Stay or Go?
Participation in Under-Provisioned Video Streams

Dave Levin* Daniel Malter† Neil Spring* Bobby Bhattacharjee*
University of Maryland, *Computer Science Department, †Robert H. Smith School of Business
Email: *{dml,nspring,bobby}@cs.umd.edu, †dmalter@rhsmith.umd.edu

ABSTRACT

We consider the problem of choosing whether or not to participate in a video streaming session under the condition that there is insufficient upload capacity to serve all peers. We explore the connections between this problem and that of market entry. The fundamental difference arises from a lack of a centralized coordinator. In end-system multicast, peers' decisions must be based on local policy and observations. We find that careful selection of local policy—specifically, the pairwise trading policy among peers—assists in not only finding the equilibrium of market entry, but can also improve the equilibrium by increasing the number of entrants.

1. INTRODUCTION

Video streaming services, such as youtube.com and hulu.com, have become immensely popular in recent years. Their ability to scale to this demand is largely based on highly provisioned, centralized servers, which can be extremely expensive. Peer-to-peer video streaming offers a less expensive alternative, without requiring extensive pre-existing infrastructure or server capacity. Rather, the peers *viewing* the stream would be largely responsible for *providing* the stream.

While the common assumption is that each peer is able to give to the system as much as she requests from the system, this assumption may not hold in practice. In this paper, we study the problem of streaming video to a set of peers who collectively do not have enough upload capacity to allow all of them to view the stream.

The fundamental goal in this space is for there to be as many peers in the system as can be supported. We begin by looking at the problem of an under-provisioned system in a centralized setting (§2), and demonstrate similarities to market entry. This parallel lends insight toward an algorithm we present for determining the maximal subset of participants who can take part in a video streaming system.

We then turn to a decentralized setting, providing an algorithm for determining when a peer should leave a system they determine to be under-provisioned (§3), and studying what effect local trading policy has on the global system-wide properties (§4). The central hypothesis to this work is that local (decentralized) trading policies are sufficient for obtaining high levels of participation. We test this hypothesis via simulations (§5) of two allocation mechanisms: BitTorrent's unchoking algorithm [2], and the proportional share allocation mechanism [3].

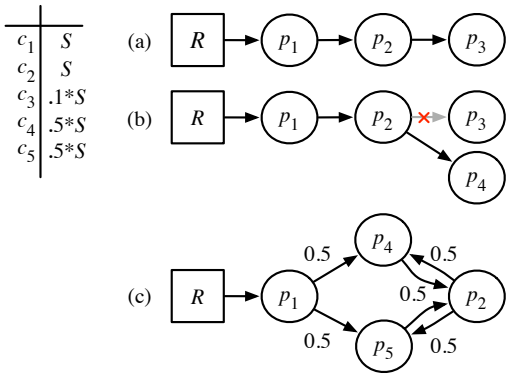

Figure 1: (a) Sample topology with stream rate S. Peer p_i has capacity c_i. (b) The server kicks out p_3 so that (c) more peers can join.

2. OPTIMAL, CENTRALIZED SOLUTIONS

We begin our study of participation in an under-provisioned system by first assuming a centralized coordinator. We demonstrate parallels between participating in a peer-to-peer system and market entry. Building off of classical market entry results, we sketch an optimal solution in this simplified setting.

2.1 A centralized system design

Consider the following strawman setting: new peers register with a centralized topology server, providing their IP address and port (so future peers can contact them), as well as the amount of capacity they are willing and able to provide to the system. The role of the centralized topology server would be to determine if a new user may join the system, and if so, to compute the topology that incorporates that user. We expect that the goals of such a service would be to optimize the number of participants in the stream, as well as the availability for the participating peers.

