-connected) to serve the local subscribers [9], [10]. The model can be generalized to a more complex architecture where the regions are organized into a k-level hierarchy, as shown in a technical report [11].

Optimizing system-wide video servicing costs requires us to deal with a *placement* problem, namely determining the quantities and locations of the movie replicas across the regions. This problem is to be addressed under a realistic service assumption that the number of offered movies can be high (e.g., Netflix's catalog includes more than 100,000 titles [12]). For each of these movies the demand distribution (which can be derived from operator predictions) may be completely arbitrary; and differs from that of the other movies. Since the expected revenue of any placement (allocation) depends

on the way incoming requests are handled, we also need to deal with an *assignment* problem, namely, find a maximal matching between the movie demands and peer servers. Of course, the two problems affect each other and therefore a combined solution must be derived.

This combined problem may seem at first sight to be quite challenging. The difficulty is that in order to find an optimal allocation one must: 1) Examine all possible (multi-dimensional) demand realizations; 2) consider for each of them the optimal assignment; 3) account for the ensemble of all these solutions to derive the expected revenue of the allocation; and 4) optimize over all allocations. Nonetheless, our analysis reveals that the problem benefits from certain structural properties which allow one to decompose it and derive an optimal solution using exact analysis.

We believe that this work is a first attempt to deal in an exact way with the combination of the assignment and allocation problems under arbitrary stochastic demand and in one model. Our main result is that optimal replica placement in a single- and multi-region P2P networks is of *max percentile* nature; that is, it is based on the tail distributions of the demand. This is in contrast to past results where various replication solutions are based on the proportional mean (we discuss their performance in detail in Section X). We show that the problem can be formulated as a (linear) revenue optimization problem which is analytically manageable, and that it possess a *balancing* property which narrows the state space of the solution to that of symmetrically-full balanced allocations.

We derive optimal algorithms for the problem and demonstrate that they are of relatively low complexity and thus very practical. The optimality of the solution and the algorithms can be used either to achieve optimal operation or as a benchmark, or to provide operational guidelines for the design of P2P replication schemes. We use numerical analysis and simulation to validate our results and study the system's behavior. We demonstrate how to evaluate the performance of alternative placement strategies relative to that of the optimal.

The rest of this paper is organized as follows: In Section III we describe the model and the problem. In Section IV we expose the reader to the principles of the analysis by first presenting the solution to the problem of a single region. Then, in Sections V, VI and VII we turn to the more general multi-region problem: Section V deals with the assignment problem; Section VI analyzes balanced-allocations which are a key to the overall solution, and Section VII provides the solution of the placement problem. In Section VIII we provide efficient implementation of the algorithms. Finally, Section X uses the solution to evaluate the system performance. Most of the article proofs, an efficient implementation of the algorithms, and a generalization of the analysis for a k-level hierarchy are presented in [11].

II. Related Work

There have been previous investigations of P2P content placement, the majority of which focus on different environment settings and goals. Several studies have considered content placement in file sharing systems. For example, [5] was perhaps the first to study a network model similar to ours, where peers can be connected using an exponential expansion topology. Their goal is to place movie replicas in peers to minimize the average number of links traversed in a download process. The optimization is done from a viewpoint of a single random downloader rather than for the aggregate demands, as in our case. The major result of [5] suggests that optimal replication should be proportional to the mean demand. In contrast, under our VoD-oriented model optimal replication follows a max percentile solution (Section X discusses in detail the performance of these allocations). [6] tried to optimize file availability when peers are infrequently online. [13] studied a similar problem where peers are associated with weights. [14] proposed a replication protocol for file sharing applications in mobile ad hoc networks aiming to reduce the average file querying time. The goal in [15] is to minimize the number of access failures assuming that every requesting peer randomly accesses a peer server. These works focus on a different goal than ours, and consequently establish different placement rules such as square root or log proportional.

There have been several works on replica placement in VoD environments. [3] studied large-scale P2P VoD system deployed in the Internet and demonstrated that movie replication is a key design issue. They propose a heuristic replication algorithm based on proportional mean, which we compare against in our performance evaluation. [4] introduced a gateway-based P2P architecture and considered popularity-based content placement driven by a heuristic linear program. These works serve as a good exposition to the problem, motivating our work.

Relatively close works to ours are [7], [8]. [7] proposes the RLB algorithm and proves it to be optimal. The analysis is carried out under the assumption that the number of movies is much smaller than the number of peers. When assumed otherwise, max-percentile algorithm preforms better than RLB, as it can seen in Section X. [8] provided an asymptotic analysis of the system as the number of peers approaches infinity. The solution is an approximate one since it is based on approximating an integer non-linear problem by real value problem. Furthermore, these works are limited to a small-scale flat P2P network, and do not consider hierarchical network topologies used in practice by providers. Note that in a single region setup the solution in [8] is identical to the proportional mean solution in [5]. [10] considers a large-scale VoD service with a hierarchical network infrastructure that resembles our multi-region approach. Also, [16] suggested a replica system in CDN with geographical distances. While we consider stochastic demand (which includes as a special case deterministic demand), the previous two articles assume a deterministic demand.

The work in [17] proposed a push-to-peer architecture and content replication strategies, where movie parts are encoded using rateless code and pushed on set-top boxes so as to serve them on-demand. That work tackles a different goal, no optimization of placement with respect to demand is

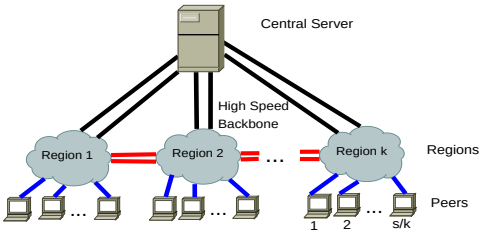

Fig. 1. System topology

attempted. The work in [18] analyzed the conditions for catalog scalability under this architecture. [19] considered a time slotted replication model where video requests are assigned to randomly selected peers. They show that the cache replacement strategy can be expressed as a dynamic program. However, the solution has exponential time complexity. Alternative solutions, e.g [20], propose to use the set of peers interested in a specific video to form a P2P tree dedicated to that video, and use it to broadcast the video to all the tree members; such approaches benefit from flexibility and scalability while may be sensitive to the delay and some uncertainty that may be caused by a tree structure (as opposed to receiving the video directly from a peer who holds it as done in our work).

