Peer-Assisted VoD Prefetching in Double Auction Markets

Yuan Feng, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{*yfeng, bli*}*@eecg.toronto.edu*

Bo Li
Department of Computer Science
Hong Kong University of Science and Technology
bli@cs.ust.hk

***Abstract*—In peer-assisted video-on-demand (VoD) streaming systems, server bandwidth costs can be astronomical when the number of videos and peers scales up. Since peers are able to seek to an arbitrary point of playback in any video at any time, prefetching is often considered a desirable way to redistribute media content in the entire system so that less server bandwidth may be consumed. In this paper, we point out that the benefits of prefetching in peer-assisted VoD do not come without considerable upfront costs of bandwidth, and as such prefetching strategies should be carefully designed to remain beneficial, but practically carried out in a decentralized manner. We show how the challenge of minimizing server bandwidth is equivalent to maximizing the system-wide utility in the context of *double auction markets*, where each peer participates in a number of double auctions by bidding for and selling video segments. With simulations, we show that prefetching strategies based on such double auction markets are decentralized, and are effective in reducing the consumption of server bandwidth as well, as compared to existing alternative heuristics in the literature.**

I. Introduction

Peers in peer-assisted video-on-demand (VoD) systems are allowed the flexibility of requesting an arbitrary point for streaming playback in any video at any time. The design of peer-assisted VoD systems is much more challenging than live streaming, due to the temporal diversity of streaming requests and a dearth of opportunities sharing video segments among peers, often with prodigious quantities of videos being made available for streaming.

Since all requests that cannot be satisfied by peers will need to be sent to dedicated streaming servers, reducing bandwidth costs incurred on these servers has become the most important challenge in VoD systems. As examples, YouTube was estimated to have consumed $470 million a year, and Facebook was estimated to be spending $500,000 a month on bandwidth [1]. A natural intuition to offload servers is to perform *prefetching* in peer-assisted VoD systems: a large peer cache can be engaged on non-volatile storage, so that a peer is able to prefetch any segment from any video in the entire system, even one that it is not prepared to playback, effectively manipulating the popularity distribution of segments. The intuitive purpose of such prefetching is to improve the availability of video segments on peers, so that they are more likely to be served by peers rather than by dedicated servers, conserving server bandwidth.

In this paper, we would first like to argue that the benefits of prefetching do not come without substantial *upfront* costs of bandwidth. When prefetching is performed to reduce the consumption of server bandwidth, it also consumes upload bandwidth in the system to download video segments with the hope that they can be useful to other peers in the near-term future. Whether prefetching is beneficial in reducing server bandwidth costs hinges upon whether the prefetched segments brings marginally *more utility* than the cost of performing such prefetching. To our knowledge, existing prefetching heuristics in the literature have not yet considered such upfront bandwidth costs of prefetching.

While it is certainly our wish to design prefetching strategies that justify such upfront bandwidth costs, we have come to realize that such a design should not be yet another heuristic based upon intuition. While our objective in this paper is to design practical and decentralized prefetching strategies that justify their costs and reduce server bandwidth consumption, our design is based upon a more in-depth understanding of the challenge at hand, by first formulating the challenge of minimizing system-wide server bandwidth consumption as a centralized optimization problem, taking into account practical constraints of upload and download bandwidth availability, as well as local caching capacities.

Since practical prefetching decisions must be made in a decentralized fashion, we believe that peer-assisted VoD systems can be treated as trading markets. Segments are considered as *commodities* in these markets, with each associated with a price at each time. Every peer makes their local decisions by participating in these trading markets with their "bids" and "asks" for segments, behaving as both buyers and sellers to maximize their benefits. We model these trading markets as *double auction markets*, and prove that the problem of minimizing server bandwidth is equivalent to that of maximizing the system utility in these decentralized markets. Peer trading actions in these markets govern their prefetching decisions.

Our motivation of using double auction markets to solve the server bandwidth minimization problem is inspired by the power of markets arbitrating decisions of both buyers and sellers in a decentralized fashion. Auctions have the ability to determine which buyer should obtain the commodity from which seller, and to match uploading and downloading peers in the context of prefetching in peer-assisted VoD systems. All decisions being made in double auction markets are decentralized, which makes it simpler to incorporate them into the design of prefetching strategies.

The remainder of this paper is organized as follows. In Sec. II, we formulate the challenge of minimizing server

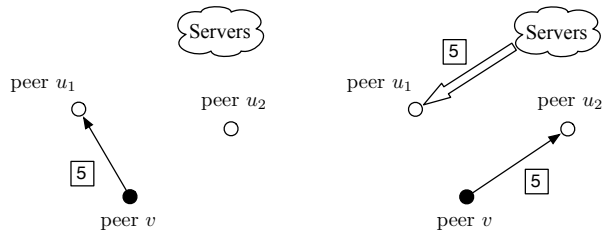
bandwidth in peer-assisted VoD systems. Sec. III transforms the formulation to problems in decentralized double auction markets through decomposition, and describes prefetching strategies governed by these auctions. In Sec. IV, we show the effectiveness of our prefetching strategies in reducing server bandwidth costs, as compared to alternative heuristics in the literature. We discuss related work and conclude the paper in Sec. V and Sec. VI, respectively.

II. Server Bandwidth Minimization in Peer-Assisted VoD Systems

We first present an example to explain why benefits of prefetching do not come without significant upfront costs of bandwidth.

A. Prefetching Is Not Free!

By utilizing idle upload bandwidth among peers to store segments that hopefully will be requested by other peers in the near future, prefetching strategies aim to provide a better playback quality in VoD systems. However, we must realize that their benefits may not come without costs. As an example, consider a simple scenario shown in Fig. 1.



(a) With no prefetching (b) With prefetching strategy *A*

Fig. 1. Prefetching is not free.

As shown in Fig. 1, peer v, which has segment 5 stored in its cache at this time, has the capacity of uploading only one segment. Without prefetching, the playback request for segment 5 from peer u_1 can be satisfied by peer v, shown in Fig. 1(a). In contrast, when a prefetching strategy *A* is involved, as demonstrated in Fig. 1(b), peer v's upload bandwidth is used to satisfy the prefetching request of segment 5 from peer u_2. As such, a subsequent playback request from peer u_1 has to be redirected to streaming servers if u_1 is sensitive to a delayed delivery. With the use of prefetching strategies, peers may be forced to upload segments to other peers who are not viewing them at the moment, or are even not planning to view them in the future. There are *risks* involved, in that less peer upload bandwidth can be devoted to serve active playback requests. If the prefetched segment can not be used to respond to future playback requests to save server bandwidth costs, consuming a higher amount of server bandwidth is unavoidable.

It now becomes clear that prefetching has to be performed with caution and due diligence. The probability of "hitting" right segments is of utmost importance. An ill-designed prefetching strategy may increase server bandwidth costs, rather than reducing them.