Deciding which peers can stay in the system can be viewed as an online scheduling problem. Consider the

example in Figure 1. Peers p_1 and p_2 both have the capacity of the stream rate, while other peers have less than the stream rate. Initially, in Figure 1(a), p_3 gains from the system without giving anything in return. Further, no peer with upload capacity less than $0.9S$ could obtain the stream rate without p_3 leaving the system.

As peers request to join the system, the centralized topology server would have to decide whether or not to let them enter. This decision can involve removing existing participants from the system. When peer p_4 requests to join the system in Figure 1(b), the server must decide whether or not to allow p_3 to stay. Keeping p_3 provides greater uptime to the participants. However, because p_4 offers greater upload capacity than p_3, allowing p_4 to join increases the number of peers who may join in the future. In the above example, p_3 loses his position in the swarm to p_4. As a result, p_5 in Figure 1(c) is able to obtain the stream; along with p_4, she can provide enough capacity to allow for four peers.

A challenge in this basic, centralized approach is in obtaining truthful capacity information from joining peers. In the example in Figure 1(a,b), for instance, how does the server know $c_3 < c_4$ if the peers do not need to upload to anyone? There are several potential solutions to this, such as uploading garbage data, as in BAR gossip [4], but this places strain on the topology server as many new peers join.

2.2 Participation as market entry

The question is thus how to design a protocol in which peers' donations are distributed in a way that assures continuing participation and efficiency as well as the distribution of the good among as many participants as possible. We derive an intuition for how we can solve this problem from the literature on market-entry games. Even though the setting of a market entry game is different, the logic behind the allocation we propose is very similar.

In the simplest case of a market entry game, a number of competitors n decides whether to enter or stay out of a market with fixed capacity c, where $n > c$. Let the payoff from entering the market be $\pi_i = c - \sum_i e_i$, where e_i is a binary indicator that takes 1 if actor i decides to enter the market. Assume for simplicity that the payoff from staying out of the market is 0. When the number of entrants into the market is greater than market capacity, the market is over-utilized and all entrants incur a loss. On the other hand, if too few competitors enter the market, the market is underutilized and all entrants incur a profit. The market is thus in equilibrium if there are as many entrants as there is market capacity.

This special, symmetric case of a market entry game can be generalized. Selten and Güth [8] derive theoretical propositions for equilibrium point selection when entering the market is costly and when costs are asymmetric. When the potential costs of entering the market are ordered among the competitors, then the equilibrium point selects market entrants by the order of the entry costs. In other words, the competitors with low entry costs enter the market to the point where the market is used to efficiency, whereas the competitors with higher entry costs stay out.

This equilibrium point selection has, in our view, important, natural implications for the design of participatory systems. Accepting into a system the peers who incur lower marginal cost appears to be a natural design principle. In the remainder of this section, we use this design principle as a basis for an optimal, centralized algorithm. We then use this optimal allocation as the baseline for comparing the decentralized protocols that we present in §4.

2.3 Finding the maximal participants

The market entry game's Nash equilibrium provides a template for determining the maximal number of participants in a video streaming system. We begin with the following observation:

OBSERVATION 1. *If a video streaming system consists of k peers, each of these peers can obtain the stream rate only if the sum upload capacity is at least $(k-1)S$.*

Ahlswede et al [1] proved that, with network coding, this condition is not only necessary but sufficient. Using this result, we can now answer: what is the maximal subset of peers who can obtain the video stream?

We present in Algorithm 1 a method for computing the maximal set of participants given their upload capacities. This algorithm parallels the market entry Nash equilibrium; peers are sorted by their upload capacities, and included in the system one by one until system capacity is reached.

Algorithm 1 Maximal participation determination

Peers p_i has upload capacity c_i, $i = 1, \ldots, N$.

1. Sort $\{c_i\}_{i=1,\ldots,N}$ in decreasing order. Suppose $c_1 \geq c_2 \geq \cdots \geq c_N$.
2. for $i = 1, \ldots, N$:
   (a) If adding p_i to the set of participants would decrease the sum capacity to less than $(i-1)S$, then return; add no more participants to the system.
   (b) Otherwise, add p_i to the set of participants.