III. The Model and the Problem

We consider a system consisting of user terminals (peers) designed to download and playback movies to the users. In addition to its download and playback capabilities, each terminal is equipped with storage and upload capabilities. These capabilities can be used to store movies at the terminals and upload them to other terminals upon request. A movie request made by a peer and which cannot be served by another peer is processed and granted by a central server.

We consider a topology as depicted in Figure 1. The system consists of k *regions* numbered $1, 2, ..., k$. The peers from the same region can talk to each other directly via a video hub, whose bandwidth capacity is large enough to allow each peer in the region both to download and upload a movie concurrently. Note that a peer can be viewed as both server (when it uploads), which we will call a *peer-server* and a client (when it downloads), which we will call a *peer-client*. The k hubs are connected to each other via a high speed backbone network. This allows a peer-server in region i to upload a movie to a peer-client in region j.

Having described the topology, we next make the following modeling assumptions:

1) **Peer storage capacity:** We assume that each peer-server can store 1 movie. This assumption will be relaxed in Section IX.
2) **Number of Peers and Peer-Servers:** The total number of peers is s. We assume that each peer can be utilized as a peer server and thus the number of peer-servers is s. We assume that each region contains s/k peers-servers (mathematically, assume that s is divisible by k).
3) **Peers occupancy (analysis assumption):** For the sake of presentation, we will assume throughout the analysis that each server contains exactly one movie replica (no empty servers). This implies that the number of replicas in a region is s/k. This assumption is relaxed in Section IX.
4) **Peer Upload/Download capacity:** We assume that each peer-server can upload to any peer-client. At a given time a peer-server can upload *only to one peer-client*; this is based on assuming that the upload capacity is in the *order* of the playback capacity and the download capacity is greater than (or equal to) the upload capacity (both are approximately the case in today's networks).
5) **Movie demand:** We assume that there are m movies, indexed $1, 2, ..., m$, in the system. We consider a static demand reflecting the demand at peak hours. Let N_i^j be a random variable denoting the number of requests for movie i made by peer-clients in region j. We assume symmetric demands, namely that the regional demands are statistically identical. That is, all the $\{N_i^j\}_{j=1}^k$ variables are identically distributed; let $\tilde{N}_i$ denote a generic variable having the same distribution.

   We do not make any assumption on the distribution of $\tilde{N}_i$, namely it can be of an *arbitrary distribution*[1]. Further, we *do not assume independence* between the demands, namely $N_{i_1}^j$ and $N_{i_2}^j$ are not necessarily independent of each other and so are $N_i^{j_1}$ and $N_i^{j_2}$. The set $\{N_i^j\}$, $1 \le j \le k$, $1 \le i \le m$ is the demand set, or in short, the demand. We will use $\{N_i^j\}$ to denote this demand.
6) **Request service cost parameters:** Consider a request made by a client in region i. The request can be downloaded from either of: 1) A peer-server in region i, 2) A peer-server in a different region. In the event that it cannot be served by the peer servers it can always fall back to be served by a central server (which is significantly more costly). We denote the cost of downloading (serving) the request in these cases by C_{loc} (local cost), C_{rem} (remote cost) and C_{ser} (central server). We assume that the costs obey $C_{ser} \ge C_{rem} \ge C_{loc}$. The latter inequality results from the structure of the system implying that downloading a movie across regions is more expensive than within a region.

A. The problem

The objective of the system is to minimize the cost of servicing the requests. To this end, we assume that each request is granted either from the peer system or from the

[1]While the mathematical analysis will hold for any arbitrary distribution, one can pick a more restrictive distribution to reflect the physical peer model. To this end, it may make sense to assume that the total demand in a region is bounded from above by the number of peer-clients in the region (equaling the number of peers, s/k) since a client cannot request more than one movie. Also, one possible reasonable choice of the distribution is a binomial distribution $B(s/k, p_i)$ where p_i would model the probability that a single peer-client is interested in movie i.

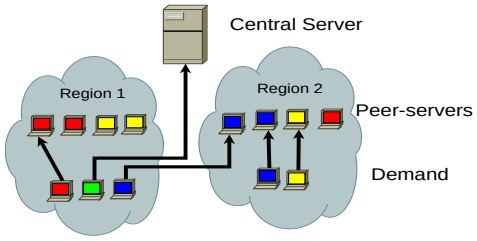

Fig. 2. Matching Example

central server. Let g_{ser}, g_{rem} and g_{loc} denote the number of requests granted (served) from the server, from a remote region or from a local region, respectively. Since all requests are granted, obviously $g_{ser} + g_{rem} + g_{loc}$ is the number of requests, N. The system request service cost is given by

$$C = C_{ser} \cdot g_{ser} + C_{rem} \cdot g_{rem} + C_{loc} \cdot g_{loc}, \qquad (1)$$

and the objective is to minimize this service cost.

An example of the system matching modeling is in Figure 2. In this example, we have $k = 2$ regions, each with $s/k = 4$ peer-servers. The number of requests granted (served) from the server, from a remote region and from a local region, are 1, 1 and 3, respectively. Therefore the cost in this case is $C_{ser} + C_{rem} + 3 \cdot C_{loc}$.

To this end, one should note that the system operation divides into two parts. First, at off-line mode, replicas of the movies are placed at the peers[2]. This replica placement can be based on the knowledge of the demand distribution. Second, once the replicas are placed in the peers, the system is faced with an actual demand, which is a *realization* of the demand distribution, at which time the system needs to decide how to assign the movies to the various demands. We call the former the *replica placement problem* and the latter the *assignment problem*. Note that the assignment problem can be solved in isolation; nonetheless the solution of the placement problem depends on that of the assignment problem.

To this end, let L_i^j denote the number of movie i replicas that are stored in region j, and L_i the number of movie i replicas placed in the whole system; These obey $\sum_{i=1}^{m} L_i^j = s/k$ (see Assumption 3 above) and $\sum_{j=1}^{k} L_i^j = L_i$. The set $L = \{L_i^j\}$, which is called an *Allocation*, is the output of the replica placement problem and the input to the assignment problem.[3] Formally the problems can be stated as follows:

1) The *assignment (matching) problem*: Given an allocation, $L = \{L_i^j\}$, a demand realization, denoted by n_i^j, $i = 1, ..., m$, $j = 1, ..., k$, and the service cost parameters $C_{ser}, C_{rem}, C_{loc}$, assign (match) the servers to the demands as to minimize the service cost C.
2) The *replica placement(allocation) problem*: Given the movie demand distributions $\{N_i^j\}$, $i = 1, ..., m$, $j = 1, ..., k$, the service cost parameters $C_{ser}, C_{rem}, C_{loc}$, and a matching algorithm solving the assignment problem, determine the replica allocations $L = \{L_i^j\}$, $i = 1, ..., m$, $j = 1, ..., k$, of each movie in each region as to minimize the expected cost $E[C]$.