B. System Model

Our primary objective in this paper is to find out how prefetching strategies can be designed so that server bandwidth costs can be reduced, or even minimized. We first present the context of our discussions and a model of peer-assisted VoD systems.

Consider a peer-assisted VoD system with a total of M constant-length segments to be shared. Once a peer logs in, it contacts tracker servers and retrieves a set of neighbours, *i.e.*, peers who can upload and download segments among one another, according to certain neighbor selection algorithms. For each segment k, assume the access probability is p_k, and the sum of access probabilities p_k satisfies the condition $\sum_{k=1}^{M} p_k = 1$. We recognize that, in practice, there are numerous neighbor selection algorithms and it is not easy to accurately determine p_k. Since challenges involved in neighbor selection and modeling access probabilities are orthogonal to the scope of this paper and have been discussed in the literature (*e.g.*, [2], [3]), we assume that they are known *a priori*.

Though we consider a realistic VoD system supplying multiple video channels, we do not focus on the notion of videos or channels, and instead focus on a (potentially large) collection of *segments*, regardless of which video they belong to. A peer is not restricted to caching and serving segments from only the video it is actively viewing, *i.e.*, inter-session prefetching is allowed. Each peer u has a known upload bandwidth x_u, download bandwidth d_u, and a limited cache space c_u, all in terms of segments (rather than bytes). Note that, again, since there are dedicated servers to guarantee the users' viewing experiences, and peer bandwidth is used as additional resources to alleviate the server load as much as possible, and incentives are not considered as concerns in this paper. In other words, x_u is the reported available upload bandwidth of peer u.

To formalize recurring user participation that takes into account of heterogeneous viewing habits, we adopt the widely used ON-OFF model. There are a total of N participating peers, with each peer either ON (*i.e.*, present in the system) or OFF (*i.e.*, logged off) at time t. This behavior can be modeled as an alternating renewal precess $\{\rho_u(t)\}$ for each peer u:

$$\rho_u(t) = \begin{cases} 1 & \text{user } u \text{ is ON at time } t \\ 0 & \text{otherwise} \end{cases}, 1 \le u \le N.$$

With this notation, the number of peers in the network at t can be denoted by $N(t) = \sum_{u=1}^{N} \rho_u(t)$. To represent the fact that peers do not synchronize their arrivals or departures, and generally exhibit uncorrelated lifetime characteristics in peer-assisted VoD systems, we assume that peers behave independently, *i.e.*, processes $\{\rho_u(t)\}$ and $\{\rho_v(t)\}$ are independent for any $u \neq v$. Without loss of generality, we treat all ON-time and OFF-time processes as *i.i.d.* sets of variables. Assume for each process $\{\rho_u(t)\}$, its ON durations follow a specific distribution $F_u(x)$ and its OFF durations follow another distribution $G_u(x)$. In this paper, both ON and OFF

durations are considered to be exponentially distributed:

$$F_u(x) = 1 - e^{-\lambda_u x},$$

$$G_u(x) = 1 - e^{-\mu_u x},$$

where $1/\lambda_u$ and $1/\mu_u$ are the average lengths of time peer u stays ON and OFF, respectively.

In the case of inter-session prefetching, what a peer is uploading and downloading can be independent from its playback behavior. Consequently, as long as a peer stays ON, no matter which video it is viewing or switching to, its resources — more specifically, all cached segments and upload bandwidth — can always be contributed to the system. On the contrary, once a peer departs, neither its upload bandwidth nor its storage space can be used. Nevertheless, after its departure, the content previously downloaded and stored will remain in its non-volatile cache, and these segments can be shared again once this peer joins the system again.

C. Centralized Server Bandwidth Minimization

We emphasize that minimizing server bandwidth costs constitutes an objective with the highest priority to content providers. Since server bandwidth is consumed only when the supply of one segment from peers cannot meet the demand for the same segment, the problem of minimizing server bandwidth cost is equivalent to minimizing the "deficit" of peer bandwidth demand over supply. To be more precise, the demand for a particular segment k at time t is the total number of online peers that are asking to access segment k; and the supply of segment k from peers at time t is the sum of committed upload bandwidth from all online peers who are caching segment k at t. Ideally, we wish to find a feasible prefetching algorithm to optimally determine which segment should be prefetched from which peer, respecting peer capacity bounds, and minimizing server bandwidth costs.

Under the assumption that all peers possess global information of which set of segments other peers have stored in their caches, at time t, the centralized server bandwidth minimization problem can be formulated as a binary integer programming problem:

$$\min \quad \sum_{k=1}^{M}\left(\sum_{u=1}^{N}\rho_u(t)p_k - \sum_{u=1}^{N}\rho_u(t)I_u^k(t+1)x_u\right)$$

$$\text{s.t.} \quad \sum_{k=1}^{M}\sum_{u=1}^{N}T_{vu}^k(t) \le \rho_v(t)x_v,\ \forall v \tag{1}$$

$$\sum_{k=1}^{M}\sum_{v=1}^{N}T_{vu}^k(t) \le \rho_u(t)d_u,\ \forall u \tag{2}$$

$$\sum_{k=1}^{M}I_u^k(t+1) \le c_u,\ \forall u \tag{3}$$

$$I_u^k(t+1) = I_u^k(t) + \rho_u(t)\left(\sum_{v=1}^{N}T_{vu}^k(t) - D_u^k(t)\right),\ \forall u. \tag{4}$$

In this formulation, $T_{vu}^k(t)$ is the optimization variable, which is the binary variable to denote whether peer v is uploading segment k to peer u at time t. If it is uploading, $T_{vu}^k(t) = 1$; otherwise $T_{vu}^k(t) = 0$. Similarly, $I_u^k(t)$ and $D_u^k(t)$ are the binary variables to denote whether segment k is stored or deleted in peer u at time t, with $I_u^k(t) = 1$ indicating segment k is stored in peer u's cache at time t and $D_u^k(t) = 1$ representing peer u decides to replace segment k at time t. In such a centralized manner, $I_u^k(t)$ is supposed to be known at time t. $D_u^k(t)$ is regulated by the adopted cache replacement algorithm. Once the cache replacement algorithm is fixed, $D_u^k(t)$ is known at time t as well. The objective is to find the optimal prefetching strategy $T_{vu}^k(t)$ for all peers based on current information, so that the server bandwidth consumption at time $(t+1)$ is minimized.

Inequality (1) stands for the upload bandwidth capacity constraint; Inequality (2) represents the download bandwidth capacity constraint; and Inequality (3) denotes the storage capacity constraint. In contrast to previous work assuming that peer uploading bandwidth is the only bottleneck in video transmissions, our analysis is based on a more in-depth understanding of the challenge at hand. We formulate the problem taking into account practical constraints of both peer upload and download bandwidth availability, as well as local caching capacities. Eq. (4) is the update equation for the cache status indicator I_u^k. Whether segment k is stored in peer u at the next time slot should be determined by if other peers upload this segment to peer u at time t, along with if this segment is chosen to be replaced at time t.