We believe that this algorithm can be extended to finding the topology by which to achieve the stream rate for all permitted participants, but investigating such topologies is beyond the scope of this paper.

Although Algorithm 1 finds the maximal number of participants, it does not impose any fairness constraints.

We return to this point in the context of the decentralized strategies that we consider in the remainder of the paper. Determining the maximal set of participants who can be arranged in such a way as to ensure resilience to free-riding is an area of future work.

3. WHEN TO LEAVE?

In the remainder of this paper, we consider only decentralized solutions. The first difficulty in moving away from a solution by fiat is in determining who can participate in the network; there is no centralized coordinator with perfect information who can inform a peer whether or not they will be able to view the video stream. Rather, peers must determine via their own local observations whether or not they are able to view the stream.

3.1 A priori information is not enough

Not all peers can base their decisions of whether or not to join the system on their upload capacity alone. Figure 1 demonstrates this; p_3 was able to join the system in configuration (a), but would not be able to join in configuration (c). Certainly, it should hold that peers with upload capacity no less than the stream rate can obtain the stream rate from the system. However, there may not be a clear cut-off capacity, such that any peer with at least this capacity can be guaranteed to obtain the stream rate.

3.2 Waiting for others to leave first

Peers must enter the system in order to determine if they have sufficient capacity to gain from it. This introduces a challenge not present in classic market entry games: peers must decide not only *whether* to leave the system, but *when* to leave the system. Certainly, a peer must remain in the system long enough to get an accurate measure of what her steady-state download rate would be. A peer also has incentive to stay in the system in the hopes that others will leave, thereby freeing up capacity for her.

We focus on this problem of avoiding mass, unnecessary departure from the system. Just as 802.11 attempts to avoid multiple hosts sending at the same time, we attempt to avoid too many hosts leaving the stream at the same time. Suppose a new peer were to join the video streaming system in Figure 1(c) such that one, but not both, of p_4 and p_5 would have to leave. As both peers are affected by the new peer's entry, they may both begin obtaining less than the stream rate at the same time. A deterministic protocol for leaving—for instance, leaving after not obtaining the stream for some fixed F rounds—would result in both p_4 and p_5 leaving at the same time. The same effect is prevalent also when multiple peers join at the same time.

We present in Algorithm 2 a randomized protocol for selecting when to depart a system. Our intuition is that peers who are "close" to obtaining the stream rate have greater expectation to receive it if other peers leave, and therefore stay in the system with greater probability.

Algorithm 2 System departure.

If p has not obtained the stream rate (and this algorithm has not been called) in the previous F rounds, leave with probability

$$1 - \frac{\text{Average received in the last } F \text{ rounds}}{\text{Stream rate}} \quad (1)$$

We considered also incorporating a peer's capacity in their decision to leave, the goal being for higher-capacity peers to wait longer before leaving. However, this presupposes knowledge of a distribution of capacities; to be properly formulated, a peer would have to take into account the expected number of peers who have greater capacity than she. For example, a peer may expect that she has very low capacity, when, in comparison, she has the greatest capacity amongst all peers not yet obtaining the stream rate. We expect that download rates are a more accurate indicator of a peer's relative contribution to the system, and limit our algorithm to incorporate only this.

4. LOCAL TRADING STRATEGIES

Our hypothesis is that *global, system-wide optimizations can be obtained by local, selfish strategies.* In this section, we review two local strategies that have received attention within the context of the BitTorrent file swarming system [2]. We discuss how well we expect them to apply in an under-provisioned multicast system, and evaluate them experimentally in Section 5.

We assume a mesh-based system, wherein peers connect to a random set of neighbors, some of which may be the streaming server(s). Mesh-based streaming systems have received attention as a more resilient alternative to tree-based systems, although trees are typically more efficient. We believe that mesh-based streaming systems offer greater flexibility in peers' local policies; peers in a mesh have more freedom in terms of with whom and how much they trade, while a tree by definition imposes a more rigid topology.