Our objective in this work is to solve the replica placement problem; this, in turn, will be assisted by a solution of the assignment problem. For the convenience of the reader a glossary of notation is provided in [11].

B. Transforming the Cost Function to a Revenue Function

For the analysis of the assignment and replica placement problem it will be convenient to transform the cost value problem to a revenue value problem. The way we defined the transformation is critical, and defining it differently may complicate the analysis. The transformation is established next:

Claim 3.1: The following holds:

1) A matching algorithm M solves the assignment problem iff M maximizes the following function:

$$R = (C_{ser} - C_{rem})g_{glo} + (C_{rem} - C_{loc})g_{loc}$$

where g_{glo}, g_{loc} represent the number of requests granted by all the peers (i.e. Global Matching, equals $g_{loc}+g_{rem}$) and the number of requests granted by peers of the same region (i.e. Local Matching), respectively.
2) An allocation L solves the placement problem iff the allocation maximizes $E(R)$.

Proof: This proof is presented in [11]. ■

For convenience we set $R_{glo} \doteq C_{ser} - C_{rem} \geq 0$ and $R_{loc} \doteq C_{rem} - C_{loc} \geq 0$. We will denote the *revenue objective function* to be:

$$R = R_{glo} \cdot g_{glo} + R_{loc} \cdot g_{loc}. \qquad (2)$$

By Claim 3.1 we have that a matching algorithm M solves the assignment problem iff M maximizes the revenue objective function, R, and an allocation L solves the placement problem iff it maximizes the expected value of the revenue objective function, $E(R)$.

IV. Analysis Exposition: A Single Region System

For the sake of exposition we start our analysis by considering the case of a simplistic system, consisting of a single region, say region 1. This system is a simple special case of the full model presented in Section III. For the sake of brevity, the analysis in this section will focus on stating the results and providing some intuitive explanations without providing rigorous proofs. [4]

In this system, peers can download movies only from the local region's peers. Thus, the number of requests granted in the global matching is equal to the number of requests granted

[2]In practice, content is pushed to the peers during off-peak hours.

[3]Note that the formulation takes into account only the number of replicas in a region and *not* the specific servers at which they are placed. This stems from the fact that the storage of each server is 1 and thus all servers in the region are interchangeable.

[4]Of course, rigorous proofs are not needed, since this system is a special case of the general model whose results will be proved in the subsequent sections.

in the local matching, i.e $g_{glo} = g_{loc}$. Therefore, the objective function is equivalent to maximize the number of requests granted by the local peers.

To solve the replica placement problem, one must first find an optimal solution to the assignment problem. This is true since the assignment problem addresses a single demand realization case while the replica problem addresses the ensemble of all possible demand realizations. The first ingredient in our placement solution is therefore to derive an optimal solution to the assignment problem, whose main advantage is that it provides a closed form expression of the revenue (R) of the solution. Recalling that the demand for movie i and the number of servers holding movies i are denoted by L_i^1 and n_i^1, respectively, the solution we derive is given by:

$$R = \sum_{i=1}^{m} \min(L_i^1, n_i^1). \tag{3}$$

The reader may verify that one cannot offer a higher assignment (match), and that this assignment is indeed possible.

This equation expressing the revenue of an optimal assignment now serves as our first key result. It is a key for addressing the replica placement problem, whose objective can now be simply stated as maximizing the expected revenue:

$$E(R) = \sum_{i=1}^{m} E(\min(L_i^1, N_i^1)), \tag{4}$$

where N_i^1 is a random variable denoting the number of movie i requests in region 1.

Our second key result is the observation that since $\min(L_i^1, N_i^1)$ gets non-negative integer values, then each term in Eq. (4) can be transformed to a simple sum of probabilities:

$$E(\min(L_i^1, N_i^1)) = \sum_{j=1}^{L_i^1} \Pr(N_i^1 \geq j). \tag{5}$$

The objective of the replica placement problem is therefore to maximize the function

$$E(R) = \sum_{i=1}^{m} \sum_{j=1}^{L_i^1} \Pr(N_i^1 \geq j), \tag{6}$$

by selecting the set $L = \{L_i^1 | 1 \leq i \leq m\}$ under the constraint(see modeling Assumption 3 in Section III) that $\sum_{i=1}^{m} L_i^1 = s$, namely that one places exactly s replicas in the region.

The structure of this equation serves as a key for the analysis as well as to its interpretation. The key implied-property of this equation is that adding a movie i replica to the peer allocation L given in the above equation will increase the expected revenue by $\Pr(N_i^1 \geq L_i^1 + 1)$. We thus may define a "delta" function $\delta_i()$ whose values are $\delta_i(l) = Pr(N_i^1 \geq l)$. As such, the objective function is to select a set L such as to maximize $\sum_{i=1}^{m} \sum_{j=1}^{L_i^1} \delta_i(j)$ under the constraint $\sum_{i=1}^{m} L_i^1 = s$.

Having this view in mind we can now turn to describe our Max Percentile algorithm whose description may be assisted by Figure 3, which depicts an example of 3 movies (and therefore three $\delta()$ vectors). The (Percentile) algorithm is carried out by going over the $\delta_i()$ vectors and using a greedy approach to select their highest values. This is done easily by observing that each $\delta_i()$ vector is monotonically non increasing (as in Figure 3). Thus the algorithm is carried out by keeping three types of $\delta_i()$ variables: 1) Already selected variables (dark background, on the left hand side), 2) Candidates for selection – at most one element for each i (light background), and 3) Non-examined variables (white background, on the right hand side). Using this classification the $\delta_i()$ selection algorithm progresses by selecting the maximal value element from the set of candidates ($4th$ element in first vector in the example) and then adding to the candidate set the element appearing to the right of the selected candidate ($5th$ in the first vector). The algorithm terminates when the number of allocated replicas equals the available number of replicas, s. The reader may observe that the percentile algorithm indeed selects the s highest values of the $\delta_i()$ vectors. This completes the Max Percentile algorithm which will return, as a solution of the replica placement problem, the selected set L where L_i^1 equals the number of elements selected from the $\delta_i()$ vector.

