ATTACK-RESISTANT COOPERATION STRATEGIES IN P2P LIVE STREAMING SOCIAL NETWORKS

W. Sabrina Lin[*], *H. Vicky Zhao*[†] *and K. J. Ray Liu*[*]

[*] ECE Dept., University of Maryland, College Park, MD 20742 USA
[†] ECE Dept., University of Alberta, Edmonton, AB T6G 2V4 Canada

ABSTRACT

Multimedia social networks have become an emerging research area, in which analysis and modeling of the behavior of users who share multimedia are of ample importance in understanding the impact of human dynamics on multimedia systems. In peer-to-peer live-streaming social networks, users cooperate with each other to provide a distributed, highly scalable and robust platform for live streaming applications. However, every user wishes to use as much bandwidth as possible to receive a high-quality video, while full cooperation cannot be guaranteed. This paper proposes a game-theoretic framework to model user behavior and designs incentive-based strategies to stimulate user cooperation in peer-to-peer live streaming. We first analyze the Nash equilibrium and the Pareto optimality of 2-person game and then extend to multiuser case. We also take into consideration selfish users' cheating behavior and malicious users' attacking behavior. Both our analytical and simulation results show that the proposed strategies can effectively stimulate user cooperation, achieve cheat free, attack resistance and help to provide reliable services.

1. INTRODUCTION

With the explosive growing of network, multimedia signal processing, and communication technologies, over millions of users share the multimedia contents over the internet, thus, those users form huge social networks over the world. These multimedia social networks have lots of users and interact with each other without geographical restrictions, which raises a crucial issue of understanding how the users influence each others' behaviors and analyzing the dynamics. Although human-to-human dynamics is an area with growing importance, human factor seldom appeared in signal processing analysis. These kinds of investigations also help to both offer more secure and personalized services and design multimedia and networking systems.

Peer-to-Peer (P2P) live streaming network [1], is one of the biggest multimedia social networks on the internet, consisting of self-organizing, distributed systems, with no centralized authority or infrastructure. Every user in a P2P live streaming social network wants to watch a live program over the internet at the same time, and the network relies on voluntary resource contributions by individual peers. Therefore, it's critical to analyze the users' behavior and provide both incentives and optimal strategies for cooperation.

Since the users in a P2P live streaming social network are strategic, they are likely to manipulate any incentive system and every rational user in the network is selfish in the sense that everyone wants to has higher video quality as possible, even with cheating. As a result, game theory [2] is a proper tool for modeling the interaction of peers to analyze the optimal and cheat-proof cooperation strategies.

There are some prior work related to the P2P live streaming social network analysis: the game theoretic framework for P2P file sharing is investigated in [3], and [4] provides a mechanism for cooperation in P2P live streaming networks assuming every one in the P2P live streaming social network is willing to cooperate, while [5] provide a reputation-based mechanism in P2P live streaming network. However, the cheat-proof and optimal cooperation strategy of P2P live streaming social network is not studied.

In this paper, we'll focus on designing cooperation stimulation strategies for P2P live streaming social networks under a game theoretic framework. We first study a two-player game and investigate the Nash equilibria. Since this game usually has multiple equilibria, we then investigated how to apply extra optimality criteria, such as Pareto optimality, fairness, and cheat-proofing, to further refine the obtained Nash equilibrium solutions. The goal of this analysis is to stimulate each pair of user in the P2P live streaming game to start cooperate with each other safely.

The rest of this paper is organized as follows: Section **??** introduces the P2P live streaming system and the game-theoretical framework. Section **??** studies the attack-resistant cooperation equilibria of all the non-malicious users in the live-streaming game. In Section **??** extensive simulations have been conducted to evaluate the performance of the proposed strategies. Finally, Section **??** concludes this paper.

The authors can be reached at wylin@eng.umd.edu, vzhao@ece.ualberta.ca, and kjrliu@eng.umd.edu.

2. SYSTEM MODEL

In this section, we first describe how two users in a P2P live streaming social network cooperate with each other. We then define the payoff function and introduce the game-theoretic modeling of user dynamics.

2.1. P2P Live Streaming Cooperation Model

In a delivery architecture for live video streaming, a video bit stream is divided into media chunks of M bits, and all the chunks are available at an original server. When a peer wants to view the video, he/she first obtains a list of peers currently watching the video, together with information about the availability of each chunk in others' buffers. At the beginning of each round, every user sends a request either to one of his/her peers or the original server. Every peer can only send one request in each round and also answer at most one request. Let τ be the duration of each round.