4.1 BitTorrent unchoker

BitTorrent [2] is a decentralized file swarming system that breaks a large file into pieces, and establishes a mesh of peers who trade these pieces with one another. There have been various studies into using BitTorrent as a video streaming system [5–7]. We do not consider the broader systems design issues in this paper, and focus only on the local allocation policy, referred to as BitTorrent's *unchoking* algorithm.

We present BitTorrent's unchoking algorithm in Algorithm 3. BitTorrent peers reward their top k (typ-

ically 4) contributors, as well as one peer at random, with equal shares bandwidth.

Algorithm 3 BitTorrent's unchoking algorithm.

Run at peer p, with capacity c_p, at round t:

1. In round $t-1$, peer q_i uploaded $b_i^p(t-1)$ to p, for $i = 1, \ldots, N$.
2. If q_i was one of the top k contributors from the prior round, $b_p^i(t) = c_p/(k+1)$.
3. Choose another peer q_o at random and *optimistically unchoke*: $b_p^o(t) = c_p/(k+1)$.

Unfortunately, the mechanism to allocate the receiver's resources to a limited number of donors by an equal sharing rule has obvious design flaws that can result in an inefficient set of system entrants. Assume a closed system of six participants in a network and consider a receiver who requires resources of 1 to obtain the video stream. Suppose the receiver obtains contributions from five donors (say, 0.30, 0.25, 0.20, 0.15, 0.1). With BitTorrent's unchoker, the fifth donor (0.1) will not be rewarded by the receiver. The fifth donor has thus the following strategies at hand: (1) exit and do not contribute to the receiver in the next round, (2) increase the contribution to the receiver to end up among the first four places (i.e. place a higher bid), or (3) keep the bid constant or even reduce the bid in the hope that one of the current bidders with a higher bid ceases donating to this receiver. Obviously, the first strategy poses a serious threat to the system. If the fifth donor stops donating to the focal receiver, the receiver will be unable to procure the good because the sum of the donations to him amounts to only 0.9. It is easily conceivable that such a system can fail when resources among participants are scarce and asymmetrically distributed so that some network participants are net contributors, whereas other network participants are net receivers.

4.2 Proportional share allocation

The proportional share auction has recently received attention in the context of file swarming [3, 9]. It is trivially fair, and has been demonstrated to be resilient to a wide range of strategic attacks [3]. We present it in Algorithm 4.

Algorithm 4 Proportional share unchoking algorithm.

Run at peer p, with capacity c_p, at round t:

1. In round $t-1$, peer q_i uploaded $b_i^p(t-1)$ to p, for $i = 1, \ldots, N$.
2. $b_p^i(t) = c_p \cdot b_i^p(t-1) / \sum_j b_j^p(t-1)$

In addition to providing greater resilience to selfish participants [3], we expect that proportional share can yield high levels of participation, for two main reasons.

First, with proportional share, the more a peer gives, the more that peer gets. This fairness property does not necessarily ensure that a peer will get as much as she gives. However, it exhibits similar behavior to the maximal participant algorithm (Alg. 1); via proportional share, peers with greater capacity stand to gain more of the stream rate. Further, with proportional share, peers have *incentive* to give as much of their capacity as is necessary to obtain the stream rate.

Second, proportional share yields a clear signal of whether a peer will be able to obtain the stream rate or not. BitTorrent's 1/4-or-nothing unchoking algorithm makes it more difficult to anticipate how much a neighbor may give in future rounds. Conversely, proportional share is more continuous by nature, allowing a peer to more accurately detect a decrease in contribution from a neighbor. As such, we expect that low-capacity peers will realize quickly and clearly that they are unable to obtain the stream rate, while higher-capacity peers can observe when they are close to the stream rate, and therefore out-wait other peers, according to Algorithm 2.