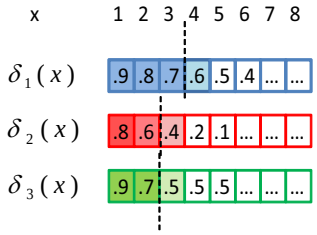

Fig. 3. The Max Percentile "delta" vectors

V. Multi-Region: The Assignment Problem and the Matching Algorithm

The assignment problem can be modeled via a maximal weighted matching problem in weighted graphs (see [21]) where a matching is made between the set of peer-servers that possess the movies and the set of demands for the movies. An edge in this bipartite graph is drawn between a peer server that holds a specific movie and the demand for that movie. The weight of the edge is its revenue, either R_{glo} or $R_{glo} + R_{loc}$ according to the case. The solution to this problem can be achieved via the Hungarian Algorithm at a complexity higher than $\Omega((n+s)^2)$.

In Algorithm 1 below we derive a specific matching algorithm tailored for our problem. Its advantages over the Hungarian Algorithm are two: 1) Its complexity is linear. 2) It yields a closed form expression of the revenue it achieves, as will be given in Eq. (7).

An effective implementation of the assignment algorithm works in linear time, $O(s+n)$ (See Section VIII). In the following claim, we will prove that the assignment algorithm

Algorithm 1 The assignment algorithm

Require: An allocation of peer-servers ,$L = \{L_i^j\}$ and the demand n_i^j, $i = 1, ..., m$, $j = 1, ..., k$.

1: **for all** movie i **do**
2: **for all** region j **do**
3: Take $\min(L_i^j, n_i^j)$ requests from the j^{th} region of the i^{th} movie, and match them to $\min(L_i^j, n_i^j)$ peer-servers in the j^{th} region, containing the i^{th} movie.
4: **end for**
5: Let n_i^{rem} be the number of movie i requests, which we did not match in Step 2, and let L_i^{rem} be the number of unmatched movie i peer-servers.
Then, match the remaining $\min(n_i^{rem}, L_i^{rem})$ requests to the remaining $\min(n_i^{rem}, L_i^{rem})$ peer-servers.
6: **end for**

assigns maximal matching, and give an expression for its revenue.

Claim 5.1: Given allocation $L = \{L_i^j\}$ and demand realization n_i^j, $i = 1, ..., m$, $j = 1, ..., k$, the following claims hold:

1) The assignment algorithm yields a revenue of:

$$R_{glo}\sum_{i=1}^{m}\min(L_i, n_i) + R_{loc}\sum_{i=1}^{m}\sum_{j=1}^{k}\min(L_i^j, n_i^j). \quad (7)$$

2) The assignment algorithm will maximize the revenue objective function. Namely, there exists no matching algorithm with higher revenue.

Proof: The proof is presented in [11]. ∎

An immediate and important result of Claim 5.1 is that for any arbitrary allocation L and for Any *stochastic* demand $\{N_i^j\}$ the matching algorithm will maximize the expected revenue.

Corollary 5.2: Given allocation L conducted by a placement algorithm, the expected revenue, under the assignment algorithm, will be:

$$E(R^L) = R_{glo}\sum_{i=1}^{m}E(\min(L_i, N_i)) + R_{loc}\sum_{i=1}^{m}\sum_{j=1}^{k}E(\min(L_i^j, N_i^j)). \quad (8)$$

Moreover, this revenue is maximal over all matching algorithms.

Proof: The proof is presented in [11]. ∎

This corollary will guarantee that replica placement problem is to to maximize the function in (8), when the free variables of the function are the movies replica, L_i^j.

VI. Balanced Allocations

A. Introduction

Having derived an optimal solution to the assignment problem, we next aim at solving the placement problem, namely at finding an optimal allocation. On its face value this problem seems to be a hard one since the number of allocations is exponential, which may lead to exponential complexity. Fortunately, as revealed by our analysis, the placement problem benefits from possessing an important balancing property which we call *the balance principle*. This will allow us to drastically reduce the complexity by searching for an optimal policy from within a reduced set of allocations, named *balanced allocations*, which we define next.

Definition 1: An *allocation* is a set $L = \{L_i^j | 1 \le i \le m, 1 \le j \le k\}$ of non-negative integer numbers. Let $L_i := \sum_{j=1}^{k} L_i^j$. The vector $\hat{L} = (L_1, L_2, \dots L_m)$ will be called the *quantity vector* . If the quantity vector satisfies $\sum_{i=1}^{m} L_i = s$ then $\hat{L}$ is called a *full quantity vector* and L is called a *full allocation*. If for every region j we have $\sum_{i=1}^{m} L_i^j = \frac{s}{k}$ then L is called a *symmetrically-full allocation*.

Definition 2: Let L be an allocation. Then L is called a *balanced allocation* if for every movie i and every two regions j_1 and j_2 we have $|L_i^{j_1} - L_i^{j_2}| \le 1$.

The search for an optimal allocation would require, in principle, examining all the symmetrically-full allocations. As stated above, balanced allocations turn out to serve a key role in reducing such a search. A striking result (shown in Theorem 6.1) is that for every arbitrary symmetrically-full allocation L there exists a symmetrically-full balanced allocation L_B which is at least as good as L. This will serve a key result since it will allow us to conduct the whole optimization over the (narrow) space of symmetrically-full balanced allocations.

Theorem 6.1: Given the total number of peer-servers s and the number of regions k, for every arbitrary full allocation L (which can be a symmetrically-full allocation) there exists a balanced and symmetrically-full allocation L_B with a higher or equal expected revenue.

Note that the above theorem is true regardless of the probability distribution of the demand, $\{N_i^j\}$.

Before proving this theorem we need to prove several properties of allocations. The proof of Thm. 6.1 will be completed at the end of Section VI-C.