Since the demand for each segment at time t, $(\sum_{u=1}^{N}\rho_u(t))p_k$, is known in our context, the minimization problem is equivalent to maximizing the latter part of the objective function under constraints (1)(2)(3). Replace $I_u^k(t+1)$ using update Eq. (4), so that Inequality (2) and (3) can be combined as:

$$\sum_{k=1}^{M}\sum_{v=1}^{N}T_{vu}^k(t) \le \min\{\rho_u(t)d_u, c_u - \sum_{k=1}^{M}\left(I_u^k(t) - D_u^k(t)\right)\}. \tag{5}$$

In subsequent analysis, we focus on decisions at a specific time. Therefore, time indices in above expressions are dropped, as we obtain the following optimization problem equivalent to the original one.

$$\max \quad \sum_{k}\sum_{u}\sum_{v}T_{vu}^k x_u \rho_u$$

$$\text{s.t.} \quad \sum_{k}\sum_{u}T_{vu}^k \le x'_v,\ \forall v$$

$$\sum_{k}\sum_{v}T_{vu}^k \le y'_u,\ \forall u$$

where $x'_v = \rho_v x_v$ and y'_u equals to the right part of Eq. (5).

The formulation is a comprehensive integer optimization problem, which is NP-hard in general. There exist many possible heuristics to solve it. For a general integer programming problem, we can relax it to a continuous linear optimization problem, use the primal-dual decomposition method, or use

dynamic programming [4]. For this particular problem, we can transform it into an equivalent minimum cost flow problem that can be solved in polynomial time [5]. Though through some of them we are able to obtain analytical lower bounds of server bandwidth cost in peer-assisted VoD streaming systems, the objective in this paper is to design prefetching strategies that can be implemented in a decentralized and light-weight fashion. Since we will later show that the utility maximization problem in double auction markets is equivalent to the dual of the server bandwidth minimization problem, we choose to adopt the primal-dual decomposition method here.

The corresponding dual problem may be formulated as follows:

$$\begin{aligned} \min \quad & \sum_v r_v x'_v + \sum_u q_u y'_u \\ \text{s.t.} \quad & r_v \geq 0, \forall v \\ & q_u \geq 0, \forall u. \end{aligned}$$

For some large constant C, replace $r_v = C - r'_v$, $q_u = C - q'_u$. The constant C can be dropped without affecting the solution. The dual problem can be rewritten as:

$$\begin{aligned} \max \quad & \sum_v r'_v x'_v + \sum_u q'_u y'_u \\ \text{s.t.} \quad & r'_v \geq 0, \forall v \\ & q'_u \geq 0, \forall u, \end{aligned} \tag{6}$$

where the dual variables r'_v and q'_u can be considered as the average profit per segment that peer v has gained by uploading no more than x'_v number of segments, and the average profit per segment that peer u has gained by downloading no more than y'_u number of segments, respectively. With these intuitive explanations, we believe that the optimization problem (6) is closely related to the social utility maximization problem in double auctions. We will prove that the two problems are equivalent in Sec. III.

III. Prefetching Strategies in Double Auction Markets

In this section, we show that double auctions can be a useful tool to solve the server bandwidth minimization problem in a decentralized manner.

A. The Concept of Double Auctions

The original auction algorithm solves the *assignment* problem, where a set of objects are to be assigned to a set of persons. The goal is to find a one-to-one mapping between persons and objects, such that the total utility gained by all persons is maximized over a given person-object utilization function. In double auction markets in particular, multiple buyers and sellers submit *bids* and *asks* simultaneously in each period, and a *trade* is made if a buyer's bid exceeds a seller's ask.

Without loss of generality, we focus on one double auction sub-market in an arbitrary round, *i.e.,* in a round that multiple holders and interested buyers trade for one particular commodity. The *reservation prices* are defined as the *true* highest price q_i^b a buyer i is willing to offer for the particular commodity, as well as the lowest price q_j^s a seller j is willing to accept. Their values are computed according to the participant's private evaluation of that commodity. The bid (ask) price b_i (s_j) is the *reported* price that buyer i (seller j) is willing to trade. The clearing price p_0 is the transaction price at which winning buyers and sellers trade. The utility gain for a buyer and a seller is denoted as $u_i(b_i)$ and $u_j(s_j)$, respectively. A buyer who buys her commodity valued at b and pays p to receive it will obtain a utility of $b-p$; a buyer who pays nothing and not receiving the commodity obtains zero utility. Similarly, a seller who sells her commodity valued at s and receiving a payment of p obtains a utility of $p - s$; otherwise zero is obtained if no trade and payment arise. If all participants bid truthfully, the social welfare maximization problem in a double auction sub-market with m buyers and n sellers can be formulated as follows:

$$\begin{aligned} \max \quad & \sum_{i=1}^{m} u_i(b_i) + \sum_{j=1}^{n} u_j(s_j) \\ \text{s.t.} \quad & x_i \in \{0,1\}, \forall i \\ & y_j \in \{0,1\}, \forall j, \end{aligned}$$

where $u_i(b_i) = x_i(b_i - p_0)$ and $u_j(s_j) = y_j(p_0 - s_j)$; x_i and y_j denote whether a buyer i or a seller j enters into a transaction or not, respectively.

***Theorem** 1:* The dual of the server bandwidth minimization problem in peer-assisted VoD streaming system is equivalent to the social utility maximization problem in double auction markets.

Proof: Consider the action of uploading segments as "selling," and downloading them as "buying." A segment is the commodity to be traded in a sub-market. Let $\mathcal{M}_v$ and $\mathcal{M}_u$ denote the set of sub-markets peer v joins as a seller and peer u joins as a buyer, respectively. The total utility gains of peer v as a seller can be represented as $\sum_{k\in\mathcal{M}_v} u_v^k(s_v)$, and the total utility gains of peer u as a buyer can be represented as $\sum_{k\in\mathcal{M}_u} u_u^k(b_u)$, where $u_v^k(s_v)$ and $u_u^k(b_u)$ denote the utility gains for peer v and peer u in trading segment k. Double auction mechanisms intend to maximize system-wide utility gains of all participants, which can be formulated as follows:

$$\begin{aligned} \max \quad & \sum_{v=1}^{N} \sum_{k\in\mathcal{M}_v} u_v^k(s_v) + \sum_{u=1}^{N} \sum_{k\in\mathcal{M}_u} u_u^k(b_u) \\ \text{s.t.} \quad & u_v^k(s_v) \geq 0, \forall v \\ & u_u^k(b_u) \geq 0, \forall u. \end{aligned}$$