5. SIMULATION STUDY

We study via simulation whether local allocation policy can yield market efficiency in under-provisioned settings. We compare the equilibria of the two local policies presented in §4 to the desirable outcomes discussed in §2.

5.1 Simulation details

In our simulation, all peers join at the same time, and are provided an initial contribution level for every other peer. Peers then run either a BitTorrent-like top-four allocation, or a proportional share allocation mechanism. We do not consider mixes of clients; in a given run, all peers run the same allocation mechanism. Peers choose to leave the system according to Algorithm 2, waiting for 10 rounds before probabilistically choosing whether to leave.

The primary goal of our simulations is to determine under what conditions of upload capacities a given local-policy-based allocation mechanism achieves high levels of participation. Of course, the set of feasible participants in a system depends on the system-wide distribution of upload capacities. Thus, peers in our simulations have different upload capacities, and the fundamental independent variable is the distribution of upload capacities.

5.2 Who stays and who goes?

We first study who is allowed entry into the system under varying allocation strategies. To understand this, we plot the inter-peer allocations at equilibrium for Bit-

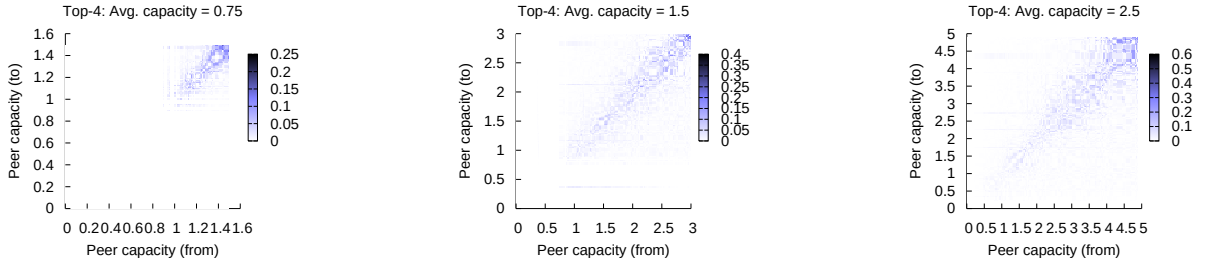

Figure 2: Clustering within a stream of top-4 clients.

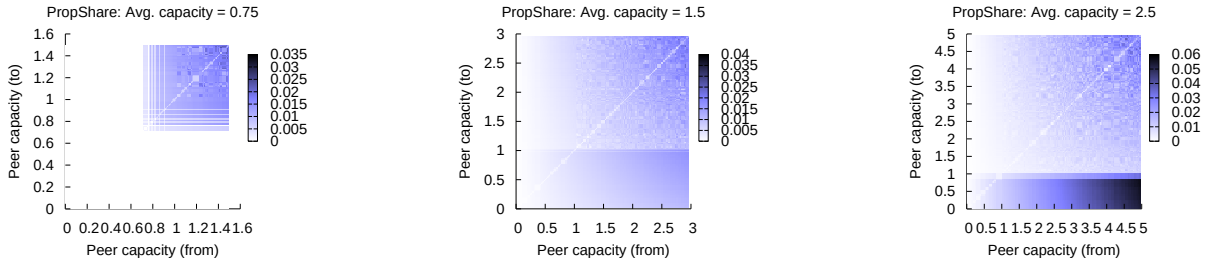

Figure 3: Clustering within a stream of proportional share clients.

Torrent's top-four (Figure 2) and proportional share (Figure 3). In these figures, a point at (x, y) denotes how much a peer with capacity x donated to a peer with capacity y, the amount of which is represented by the intensity of the shading. Peers who neither receive nor download in these figures have left the system.