B. Properties of allocations

The following two claims regarding balanced allocations and their quantity vectors will be useful in this section and in Section VII:

Claim 6.2: Let L be a balanced allocation, whose quantity vector is $\hat{L} = (L_1, L_2, \dots L_m)$. Then

1) For movie i, $1 \le i \le m$, in every region either $\lfloor \frac{L_i}{k} \rfloor$ or $\lceil \frac{L_i}{k} \rceil$ peer-servers are allocated.
2) The number of regions with $\lfloor \frac{L_i}{k} \rfloor + 1$ peer-severs is equal to $L_i \bmod k$.

3) The contribution of movie i to the expected revenue $E(R^L)$ is:

$$R_{glo}E(\min(L_i,N_i)) + R_{loc}\sum_{j=1}^{r} E(\min(\left\lceil \frac{L_i}{k} \right\rceil, N_i^j)) + R_{loc}\sum_{j=r+1}^{k} E(\min(\left\lfloor \frac{L_i}{k} \right\rfloor, N_i^j))$$

where $r = L_i \, mod \, k$.

Claim 6.3: Given a quantity vector $\hat{L} = (L_1, L_2, \dots L_m)$, the following properties hold:

1) There exists a balanced allocation whose quantity vector is $\hat{L}$.
2) Every two balanced allocations, having the same quantity vector $\hat{L}$, have the same revenue.

The proof of these claims are presented in [11]. The next two claims are also important. We will used them in Subsection VI-C and in Section VII:

Claim 6.4: Let C be an integer constant, and let X be a non-negative integer valued random variable. Then we have:

$$E(\min(X,C)) = \sum_{i=1}^{C} \Pr(X \geq i). \tag{9}$$

Proof: The proof is presented in [11]. ■

Claim 6.5: Let L be an allocation. Let j_1 and j_2 be regions such that $L_i^{j_1} \geq L_i^{j_2} + 2$. Let L' be an allocation produced from L by moving a peer-server with movie i from region j_1 to region j_2 (i.e. setting $L'^{j_1}_i \leftarrow L_i^{j_1} - 1$ and $L'^{j_2}_i \leftarrow L_i^{j_2} + 1$). Let the expected revenues of L and L', be denoted $E(R^L)$ and $E(R^{L'})$ respectively. Then $E(R^{L'}) \geq E(R^L)$.

Proof: This proof is presented in [11]. ■

C. Quantity-Equivalent Allocations

In order to prove Theorem 6.1 we will define and use a quantity-equivalence relation between allocations.

Definition 3: Let L and L' be allocations. We call them *quantity-equivalent allocation* if they have the same quantity vector (i.e. if for every movie i L and L' allocate the same allocation, namely $L_i = L'_i$). The *equivalent class of* L is defined to be all the allocations L' which are quantity-equivalent to L.

It is easy to see that the relation defined is an equivalence relation. The following lemmas will lead directly to the proof of Thm. 6.1:

Lemma 6.6: For every arbitrary full allocation (not necessarily symmetrically-full) L, there exists a balanced and symmetrically-full allocation L_B, which is quantity equivalent to L.

Lemma 6.7: The expected revenue of L_B (as defined in Lemma 6.6), $E(R^{L_B})$, is not smaller than that of L, $E(R^L)$. That is $E(R^{L_B}) \geq E(R^L)$.

Proof of Lemma 6.6:

We should look in following algorithm:

Algorithm 2 Balanced Spread Algorithm

Require: A full quantity vector . I.e $\hat{L} = (L_1, L_2, \dots L_m)$ where $\sum_{i=1}^{m} L_i = s$.

Ensure: A symmetrically-full balanced allocation.

1: **for all** movie i **do**
2: Allocate to every region j, $\lfloor \frac{L_i}{k} \rfloor$ replicas of movie i.
3: Set $r_i \leftarrow L_i - \lfloor \frac{L_i}{k} \rfloor \cdot k$ (Notice that $r_i = L_i \, mod \, k$).
4: **end for**
5: Allocate one additional replica of movie 1 to each of regions $1, 2 \dots, r_1$. Then, allocate one additional replica of movie 2 to each of the regions $r_1+1, r_2+2 \dots, r_1+r_2$ and so on. Note that all region index counting is done modulo k, guaranteeing circular allocation.

Denote L_B to be the allocation returned from a run of the Balanced Spread Algorithm on the quantity vector of L. In [11] we presented a proof that L_B is a symmetrically-full balanced allocation, which is quantity-equivalent to L. ■

Remark 6.8: The construction of the symmetrically-full balanced allocation depends only on the quantity vector and not on how the original allocation L allocates its replicas in any region. The only requirement of the input is that it represents a quantity vector of a full allocation, i.e. $\sum_{i=1}^{m} L_i = s$.

Proof of Lemma 6.7: The proof is presented in [11]. ■

We finally complete the proof of the key theorem of this section:

Proof of Theorem 6.1: Given a full allocation L, its symmetrically-full balanced allocation, L_B, has a higher (or equal) revenue than L, as stated in Lemma 6.7. ■

VII. The Multi Region Max Percentile Algorithm

In this section we describe the *Multi Region Max Percentile Algorithm* (and, in short, the *MuRMaP* algorithm), and prove that the algorithm constructs the allocation with the highest expected revenue.

As shown in the previous section (Thm 6.1) within the class of symmetrically-full allocations the balanced allocations are better than the unbalanced ones. Therefore, the search for an optimal allocation is carried out in the (narrow) space of symmetrically-full balanced allocations. The MuRMaP algorithm will therefore operate by finding the allocations with the highest expected revenue among these allocations.

The next claim will calculate the increase of the revenue when inserting a new replica to the allocation:

Claim 7.1: Given allocation L, inserting a replica of movie i_0 to region j_0, will increase the revenue by:

$$R_{glo} \cdot \Pr(N_{i_0} \geq L_{i_0} + 1) + R_{loc} \Pr(N_{i_0}^{j_0} \geq L_{i_0}^{j_0} + 1).$$

The claim is proved in [11].