Since a peer cannot upload/download segments more than its upload/download bandwidth, the number of sub-markets a peer joins as a seller/buyer cannot exceed the number of segments a peer can upload/download simultaneously. If we let all the peers participate in exactly the same number of sub-markets with this upper bound, *i.e.* $|\mathcal{M}_v| = x'_v$ and $|\mathcal{M}_u| = y'_u$, the optimization problem above is equivalent to

the following one:

$$\begin{aligned} \max \quad & \sum_{v=1}^{N} \frac{x'_v}{|\mathcal{M}_v|} \sum_{k\in\mathcal{M}_v} u_v^k(s_v) + \sum_{u=1}^{N} \frac{y'_u}{|\mathcal{M}_u|} \sum_{k\in\mathcal{M}_u} u_u^k(b_u) \\ \text{s.t.} \quad & u_v^k(s_v) \geq 0, \forall v \\ & u_u^k(b_u) \geq 0, \forall u. \end{aligned}$$

Now the underlying connection between our server bandwidth minimization problem and the double auction method is rather evident. Let

$$r'_v = \frac{1}{|\mathcal{M}_v|} \sum_{k\in\mathcal{M}_v} u_v^k(s_v),$$

$$q'_u = \frac{1}{|\mathcal{M}_u|} \sum_{k\in\mathcal{M}_u} u_u^k(b_u),$$

we shall see that the social utility maximization problem is equivalent to the dual of the server bandwidth minimization problem. ■

Based on the established connection between our problem and the double auction method, the primal variable T_{vu}^k can be viewed as an indicator showing whether peer v sells segment k to peer u, and the dual variables r'_v and q'_u can be treated as the average profit per segment for peer v to upload and peer u to download segments. Then $r'_v x'_v$ is the utility gain of peer v as a seller, and $q'_u y'_u$ is the utility gain of peer u as a buyer. This implies that the dual of the server bandwidth minimization problem can be solved using the mechanism of double auction markets.

B. Segment-Based Market Organization

We envision the existence of an online market place, where all participating peers behave as *buyers* and *sellers*; segments are treated as *commodities*, the basic trading unit in our design. The entire market consists of M sub-markets, with each of them trading one of the M segments. Every segment is associated with a price $\mathbf{P}_k$ at each time and an invisible currency is adopted. Uploading segments is considered as selling, and downloading is considered as buying. It is critical to note that it is not actually selling, but "duplicating," as the seller still owns that segment in its cache. However, due to the possible price degradation of segments incurred by duplication, risks exist for uploading segments to others.

From the system point of view, the auction among peers is a double auction with *multiple commodities*, referring to the large number of segments waiting to be traded in the system. If we divide the market into several sub-markets, where each sub-market only trade one segment, a particular segment in a sub-market may be interested by multiple peers, either buying or selling. That is to say, for each sub-market, it is a *single-commodity single-unit double auction*, *i.e.*, each agent wishes to buy (sell) one unit of the same item (segment), as no peer wishes to save the same segment repeatedly given her limited storage space. Peers can participate in multiple sub-markets to trade their holdings simultaneously. However, these single-commodity double auctions are inter-dependent with one another, due to the upload/download bandwidth and storage capacity constraints.

To decouple these inter-dependent double auctions, We impose the following constraints to restrict peer behavior:

▷ Each peer can only participate in $\min\{x_u, c_u\}$ number of auctions at each time as a seller. It prefers auctions trading $\min\{x_u, c_u\}$ number of segments that have the least risk.

▷ Each peer are restricted to join in $\min\{d_u, c_u\}$ number of auctions at each time as a buyer. It intends to join such auctions trading the most valuable $\min\{d_u, c_u\}$ segments that it does not have in its cache at the moment.

The rationale behind these assumptions are as follows. The number of sub-markets a peer can participate in is determined by the proof of **Theorem 1**, *i.e.*, it should be equal to the upper bound of the "available" upload and download bandwidth. It is reasonable that peers intend to join as many sub-markets as they can to gain more profit. However, the number of segments a peer uploads at each time is restricted by not only its upload bandwidth but also its local storage capacity, as it cannot upload segments more than the number it is able to store. Similarly, the number of segments a peer can download at each time is also limited by its download bandwidth and its storage capacity. Downloading segments more than its storage capacity will be a waste of bandwidth resource. By joining auctions trading segments of the least risk, *i.e.*, segments that have the least reduced price when duplicating them once, the profit gained by uploading can be maximized. Also, peers favor those valuable segments as they wish to maximize the total value stored in their limited caches. Based on the same objective, we adopt a simple but intuitively reasonable cache replacement algorithm: when the storage capacity is exceeded, peers will choose to delete the least valuable segment from their caches.

Under the restriction of aforementioned rules, single-commodity double auctions in sub-markets are independent from one another. Peers can be buyers and sellers at the same time in different sub-markets. These auctions are executed periodically and peers submit bids/asks simultaneously in each trading period. Such distributed double auctions can be illustrated by the example in Fig. 2, where peer v (the black dot in the figure), which has segment 5 stored in its cache, participates in sub-markets trading segment $3, 4, 5$ simultaneously. It serves as a seller in the sub-market trading segment 5, *i.e.*, uploading segment 5 to another peer, while as buyers in other two sub-markets, *i.e.*, downloading these segments from other peers.

C. Double Auctions in Prefetching

Based on segment-based markets, we now discuss the allocation and pricing strategy used in each of the sub-markets. For a given segment in a given trading period, suppose there are m buyers and n sellers willing to trade. Each buyer i reports the bid price as b_i, and each seller j asks for s_j for this segment.

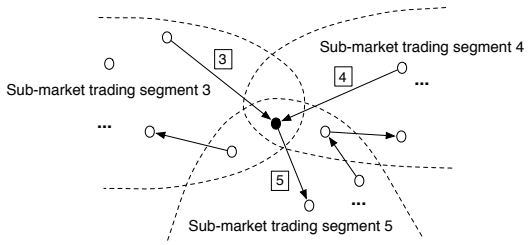


Fig. 2. Prefetching decisions in double auction markets.

Bids and asks of buyers and sellers. Let the price of a segment, $\mathbf{P}_k$, at a specific time be proportional to the current demand of this segment, and be inversely proportional to the availability of it in the system, *i.e.*, $\mathbf{P}_k = \hat{N}p_k/A_k$, where $\hat{N}p_k = \sum_{u\in\mathcal{N}_{ngh}} \rho_u p_k$ is the local demand for segment k, and $A_k = \sum_{u\in\mathcal{N}_{ngh}} \rho_u I_u^k x_u$ is the local segment availability in the system at the time. $\mathcal{N}_{ngh}$ is the set of neighbors of a certain peer. The intuition is that, segments with more requests and with scarcer availability in the system should have higher prices than others. Since peers only have segment availability of their neighbors by periodically exchanging buffer maps, peers compute the price of each segment by using their local information. As a consequence, it is possible for different peers to have different evaluations of the same segment. In addition, the local demand and local availability of a segment vary over time, leading to price fluctuation of all segments. Peers adjust their bidding strategies according to such price fluctuations, so that the number of each segment stored in the system is controlled dynamically and the diversity of segments is maintained.