2.2. Game Theoretical Model

Next, we will investigate how to stimulate cooperation for all members in peer-to-peer live streaming over heterogeneous and error-prone networks, and analyze users' behavior dynamics. We focus on the scenario that video streaming will keep alive for a relatively long time, and there exist a finite number of users, for example, people watch live Super Bowl over the Internet. Each user will stay in the social network for a reasonably long time, for example, from the beginning to the end of the game. They are allowed to leave and reconnect to the network when necessary.For each user, uploading chunks to other users will incur some cost, and successfully receiving chunks can improve the quality of his/her video and thus brings some gain. To simplify the analysis, in this section, we assume the video stream is encoded using non-scalable video codec. Therefore, for each user i, each received chunk gives the same gain g_i, whose value is specified by the user individually and independently. As discussed in Section **??**, g_i, the gain of receiving a chunk for the live video, is evaluated by user i by how much he/she wants to watch the video. For instance, g_i should be set to 1 if at this moment, all user i wants to do is watch the live streaming. The more activities user i is doing simultaneously using the network bandwidth, the lower the g_i is. If user i is utilizing lots of his/her upload bandwidth and does not care about the quality of the live video stream, g_i should be set to 0, and user i will not join the P2P live stream social network.

In a real-world social network, some users may be malicious, whose goal is to cause damages to other users. In this paper, we focus on insider attackers, that is, the attackers also have legitimate identities, and their goal is to prevent the selfish users from getting chunks. In P2P live streaming social networks, there are two ways to attack the system:

1. **Incomplete chunk attack**: The malicious user agrees to send the entire requested chunk to the peer, but sends only portions of it or no data at all. By doing so, the requesting peer wastes his/her request quota in this round, and has to request the same chunk again in the next round.

2. **Pollution attack**: The other kind of attack in peer-to-peer live streaming is pollution [**?**]. In P2P streaming system, a malicious user corrupts the data chunks, renders the content unusable, and then makes this polluted content available for sharing with other peers. Unable to distinguish polluted chunks from unpolluted files, unsuspecting users download the polluted chunks into their own buffers, from which others may then download the polluted data. In this manner, polluted data chunks spread through the system.

Instead of forcing all users to act fully cooperatively, our goal is to stimulate cooperation among selfish users as much as possible and minimize the damages caused by malicious users. In general, not all cooperation decisions can be perfectly executed. For example, when a peer decides to send another peer the requested chunk, packets of the chunk may be dropped due to the overloaded routers. It is also possible that the chunk may fail to be completely received in one round due to the significant delay caused by the congested network. In this paper, we assume that the requesting peer gives up the chunk once it does not arrive in the round, and we use p_{ij} to denote the probability of successful transmission of a chunk from peer i to peer j in one round of τ second. At the beginning of every round, each user will send only one chunk request to one user. Each user will respond to only one request. We assume every chunk request can be received immediately and perfectly.

In order to formally analyze cooperation and security in such peer-to-peer live streaming networks, we model the interactions among peers as the following game:

- **Server**: The video is originally stored at the original streaming server with upload bandwidth W_s, and the server will send chunks in a round-robin fashion to its peers.

- **Players and player type**: There are finite number of users/peers in the peer-to-peer live streaming social network, denoted by N. Each player $i \in N$ has a type $\theta_i \in \{\text{selfish}, \text{malicious}\}$. Let N_s denote the set of all selfish players and $N_m = N \backslash N_s$ is the set including all insider attackers. A selfish user aims to maximize his/her own payoff, and may cheat other peers if cheating can help increase his/her payoff. A malicious user wishes to exhaust other peers' resources and attack the system.

- **Chunk requesting**: In each round, each player has *one* chunk-request quota, where he/she either *requests a chunk from a peer*, *requests a chunk from the video streaming source*, or *does not request any chunks* in this round.

- **Request answering**: For each player, after receiving a request asking for the upload of a chunk in its buffer, it can either *accept* or *refuse* the request.
- **Cost**: For any player $i \in N$, uploading a chunk to another player incurs cost $c_i = M/W_i\tau$, where W_i is player i's upload bandwidth and $W_i \geq W_{min} \geq M/\tau$.
- **Gain**: For each selfish user $i \in N_s$, if he/she requests a data chunk from another peer j, and if an unpolluted copy is successfully delivered to him/her, his/her gain is g_i where $P_{ji}g_i > c_i$.
- **Utility function**: We first define the following symbols: for each player $i \in N$,
  - $Cr^{(i)}(j,t)$ is the total number of chunks that i has requested from j by time t. Here, j can be either a peer ($j \in N$) or j is the streaming server. $Cr^{(i)}(t) = \sum_{j\in\{N,\text{ source}\}} Cr^{(i)}(j,t)$ denotes the total number of chunks that i has requested by time t.
  - By time t, peer i has successfully received $Cs^{(i)}(j,t)$ chunks from peer j in time (a chunk is received in time if and only if it is received within the same round that it was requested). $Cs^{(i)}(t) = \sum_{j\in\{N,\text{source}\}} Cs^{(i)}(j,t)$ is peer i's total number of successfully received chunks by time t.
  - By time t, $C_p^{(i)}(j,t)$ is the total number of polluted chunks that peer i received from peer j. The total number of successively received unpolluted data chunks that peer i received from peer j is $Cs^{(i)}(j,t) - C_p^{(i)}(j,t)$, and each successfully received unpolluted chunk gives peer j a gain of g_i.
  - $Cu^{(i)}(j,t)$ denotes the number of chunks that i has uploaded to player j by time t. $Cu^{(i)}(t) = \sum_{j\in\{N,\text{source}\}} Cu^{(i)}(j,t)$. The cost of uploading each chunk is c_i for peer i.