With proportional share, all peers with capacity greater than or equal to the stream rate obtain the stream rate. Additionally, there are some peers with capacity less than the stream rate who also are able to remain in the system. As demonstrated in the series of plots in Figure 2, the number of lower-capacity peers who obtain the stream rate increases with excess market capacity. Further, these figures show that there is not a clear "cut-off capacity" that distinguishes between peers who may and may not obtain the stream. Rather, for low-capacity peers (who require extensive market excess), it is their random set of neighbors and initial contribution levels that determine whether or not they are able to obtain the stream.

BitTorrent's top-four unchoking algorithm does not yield as favorable results. While peers with greater capacity are more likely to obtain the stream rate, no peer appears to be guaranteed access to the system. Rather, there are peers with significantly more capacity than the stream rate who do not obtain the stream, as well as peers with considerably less capacity who do obtain the stream.

5.3 Comparison with maximal participants

We next study how close to the maximal number of participants these two local allocation algorithms are able to obtain. Recall that the number of peers able to enter the system increases with greater excess market capacity. We therefore plot in Figure 4 the fraction of the maximal participants (as computed by Algorithm 1) with varying average system-wide capacity.

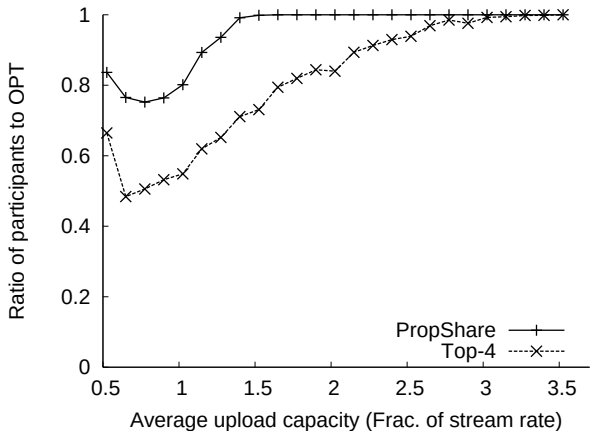

Figure 4: Number of system participants compared to the maximal amount from Algorithm 1.

Proportional share obtains nearly the maximal number of participants when the average system-wide capacity is 25% greater than the stream rate. The BitTorrent unchoking algorithm requires more than double this amount of capacity to obtain the same fraction of the maximal amount. This result demonstrates the extent to which the choice of a local allocation mechanism can affect the global properties of system participation.

5.4 When do peers leave?

We close our evaluation study by investigating when peers leave the system. As indicated in §3, low-capacity

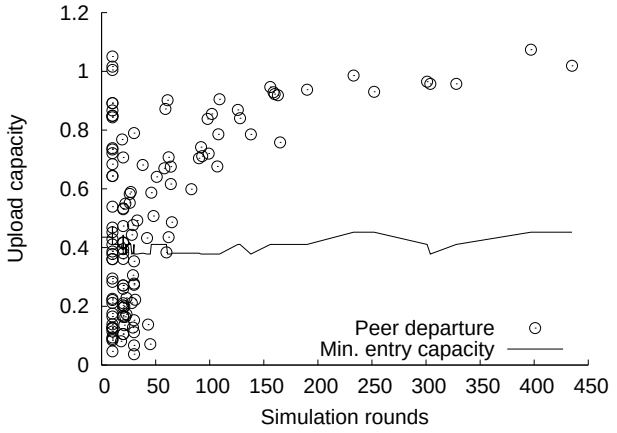

Figure 5: Order of departure from a stream of BitTorrent clients. Average capacity is 0.75