Here, the bid price b_i for a buyer is the price of the segment computed by i's partial information; and the ask price s_j is the estimated loss by selling this segment, which is the locally computed reduced price of this segment by duplicating it once. To be precise, $s_j = \mathbf{P}_k - \hat{N}p_k/(A_k + \bar{x})$, where $\bar{x} = \frac{1}{\hat{N}}\sum_{u\in\mathcal{N}_{ngh}} \rho_u I_u^k x_u$. Note that the ask price is not the segment price but the risk of executing this trade, seeing that it is "duplicating" not "selling." What the seller loses is the risk, *i.e.*, price degradation, caused by increasing the segment availability in the system. As a result, sellers ask for prices trying to compensate for this loss.

At first glance, it may seem that there exists no benefit for peers to sell and upload segments. By uploading any segment, its price decreases due to the increased availability, leading to a decrease of the total value of all segments in a seller's cache. It seems that "duplicating" any segment will create a net loss for sellers. In fact, however, if all peers refuse to upload, segment prices will remain the same and none of them suffers value degradation in its cache. Yet if any one of them chooses to upload, all other peers will suffer, while the uploading peer will be adequately compensated by the buyer. This is similar to the classical Prisoner's Dilemma problem in game theory. It has been proved that the concern of each player to maximize her own payoff results in a unique equilibrium for this game. In our context, the rational choice is for every seller to upload, similar to players deciding to defect in the Prisoner's Dilemma problem.

The continuous double auction. Our objective is to find an economically robust double auction mechanism with high efficiency. The unique properties of peer-assisted VoD systems further impose requirements in designing the auction protocol. As we discussed in Sec. III-B, each peer chooses proper sub-markets to attend as either a buyer or a seller after it has completed downloading or uploading a segment. With asynchronous peers in real-world peer-assisted VoD systems, such downloading or uploading processes at different peers complete at different times. Since the computation of bids and asks occur after completing the previous downloading or uploading processes, they are asynchronous events as well. As a result, it is cumbersome to require that trades in the market is executed, synchronously.

As such, we propose to adopt the *continuous* double auction [6] in this paper, which is widely used in major foreign exchanges (FX) or stock exchanges (such as NASDAQ and NYSE). In the continuous double auction, there is usually a fixed-duration trading period, and buyers and sellers can submit bids and asks, respectively, at any time during the trading period and the market clears continuously. Specifically, the market partly clears whenever there is a match between a pair of bid and ask, *i.e.*, a trade is executed immediately when a bid exceeds an ask.

The continuous double auction fits our needs in peer-assisted VoD systems since its continuity caters to the asynchronously arrival of bids and asks. Since one peer trades with another peer once it finds a compatible pair of bid or ask, there is no global knowledge needed to efficiently allocate traded resources in a market. The efficiency can be achieved by local interactions of market participants, each of which possess only partial knowledge of the market condition, and no synchronization is required among peers. It has been shown that, with a continuous double auction, trading prices converge very quickly to the price computed as the intersection point of the true demand and supply curves [7], after only several trading periods, and the trading becomes almost fully efficient. Experimental results also show that the double auction is better from both resource's and user's perspectives, facilitating high resource utilization [8].

To make the continuous double auction more amenable to implementation in peer-assisted VoD streaming systems, we modify it by eliminating the existence of auctioneers. The role of auctioneer is much less emphasized in a continuous double auction. As the auctioneer only needs to collect bids and asks and informing players about them [9], the auctioneer can be readily replaced by *broadcasting* all bids and asks to every participating player in the same sub-market. Similar to the traditional *open outcry* mechanism used in a real-world stock exchange such as NYSE, where stock brokers transfer information about bids and asks through shouting and the use of hand signals, participating peers collectively act as an "auctioneer" as everyone has complete information of all bids and asks in the sub-market. Both the executed trade

and its trading price are monitored by all participating peers, which provides the trusted environment that an auctioneer traditionally provides.

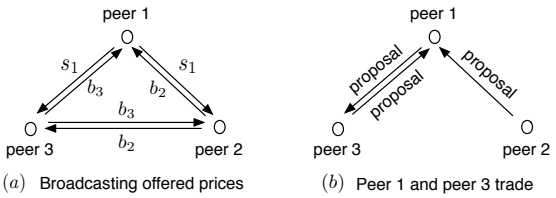
In addition, the partially connected peer topologies in peer-assisted VoD systems impose further restrictions that the trade can only be executed between neighbors, which means bids and asks of each peer only need to be broadcast to its neighbors. To a certain peer in a certain sub-market, only its participating neighbors' bids or asks have impact on its decision. In our double auctions, every peer announces the sub-market it intends to join, and the corresponding bid or ask it is going to report after it has finished downloading or uploading a segment. These announcements are broadcast among its neighbors only, which reduced the bandwidth overhead incurred. Decisions are made based solely on local broadcast announcements.

The continuous double auction in peer-assisted VoD systems is described in **Protocol 1**.

Protocol 1 *The prefetching strategy using continuous double auctions in peer-assisted VoD systems.*

Each peer announces a bid or an ask in one of the participating sub-markets and broadcast it to its neighbors.
When peer i with bid b_i as a buyer receives one ask s_j from its neighbor j, it does the following:
if $s_j \leq b_i$ **then**
  Peer i proposes to peer j that they can trade at the trading price $p_0 = \frac{1}{2}(b_i + s_j)$, and sends the proposal to j.
  if peer i has already received peer j's proposal **then**
    Peer i and j confirm the trade at the trading price p_0 and both of them exit this sub-market.
  else
    No trade is confirmed yet.
  end if
end if
When peer j with ask s_j as a seller receives one bid b_i from its neighbor i, it does the following:
if $b_i \geq s_j$ **then**
  Peer j proposes to peer i that they can trade at the trading price $p_0 = \frac{1}{2}(b_i + s_j)$, and sends the proposal to i.
  if peer j has already received peer i's proposal **then**
    Peer i and j confirm the trade at the trading price p_0 and both of them exit this sub-market.
  else
    No trade is confirmed yet.
  end if
end if

We now use a 3-player sub-market to illustrate our auction mechanism. Shown in Fig. 3, peer 1, 2 and 3 broadcast their offering prices in the sub-market, with peer 1 acting as a seller and peer 2 and 3 as buyers. Assume b_2 and b_3 are all qualified bids to trade, *i.e.*, $b_2 \geq s_1$ and $b_3 \geq s_1$. Let us assume that peer 1 receives b_3 first. It will send a proposal to trade with peer 3 immediately after receiving b_3. Once the proposal from peer 3 arrives at peer 1, which indicates peer 3 has received s_1 and is also willing to trade, a trade is executed between peer 1 and 3 at a trading price of $p_0 = \frac{1}{2}(s_1 + b_3)$. Note that the trading price is computed and monitored by both peer 1 and 3, which assures that no one can cheat in the trading process.