By Claim 6.3, we know that for every given quantity vector $\hat{L} = (L_1, L_2, \dots L_m)$ there exists a balanced allocation L_B with a quantity vector $\hat{L}$. The expected revenue of L_B, which

is unique to the quantity vector, will be called the *balanced revenue of the quantity vector* $\hat{L}$ and be denoted by $R_B(\hat{L})$. By inserting a movie to the quantity vector, we increase its balanced revenue, as the next claim states:

Claim 7.2: Let $\hat{L} = (L_1, L_2, \dots L_m)$ be a quantity vector and let $\hat{L}'$ be a quantity vector such that $L'_{i_0} = L_{i_0} + 1$ and $L'_j = L_j$ for all $j \neq i_0$. Then the increase of the balanced revenue, i.e $R_B(\hat{L}') - R_B(\hat{L})$, is

$$R_{glo} \cdot \Pr(N_{i_0} \geq L_{i_0} + 1) + R_{loc} \Pr(\tilde{N}_{i_0} \geq \left\lfloor \frac{L_{i_0}}{k} \right\rfloor + 1). \quad (10)$$

Proof: The proof is presented in [11]. ∎

Remark 7.3: Note that Eq.(10) can be viewed as the marginal increment in the balanced revenue of the quantity vector due to an increase in L_{i_0}. Thus, it can be written as a function $\delta_i(L_i)$ which expresses the contribution of movie i to the balanced revenue of L_i. Eq.(10) becomes

$$\delta_{i_0}(L_{i_0} + 1) = R_{glo} \cdot \Pr(N_{i_0} \geq L_{i_0} + 1) + R_{loc} \Pr(\tilde{N}_{i_0} \geq \left\lceil \frac{L_{i_0} + 1}{k} \right\rceil), \quad (11)$$

where we used $\left\lceil \frac{L_i+1}{k} \right\rceil = \left\lfloor \frac{L_i}{k} \right\rfloor + 1$.

Corollary 7.4: The balanced revenue of quantity vector $\hat{L} = (L_1, L_2, \dots L_m)$ is identical to the integration of the marginal contribution δ. That is

$$R_B(\hat{L}) = \sum_{i=1}^{m} \sum_{j=1}^{L_i} \delta_i(j). \quad (12)$$

This can be proved by a simple induction on $\sum_{i=1}^{m} L_i$.

By the previous claim we can construct in a greedy way a quantity vector, which we call the ***MuRMaP quantity vector***, to have the highest balanced revenue among all the full quantity vectors. The MuRMaP quantity vector is constructed by the following algorithm:

Algorithm 3 Deriving the MuRMaP quantity vector

1: Initiate a new minimum priority queue Q.
2: Initiate a quantity vector $\hat{L} = (L_1, L_2, \dots L_m)$ such that $L_i = 0$ for every movie i .
3: **for all** movie i **do**
4: Insert to Q the value of $\delta_i(1)$.
5: **end for**
6: **repeat**
7: Take movie i with the maximal value in Q.
8: $L_i \leftarrow L_i + 1$.
9: Insert to Q the value $\delta_i(L_i + 1)$.
10: **until** $\sum_{i=1}^{m} L_i = s$
11: **return** the quantity vector, $\hat{L} = (L_1, L_2, \dots L_m)$

In Lemma 6.6 we proved that given a full quantity vector, we can construct a symmetrically-full balanced allocation (See also Remark 6.8). Therefore, constructing a symmetrically-full balanced allocation from the MuRMaP quantity vector is presented in the following algorithm.

Algorithm 4 The max percentile inter region (MuRMaP) algorithm

1: Run Algorithm 3 for finding the MuRMaP quantity vector $\hat{L}$.
2: Run the balanced spread Algorithm (Algorithm 2) on the quantity vector to yield the MuRMaP allocation.

We will prove that the allocation constructed by the MuRMaP algorithm, called the *MuRMaP allocation* is the solution for the replica placement problem:

Theorem 7.5: The MuRMaP allocation obtains the highest expected revenue among all the symmetrically-full allocations.

Proof: The theorem is proved in [11]. ∎

Corollary 7.6: Running the MuRMaP algorithm (Algorithm 4) followed by the assignment algorithm (Algorithm 1) yields the highest expected revenue, among all the placement and assignment algorithms.

A. Optimality of the MuRMaP Allocation Over a Wider Class of Allocations

The analysis carried out above was based on Assumptions 2 and 3 in Section III assuming that each of the regions in the system is allocated exactly s/k replicas. That is, the MuRMaP allocation was shown to be optimal over all allocations for which $\sum_{i=1}^{m} L_i^j = s/k$, $j = 1, ..., k$ (and thus $\sum_{i=1}^{m} L_i = s$). Below we extend the class of allocation over which the MuRMaP allocation is optimal.

A partial allocation is defined to be an allocation L such that $\sum_{i=1}^{m} L_i \leq s$. The following theorem establishes that the MuRMaP allocation is optimal over the class of partial allocations.

Theorem 7.7: The MuRMaP allocation has the highest expected revenue within the class of partial allocations.

Proof: The proof is presented in [11]. ∎

B. Server Allocation

All the analysis carried out prior to Section VII was based on the *assumption* that each region is allocated s/k servers. A possible design question that one may want to consider, and that was not addressed so far, is the *server allocation problem* which can be formulated as follows: If one has s servers, then how many servers one should allocate to each of the regions.

To this end, one may easily conclude from Thm. 7.7 that among all possible server allocations, allocating s/k servers to each region is an *optimal allocation*. Of course, this result is quite intuitive due to the symmetric demand in the regions. Nonetheless - Thm. 7.7 provides a formal proof of this result.

VIII. Complexity of Algorithms: Optimization

The MuRMaP algorithm and the assignment algorithm were written above in a way to clarify their presentation (and not to reduce their complexity). They can be implemented efficiently to reduce their complexity: the assignment algorithm can be implemented in $O(s+n)$ time complexity, where n is the number of requests, and the MuRMaP algorithm can be implemented efficiently and have time complexity of $O(s\cdot(d+\log s))$, where d is the complexity of calculating one element in the probability mass function of the demand distribution (namely, the complexity of computing $\Pr(N_i=j)$). More details on the implementation is given in [11].

IX. Multiple movies in a peer

The analysis given so far dealt with peers storing a single movie. Next, we examine systems where peers can store multiple movies. To this end, we propose to use a heuristic algorithm operating as follows: first, the number of replicas in a region is determined by the Max Percentile algorithm operating on sc peers, where c is the number of copies a peer can hold. Then, the sc/k replicas destined for a region are sorted in the order of 'their marginal contribution' to the revenue, namely, in the order they were selected by the MuRMaP algorithm. Placement of these sc/k replicas in a region is roughly done in an elevator type approach: First place the first s/k largest replicas in decreasing marginal contribution order starting at peer-server 1 of the region and ending at server s/k of the region. Then, place the next largest s/k replicas, starting at server s/k and ending at server 1. Then, continue placing from server 1 to server s/k, and so on. This will provide a relatively balanced spread of marginal contributions over the s/k servers.