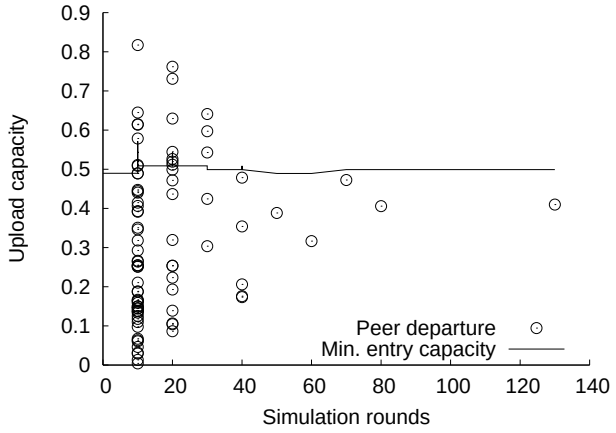

Figure 6: Order of exit from a PropShare stream. Average capacity is 0.75.

peers would leave first, allowing higher-capacity peers the opportunity to benefit from excess market capacity, while providing more to the system in return. We plot which peers leave, identified by their upload capacity, over the course of a simulation run for both BitTorrent (Fig. 5) and proportional share (Fig. 6) clients. Both clients use Algorithm 2 to determine if and when to leave.

Note first the distribution of the dots in Figs. 5 and 6. As in our other results, no proportional share clients with capacity greater than the stream rate leave the system. For both clients, the majority of low-capacity peers leave first, and higher-capacity peers typically leave later. This is particularly evident in the case of the BitTorrent client, perhaps due to the fact that there are more high-capacity peers leaving the system.

The lines in Figures 5 and 6 represent the minimal capacity necessary to participate in the system, as determined by Algorithm 1. Recall that this algorithm sorts the remaining peers in the system and adds them to the maximal participation set until the system is saturated. We run this algorithm every time a peer leaves the system, and plot the capacity of the last peer added to the maximal participation set. We emphasize that we determined the maximal participants only to calculate this capacity value; it did not affect the simulated behavior in any way.

Ideally, no peers above this line would ever leave the system, and all peers with capacity below this line would; this would precisely yield the maximal allocation. In other words, departures below the line are desirable for achieving a large number of system participants, while departures above the line are undesirable.

With these minimal capacities in mind, a comparison between the proportional share and BitTorrent clients becomes clearer. Few proportional share peers above the minimal capacity line leave after a short period of time, while the vast majority of BitTorrent departures are of peers who ought not have needed to leave.

6. CONCLUSION

We have considered the problem of determining the optimal number of participants in an under-provisioned system. In the centralized setting, we demonstrated parallels to the problem of market entry, which lent insight into the formulation of an algorithm for determining the maximal participation set. In the context of decentralized systems, proposed the hypothesis that local-policy allocation mechanisms can achieve greater amounts of participation. Our simulation evaluation supports this hypothesis, and demonstrates that the choice of an allocation mechanism can have significant effect on the resulting number of viable participants.

7. REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, 2000.

[2] B. Cohen. Incentives build robustness in BitTorrent. In *P2PEcon*, 2003.

[3] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee. BitTorrent is an auction: Analyzing and improving BitTorrent's incentives. In *SIGCOMM*, 2008.

[4] H. C. Li, A. Clement, E. L. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin. BAR gossip. In *OSDI*, 2006.

[5] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. E. Mohr. Chainsaw: Eliminating trees from overlay multicast. In *IPTPS*, 2005.

[6] V. Pai and A. E. Mohr. Improving robustness of peer-to-peer streaming with incentives. In *NetEcon*, 2006.

[7] M. Piatek, C. Dixon, A. Krishnamurthy, and T. Anderson. LiveSwarms: Adapting BitTorrent for end host multicast. Technical report, UW Tech. Report TR-2006-11-01, 2006.

[8] R. Selten and W. Güth. Equilibrium point selection in a class of market entry games. In *Games, Economic Dynamic, and Time Series Analysis - A symposium in memoriam Oskar Morgenstern*, pages 101–116. Vienna and Wrzburg: Physica-Verlag, 1982.

[9] F. Wu and L. Zhang. Proportional response dynamics leads to market equilibrium. In *STOC*, 2007.