(a) Broadcasting offered prices (b) Peer 1 and peer 3 trade

Fig. 3. Prefetching decisions in double auction markets.

D. Required Economic Properties

Truthfulness, ex-post individual rationality and budget balance are critical properties required to design economically robust double auctions [10]. We now prove these properties in our double auctions.

***Theorem** 2:* The periodic double auction mechanism as described in Sec. III-C is economically robust.

Proof: We first prove the *truthfulness* property. A double auction is defined to be truthful if no buyer or seller can obtain a higher utility gain by cheating, *i.e.*, setting bid or ask not equal to its true utility value. Trustfulness is essential to resist market manipulation and ensure auction fairness and efficiency. It also eliminates the overhead of strategizing over other players. Suppose a buyer i with true utility value b_i reports $d_i \neq b_i$. Consider the outcomes of bidding d_i and b_i. It is not difficult to find out that the only possible scenario of obtaining higher utility gain by bidding d_i is that i wins by bidding d_i but loses by bidding b_i, which only happens when $d_i > b_i$. In this case, let $p_0(d_i) = \frac{1}{2}(d_i + s_j)$ and $p_0(b_i) = \frac{1}{2}(b_i + s_j)$ be the trading prices with bids d_i and b_i, respectively. We then have $p_0(d_i) \geq p_0(b_i) > b_i$. Recall that the utility gain of a buyer i by trading with seller j with bid d_i equals to $q_i^b - p_0(d_i) = b_i - p_0(d_i)$, which is negative in this case. Therefore, our double auction markets are truthful.

We then prove *ex-post individual rationality* and *budget balance*. A double auction is ex-post individually rational if the expected utility gain of any truthful participant in the auction is non-negative for all possible outcomes. Ex-post individual rationality ensures a non-negative utility gain if a participant reports its true utility value, and therefore provides incentives in the auction. The clearing price of a pair of buyer i and seller j is defined as $p_0 = \frac{1}{2}(b_i + s_j)$ and a trade is made only when $b_i \geq s_j$. By definition, all bids and asks are non-negative in every sub-market. So $b_i - p_0 = \frac{1}{2}(b_i - s_j) \geq 0$ and $p_0 - s_j = \frac{1}{2}(b_i - s_j) \geq 0$, which implies $p_0 \leq b_i$ and $p_0 \geq s_j$ for any winning buyer i and seller j. This proves that the double auction mechanism is ex-post individual rational.

A double auction is ex-post budget balanced if the mechanism's payoff is non-negative for all possible outcomes, which ensures that the auction will never run into deficit. Since the

total buying and selling quantities are guaranteed to match, and in each of the trading pairs, the payoff of a buyer is always the same as the revenue of a seller. Therefore, by the definition of budget balance, our double auctions are ex-post budget balanced. ■

E. Implementation Issues

Towards a practical implementation of double auction markets in realistic peer-assisted VoD systems, we briefly discuss a few implementation concerns.

▷ *Locally computed bids and asks:* The bids and asks reflect segment prices and estimated losses. All practical peer-assisted VoD streaming systems require periodic buffer map exchanges among neighboring peers. Traditionally, a peer's buffer map contains the availability of segments stored in its cache. If we allow its upload bandwidth information to be included in its buffer map, then peers will be able to receive sufficient local information to compute the necessary bids and asks through periodic buffer map exchanges.

▷ *Multiple qualified bids and asks:* Since a trade is executed when a bid is larger than or equal to an ask in our markets, there might be scenarios that a seller with ask s_j receives multiple bids or a buyer with bid b_i receives multiple asks that are qualified for successful trades. An intuitive question will be with whom should the seller or buyer to trade. This can be solved naturally by the *continuity* property. Since bids and asks are allowed to arrive asynchronously and the market is cleared partially as time goes by, the seller will always trade with the *first* arrived bid that is larger than or equal to s_j, and vice versa.

▷ *Peer dynamics:* In practical scenarios, caused by peer departures and varied interests, participants in each sub-market are different. Our auction-based prefetching strategy can readily adapt to such dynamics. When a new peer joins, it immediately initiates the prices and estimates reduced prices of segments based on its partial information, and then participates in auctions of the corresponding sub-markets. When a peer leaves, its bids and asks will not arrive to other peers, so that it will be naturally excluded from auctions.

IV. Experimental Evaluation

Our evaluation of the proposed auction-based prefetching strategy is based on its implementation in a time-slotted simulator using C++. For comparisons, we have also simulated the following alternative heuristics: (1) The near-sequential prefetching strategy [11]: it prefetches segments randomly after the playback point, while giving higher priorities to a set of segments right after playback; (2) The popularity-based prefetching strategy [12]: it determines which segment to prefetch according to the segment access probability and preferring those segments with high access probabilities; (3) The worst-seeded prefetching strategy [13]: it requires peers to prefetch the rarest segments in the system; (4) The price-based global prefetching strategy: it always asks for the most valuable segments under the assumption that each peer has global information of segment prices. Note that the last alternative is considered only for comparison purposes and is referred to as *optimal* henceforth.

We compare the performance of alternative heuristics to our auction-based protocol in terms of server bandwidth costs. To simulate heterogeneous peers, we set the upload and download bandwidth distribution as follows: 80% of peers have a uniformly random upload bandwidth within $3-7$ segments per time slot, representing the DSL users; 20% of peers have a uniformly random upload bandwidth within $5-10$ segments per time slot, representing Ethernet users. All peers have a uniformly random download bandwidth within $5-10$ segments per time slot. With the changing parameter specifically indicated, all experiments are conducted in a system consists of 1000 peers and 10000 segments, with a random graph topology. The local cache size of each peer is set to be uniformly random within $10-20$ segments. We assume that every peer joins the system with a full cache, caching segments randomly chosen from all segments. The default average time that peer stays ON or OFF in the system is 10 time slots.

Since our objective is to mitigate the server bandwidth cost, the main performance metric in our simulation is the average server cost over the total demand, *i.e.*, the portion of requests served by servers. We conduct our simulations for 100 time slots, and the average is computed from time slot 30 to 100, when relatively stable performance levels are achieved. We assume the segment popularity follows a stretched exponential distribution (SE), which has been discovered to be the access pattern for most media workloads in the Internet [14].