X. Performance Evaluation

We use numerical analysis and simulation to evaluate the system's performance. We study video servicing costs and fraction of the demand that can be sustained as a function of the system parameters. The diversity of movies is a crucial parameter for the system performance: the larger the number of movies and the larger the weight of esoteric movies in this population, the harder it is on the system to satisfy their demand. Motivated by previous analytical works [5], [8] and empirical studies on the usage patterns in VoD [22], [4], we assume that the movie demand follows a Zipf distribution. Since it is unclear how future demands will behave (the prior references mentioned above used a Zipf parameter of values 0.56 - 1.5, depending on the study) we consider a wide range of Zipf distributions and vary the Zipf parameter from 0.5 to 1.5. We assume that the aggregate demand consists of n requests, where each of the request picks a movie i with probability p_i, $\sum_i p_i = 1$, and p_i follows a Zipf distribution.

Single-Region. Figure 4 shows the number of requests granted by the P2P network as a function of the Zipf parameter for a single-region P2P setting consisting of a catalog of $m=60,00$ movies, $s=5000$ servers, and aggregate demand of $n=4000$ requests. The number of granted requests is computed from Eq. (4). The figure demonstrates that at low values of the Zipf parameter the system can grant about 50% of the requests; This is due to the fact that there are many esoteric movies and the system simply cannot hold all of them. In this case the P2P system will require a significant support from the central server. When the Zipf parameter is large, the system can handle almost all the requests since there are not many esoteric movies. For the sake of comparison, we also plot the performance of the proportional mean allocation [5] and of the RLB allocation [7]. As shown, both are not as efficient as the Max Percentile allocation. We repeat the analysis for a larger catalog of $m=100,000$ movies and observe similar results (not shown).

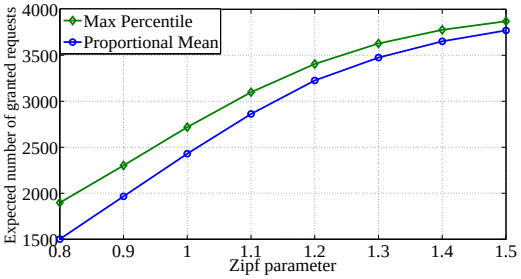

Fig. 4. Performance of single-region network with n=4,000, s=5,000, m=60,000.

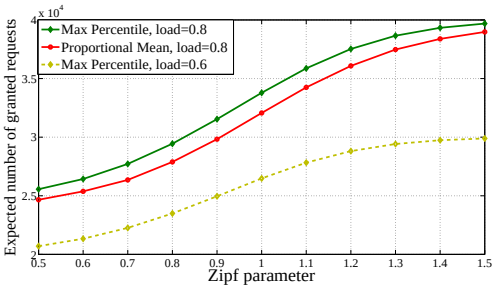

Fig. 5. Performance of a large-scale single-region network with s=50,000, m=60,000, n=40,000 (load 0.8) and 30,000 (load 0.6).

Figure 5 demonstrates the impact of the system load on performance (the number of granted requests) for a large-scale single-region P2P setting consisting of a catalog of $m=60,000$ movies and $s=50,000$ servers. The number of requests is $30,000$ and $40,000$, corresponding to loads of 0.6 and 0.8, respectively. As expected, we see that the number of granted requests increases with the number of submitted ones. We also show the performance of proportional and RLB allocations for the load of 0.8 and observe similar results to the smaller-scale network in Figure 4.

Max percentile and proportional mean. As shown in Figure 4 and 5, the performance gap between max percentile and proportional mean is largest when the Zipf parameter is small. This happens because the settings include large number of esoteric movies for which the variability of the demand is high, resulting in large inaccuracies of proportional mean (and RLB) allocations. The gap will be large in other scenarios as

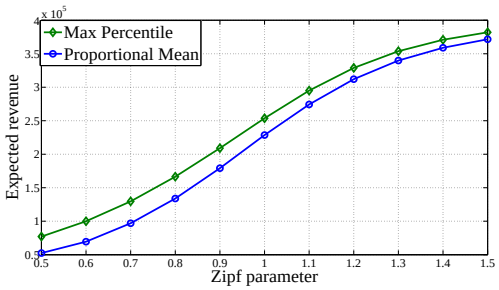

Fig. 6. Performance of a 20-region network with n=40,000, s=50,000, m=60,000, R_{glo}=1, R_{loc}=9.

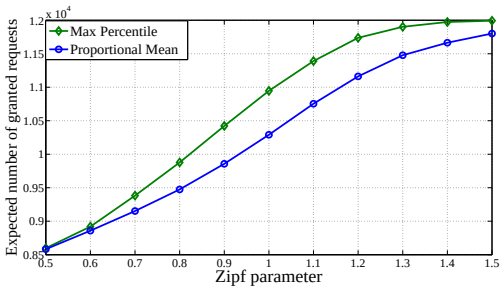

Fig. 7. Performance of single-region setting where each peer holds 5 movies and n=12,000, s=15,000, m=75,000.

well. For example, consider a 2-movie system with n peer-servers. The first movie has a deterministic demand of n requests and the second one has a stochastic demand of nk requests w.p $1/k$ and 0 requests w.p $1-1/k$. [5] Max percentile will allocate n replicas for the first movie, yielding maximum revenue of n (number of peers), while proportional mean will allocate $n/2$ replicas for each of the movies, yielding a smaller revenue of $n(1+1/k)/2$. Note that as k approaches infinity, the performance ratio approaches $1/2$.

Multi-Region. Next, we study the performance of a multi-region system. We consider a setting with $k = 20$ regions, local revenue of $R_{loc} = 9$, global revenue of $R_{glo} = 1$, and remaining parameters as before, $m = 60000$, $s = 50,000$ and $n = 40000$. Figure 6 shows the revenue computed by Eq. (8) as a function of the Zipf parameter. The revenue of Max-Percentile is compared to that of proportional mean when it is applied locally to each region. We observe similar behavior to the single-region case. In a multi-region setting, the number of locally served requests increases with the Zipf parameter. For large parameter values, majority of requests are served locally, and the gap between max-percentile and proportional is small. The gap is larger for small values of the parameter since many requests are served from remote regions.