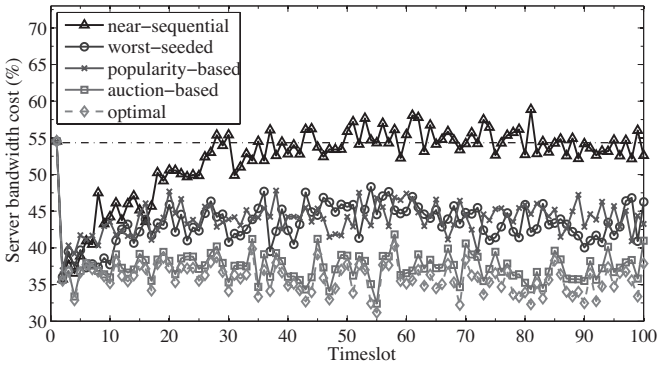


Fig. 4. Server bandwidth cost over time.

A. Overall Performance of the Auction-based Prefetching Strategy

First, we would like to show the overall performance of every prefetching strategy. We simulate each of them in the experiment setting with a larger scale — 1200 peers and 12000 segments for 100 time slots — and their respective performance over time is shown in Fig. 4. Due to the inherent randomness of stored segments at the beginning when all peers join the system, the server cost starts from around 55%, *i.e.*, 55% of the requests are satisfied by servers. From the figure, we can observe that all prefetching strategies are able to reduce the server cost to some extent. It can be observed that our

auction-based prefetching strategy defeats other alternatives by a substantial margin in terms of server bandwidth costs.

It is critical to point out that sometimes the server cost with the use of the near-sequential prefetching strategy becomes higher over time. This indicates that, rather than reducing the server bandwidth cost, the near-sequential prefetching strategy even *increases* the server bandwidth cost at times. Compared with the cost of preforming such a prefetching strategy, the benefits it brings are marginal or even negative. This reflects our intuition that the benefits of prefetching do not come without substantial upfront costs of bandwidth.

B. Performance with Different System Scales

We also show the effect of varying system scales on prefetching strategies. We first discuss the effect of varying the number of peers. Consider experiment settings with the number of peers varying from 800 to 1200, respectively. The comparison result with respect to the average server bandwidth cost of all prefetching strategies is shown in Fig. 5. We then turn our attention to different performance levels as the number of segments varies. In experiment settings with the number of segment ranging from 8000 to 12000, respectively, Fig. 6 illustrates the average ratio of server bandwidth consumption. We observe that the server load decreases when the number of peers increases, and increases with an increase of the number of segments. The curves in these two figures show reverse trends, since an increase of the peer population size is equivalent to a decrease of the number of segments. It is easy to see that our auction-based prefetching strategy shows consistently better performance as compared to alternative prefetching strategies.

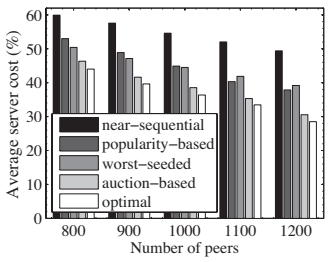


Fig. 5. Average server bandwidth cost vs. the number of peers.

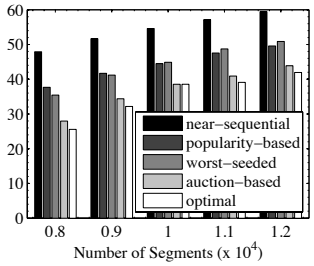


Fig. 6. Average server bandwidth cost vs. the number of segments.

C. Effect of Cache Size and Peer Churn

To investigate the impact of the cache size on the performance of average server bandwidth costs, peers in each experimental setting are equipped with different local caches, ranging from 10 segments to 30 segments. As shown in Fig. 7, we can observe a steep decrease of the average server bandwidth cost with an increase of cache sizes. It conforms with the intuition that with larger local caches, peers can store a higher number of segments so that more requests could be satisfied by peers. Though all prefetching strategies follow the same descending trend, the performance improvement of the auction-based strategy is more evident, which reflects its effectiveness on "hitting" proper segments.

We now proceed to evaluate the performance of our auction-based prefetching strategy in dynamic environments. With other parameters fixed in the experimental setting, we vary the average length of time peers stay ON or OFF in the system. We assume that all the peers' ON and OFF durations have the same exponential distribution. When the average length of time that peers stay ON/OFF in the system changes from 2 time slots to 25 time slots, as shown in Fig. 8, we can see a graceful decrease of average server bandwidth costs. When peers join and depart abruptly, the auction-based and the worst-seeded strategies preform equally well. However, when the average duration of peers staying in the system becomes longer, the performance gap between these two strategies gradually becomes more substantial. According to a recent measurement study [15], there is a large fraction of peers that tends to stay in the peer-assisted VoD system for over 15 minutes and nearly a half of them will stay for over 30 minutes, which implies that peer dynamics are rather mild in practice.

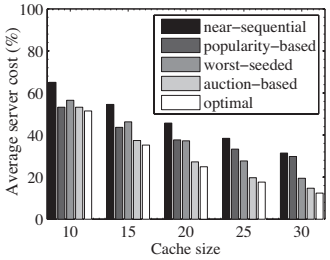


Fig. 7. Average server bandwidth cost vs. cache sizes (segments).

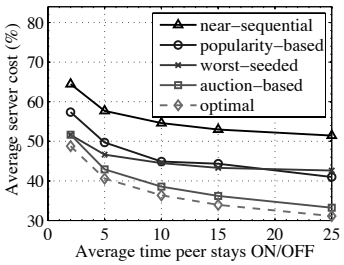


Fig. 8. Average server bandwidth cost vs. average time peer stays ON/OFF (time slots).

D. Overhead of the Continuous Double Auction

To investigate the communication overhead incurred by double auctions, we compute it as the ratio of bandwidth devoted to broadcasting bids and asks to neighbors over the total system traffic. Shown in Fig. 9, it is clear that the bandwidth overhead is reasonably low, with a consumption of only 1.5% of the total bandwidth usage. This is because the broadcast of bids and asks is only restricted to neighbors, and the number of neighbors of a peer is usually small, around $20 - 50$ in real-world peer-assisted VoD systems. Since the process of computing bids and asks is performed by each peer individually, we argue that the processing overhead raised by the double auction is reasonable.

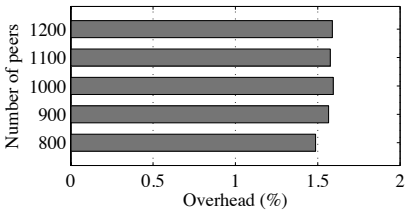


Fig. 9. Overhead of double auctions.

V. Related Work

Prefetching strategies in peer-assisted VoD streaming systems have been studied extensively. They allow peers to prefetch and store various portions of the streaming media ahead of their playing positions, which improves the playback experience. In addition to prefetching strategies we mentioned in Sec. IV, there are several other heuristics. Considering random seek, Zheng *et al.* [16] propose a prefetching strategy to minimize the random seeking distance by using quantization theory. He *et al.* [12] develop a prefetching strategy based on the estimated segment access probability to minimize the expected seeking delay at every viewing position.

To minimizing server bandwidth cost in peer-assisted VoD streaming systems, significant research efforts have been undertaken. In [17], by comparing cache prefetching, opportunistic cache update and a hybrid caching strategy, Kozat *et al.* find out that a utility maximization solution based on dynamic programming performs better in minimizing the average load on streaming servers. Wang *et al.* [18] design a distributed rate allocation algorithm to cut down on ISP-unfriendly traffic, thus reducing server load.

In addition, game theoretic analyses, such as auctions and bargaining, have been employed in P2P streaming. Hausheer *et al.* [19] present PeerMart, which introduces double auctions market in trading P2P services. It is applied in structured P2P overlay networks only, where peers in a leaf-set is clustered with one broker. Clustering peers is hard to be generalized to pull-based P2P overlay networks which are widely used in commercial peer-assisted streaming systems and it just shows effectiveness in small scale networks. Another proposal that uses auctions in peer-assisted VoD system is presented by Chu *et al.* [20], in which a protocol design with network coding is proposed. However, their objective is to minimize the aggregate link cost, and they discuss the case of a single video with the only peer upload bandwidth as a constraint.

Our work distinguishes from all existing works in the following three important aspects. *First*, we seek to design a prefetching strategy with the sole objective of minimizing server bandwidth cost. *Second*, rather than considering the single video case, or restricting inter-session content sharing, our prefetching strategy allows for multiple video streams and heterogeneous peers with respect to both upload and download constraints. *Third*, our prefetching strategy in double auction markets regulates not only which peer should prefetch which segment, but also from which serving peer the segment is to be served. Our modified continuous double auctions allow for the lack of a central auctioneer, which may be cumbersome to implement and cause substantial messaging overhead in real-world systems.

VI. Concluding Remarks

Our focus in this paper is to reduce server bandwidth costs in peer-assisted VoD systems when prefetching strategies are used. We have pointed out that the benefits of prefetching do not come without substantial upfront costs of bandwidth. In contrast to previously designed strategies, we show that prefetching strategies can be designed following principles of double auction markets, inheriting their decentralized and practical nature. We show that the dual of the server bandwidth minimization problem is equivalent to the system-wide utility maximization problem in double auction markets. Peer trading actions in these double auction markets govern their prefetching behavior. With double auction markets, our prefetching strategies regulate not only which segments should be uploaded or downloaded, but also to and from whom. As shown by our simulation results, auction-based prefetching strategies enjoy substantial improvement with respect to the reduction of server bandwidth costs.

References

[1] Do You Think Bandwidth Grows On Trees? [Online]. Available: http://www.slate.com/id/2216162.

[2] T. Syeda-Mahmood and D. Ponceleon, "Learning Video Browsing Behavior and Its Application in The Generation of Video Previews," in *Proc. ACM Mulitmedia*, vol. 9, 2001, pp. 119–128.

[3] X. Liao, H. Jin, Y. Liu, L. M. Ni, and D. Deng, "AnySee: Peer-to-Peer Live Streaming," in *Proc. IEEE INFOCOM*, 2006, pp. 1–10.

[4] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, Dec. 2002.

[5] M. Zhang, C. Chen, Y. Xiong, Q. Zhang, and S. Yang, "Optimizing the Throughput of Data-Driven based Streaming in Heterogeneous Overlay Network," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, pp. 97–110, 2009.

[6] S. Gjerstad and J. Dickhaut, "Price Formation in Double Auctions," EconWPA, Tech. Rep., 2001.

[7] V. L. Smith, "An Experimental Study of Competitive Market Behavior," *Journal of Political Economy*, vol. 70, p. 322, 1962.

[8] U. Kant and D. Grosu, "Double Auction Protocols for Resource Allocation in Grids," in *Proc. Int'l Conf. on Information Technology: Coding and Computing (ITCC)*, 2005, pp. 366–371.

[9] Z. Despotovic, J.-C. Usunier, and K. Aberer, "Towards Peer-to-Peer Double Auctioning," in *Proc. 37th Annual Hawaii Int'l Conf. on System Sciences*, Jan. 2004.

[10] P. Klemperer, "What Really Matters in Auction Design," *Journal of Economic Perspectives*, vol. 12, no. 1, pp. 169–189, 2002.

[11] Y. R. Choe, D. L. Schuff, J. M. Dyaberi, and V. S. Pai, "Improving VoD Server Efficiency with BitTorrent," in *Proc. ACM Mulitmedia*, 2007, pp. 117–126.

[12] Y. He, G. Shen, Y. Xiong, and L. Guan, "Optimal Prefetching Scheme in P2P VoD Applications With Guided Seeks," *IEEE Trans. Multimedia*, vol. 11, no. 1, pp. 138–151, Jan. 2009.

[13] S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, and P. Rodriguez, "Exploring VoD in P2P Swarming Systems," in *Proc. IEEE INFOCOM*, May 2007, pp. 2571–2575.

[14] L. Guo, E. Tan, S. Chen, Z. Xiao, and X. Zhang, "The Stretched Exponential Distribution of Internet Media Access Patterns," in *Proc. 27th ACM Symposium on Principles of Distributed Computing (PODC)*, 2008, pp. 283–294.

[15] 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 *Proc. ACM SIGCOMM*, 2008, pp. 375–388.

[16] C. Zheng, G. Shen, and S. Li, "Distributed Prefetching Scheme for Random Seek Support in Peer-to-Peer Streaming Applications," in *Proc. ACM Workshop on Advances in Peer-to-Peer Multimedia Streaming (P2PMM)*, 2005, pp. 29–38.

[17] U. Kozat, O. Harmanci, S. Kanumuri, M. Demircin, and M. Civanlar, "Peer Assisted Video Streaming With Supply-Demand-Based Cache Optimization," *IEEE Trans. Multimedia*, vol. 11, no. 3, pp. 494–508, April 2009.

[18] J. Wang, C. Huang, and J. Li, "On ISP-friendly Rate Allocation for Peer-Assisted VoD," in *Proc. ACM Mulitmedia*, 2008, pp. 279–288.

[19] D. Hausheer and B. Stiller, "PeerMart: The Technology for a Distributed Auction-based Market for Peer-to-Peer Services," in *Proc. IEEE Int'l Conf. on Communications (ICC)*, vol. 3, 2005, pp. 1583–1587.

[20] X. Chu, K. Zhao, Z. Li, and A. Mahanti, "Auction Based On-Demand P2P Min-Cost Media Streaming with Network Coding," *IEEE Trans. Parallel Distrib. Syst.*, 2009.