We use **simulation** to validate our revenue-based model by measuring the revenue achieved by our placement algorithm under the multi-region setting above. To this end, we first compute an optimal allocation using the max-percentile algorithm, populating the peers. Then, we randomly generate demands according to Zipf distribution and apply optimal matching (Algorithm 1) 10 times to compute the expected revenue. Though not shown, we observe that the revenue function given by Eq (8) matches the measured one.

Storage capacity > 1. Up to this point, we considered peers storing a single movie. Next, we examine systems where peers can store multiple movies. To this end, we use the heuristic algorithm in Section IX. That is, the number of replicas in a region is determined by the max percentile algorithm operating on sc peers, where c is the peer's storage capacity; in our case $c = 5$. Then, replicas are sorted in decreasing order of popularity and placed on the individual peers of the region using an elevator-type placement. We use simulation to evaluate performance: we compute the heuristic allocation once and apply matching multiple times to derive mean revenue. We compute the proportional mean allocation in a similar fashion, and show the performance of both in Figure 7. When the Zipf parameter is large, the demand variability is low, and the proportional mean becomes a good predictor for the distribution's tail. Hence, both algorithms produce similar results. The performance gap is larger when the Zipf parameter is around 1. When the Zipf parameter is small, both algorithms face the same heuristic decision whether to place a copy of an esoteric movie, resulting in similar performance.

[5] This can represent the case where the demand depends on the review a movie is about to receive.

XI. Concluding Remarks

In this paper we formulated the assignment and replica placement problems of peer to peer video systems in multi-region systems. We presented an exact solution leading to finding the optimal assignment as well as the optimal allocation. The optimality was derived under the assumption of symmetric stochastic demands with arbitrary distributions. Algorithms for solving the problems were proposed; these are very efficient and can deal with networks consisting of millions of peers. The analysis provided in this work can be further extended to deal with k-level hierarchies. In an ongoing work, we explore algorithms for solving the problem under asymmetric loads.

References

[1] D. Qiu and R. Srikant, "Modeling and Performance Analysis of Bittorrent-like Peer-to-Peer Networks," in *ACM SIGCOMM*, Portland, OR, USA, August 2004.

[2] D. Wu, Y. Liu, and K. Ross, "Queuing Network Models for Multi-Channel P2P Live Streaming Systems," in *IEEE INFOCOM*, Rio de Jeniero, Brasil, April 2009.

[3] Y. Huang, T. Z. J. Fu, D. M. Chiu, J. C. S. Lui, and C. Huang, "Challenges, Design and Analysis of a Large-scale P2P-VoD System," in *ACM SIGCOMM*, Seattle, WA, USA, August 2008.

[4] V. Valancius, N. Laoutaris, L. Massoulie, C. Diot, and P. Rodriguez, "Greening the Internet with Nano Data Centers," in *ACM CoNext*, Rome, Italy, Dec 2009.

[5] S. Tewari and L. Kleinrock, "Proportional replication in peer-to-peer networks," in *IEEE INFOCOM*, Barcelona, Spain, April 2006.

[6] J. Kangasharju, K. W. Ross, and D. A. Turner, "Optimizing file availability in peer-to-peer content distribution," in *IEEE INFOCOM*, Anchorage, Alaska , USA, May 2007.

[7] Y. P. Zhou, T. Z. J. Fu, and D. M. Chiu, "Statistical modeling and analysis of p2p replication to support vod service," in *IEEE INFOCOM*, Orlando, FL , USA, July 2011.

[8] B. Tan and L. Massoulié, "Optimal content placement for peer-to-peer video-on-demand systems," in *IEEE INFOCOM*, Orlando, FL , USA, July 2011.

[9] M. S. Allen, B. Y. Zhao, and R.Wolski, "Deploying Video-on-Demand Services on Cable Networks," in *ICDCS*, Toronto, Canada, June 2007.

[10] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan, "Optimal content placement for a large-scale vod system," in *ACM CoNEXT*, Philadelphia, USA, Dec 2010.

[11] Y. Rochman, H. Levy, and E. Brosh, "Max Percentile Replication for Optimal Performance in Multi-Regional P2P VoD Systems- Technical Report. [Online]. Available: http://www.cs.tau.ac.il/~yuvalroc/publications/qest2012tech.pdf," 2012.

[12] "Netflix home page. http://www.netflix.com/," 2000.

[13] C. Ye and D. M. Chiu, "Peer-to-peer replication with preferences," in *InfoScale*, 2007, pp. 4:1–4:8.

[14] K. Chen and H. Shen, "Global optimization of file availability through replication for efficient file sharing in manets," in *Network Protocols (ICNP), 2011 19th IEEE International Conference on*, oct. 2011, pp. 226 –235.

[15] E. Cohen and S. Shenker, "Replication strategies in unstructured peer-to-peer networks," in *ACM SIGCOMM*, 2002, pp. 177–190.

[16] F. L. Presti, C. Petrioli, and C. Vicari, "Distributed dynamic replica placement and request redirection in content delivery networks," in *MASCOTS*. IEEE Computer Society, 2007, pp. 366–373.

[17] K. Suh, C. Diot, J. Kurose, L. Massoulie, C. Neumann, D. F. Towsley, and M. Varvello, "Push-to-peer video-on-demand system: Design and evaluation." *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 9, pp. 1706–1716, 2007.

[18] Y. Boufkhad, F. Mathieu, F. de Montgolfier, D. D. Perino, and L. Viennot, "Achievable Catalog Size in Peer-to-Peer Video-on-Demand Systems," in *IPTPS*, Tampa Bay, FL, USA, February 2008.

[19] J. Wu and B. Li, "Keep Cache Replacement Simple in Peer-Assisted VoD Systems," in *IEEE INFOCOM*, Rio de Janeiro, Brasil, April 2009.

[20] Y. Guo, K. Suh, J. F. Kurose, and D. F. Towsley, "P2cast: peer-to-peer patching scheme for vod service," in *WWW*, 2003, pp. 301–309.

[21] W. D. Brent, *Introduction to Graph Theory*, 2nd ed. Prentice Hall, 1999, ch. 3.

[22] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "Youtube traffic characterization: a view from the edge," in *ACM Internet Measurement Conference*, ser. IMC '07. New York, NY, USA: ACM, 2007, pp. 15–28. [Online]. Available: http://doi.acm.org/10.1145/1298306.1298310