Let t_f be the lifetime of the peer-to-peer live streaming social network, and $T^{(i)}(t)$ denotes the total time that peer i is in the network by time t. Then, we model the player's utility as follows:

1. For any selfish player $i \in N_s$, its utility $U_s^{(i)}(t_f)$ is defined as in (1), where the numerator denotes the net profit (i.e., the total gain minus the total cost) that the selfish peer i obtained, and the denominator denotes the total number of chunks that i has requested. This utility function represents the average net profit that i can obtain per requested chunk, which i aims to maximize.
2. For any malicious player $j \in N_m$, its objective is to maximize its utility as defined in (2). The numerator in (2) represents the net damage caused by j: the first term describes the total costs to other peers when sending the requested chunks to the malicious user j; the middle term evaluates other selfish peers' potential loss in gain due to the incomplete chunk attack by peer j; and the last term is peer j's cost by uploading chunks to other peers. We normalize it using the lifetime of peer j, $T^{(j)}(t_f)$. Now, this utility function represents the average net damage that j causes to the other nodes per time unit.

3. ATTACK-RESISTANT COOPERATION STIMULATION STRATEGIES

?? In this section, we will discuss the two methods to resist pollution attack and incomplete-chunk attack and propose the attack-resistant cooperation strategy for P2P live-streaming social networks.

3.0.1. Credit Mechanism for Malicious User Detection

To distinguish "intentional" malicious behavior from "innocent" misbehavior caused by packet delay, we introduce the credit mechanism. Addressing the pollution attack, for any two peers $i, j \in N$,

$$Cc^{(i)}(j,t) = Cu^{(i)}(j,t) - C_p^{(j)}(i,t) \tag{3}$$

calculates the total number of *unpolluted* chunks that peer i has uploaded to peer j by time t. If the chunk is unpolluted, and is received before its playback time, then the chunk is useful. Note that for a selfish user $i \in N_s$, as discussed in the previous section, he/she has no incentives to intentionally send others polluted data chunks, since doing so will ultimately hurt himself/herself and lower the quality of his/her own video. However, since peer i cannot identify a chunk as a polluted one until he/she starts decoding and playing that chunk, it is possible that user i *unintentionally* forwards a polluted chunk to other peers. In this paper, addressing the above issue, we include the term $C_p^{(j)}(i,t)$ in (3) and consider the potential unintentional forwarding of polluted data chunks.

Given (3), we then define

$$D^{(i)}(j,t) = Cc^{(i)}(j,t) - Cc^{(j)}(i,t) = \Big(Cu^{(i)}(j,t) - C_p^{(j)}(i,t)\Big) - \Big(Cu^{(j)} \tag{4}$$

which is the difference between the number of *useful* chunks that peer i has sent to peer j and the number of *useful* chunks that peer j uploaded to peer i. Now, similar to the 2-player cooperation-stimulation strategy in Section **??**, we consider the following strategy: each selfish peer $i \in N_s$ limits the number of chunks that he/she sends to any other peer j such that by any time t, the total number of useful(unpolluted) chunks that i has forwarded to j should be no more than $Cu^{(j)}(i,t) - C_p^{(i)}(j,t) + D_{max}^{(i)}(j,t)$, that is,

$$D^{(i)}(j,t) \leq D_{max}^{(i)}(j,t), \quad \forall t \geq 0. \tag{5}$$

Here, $D_{max}^{(i)}(j,t)$ is the "credit line" that user i sets for user j at time t. The credit line is set for two purposes: 1) to prevent

$$U^{(i)}(t_f) = \frac{\left[Cs^{(i)}(t_f) - \sum_{j\in N} C_p^{(i)}(j,t_f)\right] g_i - Cu^{(i)}(t_f)\frac{M}{W_i\tau}}{Cr^{(i)}(t_f)} \quad (1)$$

$$U_m^{(j)} = \frac{\sum_{i\in N_s} Cu^{(i)}(j,t_f)\frac{M}{W_i\tau} + \sum_{i\in N_s}\left[Cr^{(i)}(j,t_f) - Cs^{(i)}(j,t_f)\right] P_{ji}g_i - Cu^{(j)}(t_f)\frac{M}{W_j\tau}}{T^{(j)}(t_f)} \quad (2)$$

egoism when favors cannot be simultaneously granted and to stimulate cooperation between i and j, and 2) to limit the possible damages that j can cause to i. By letting $D_{max}^{(i)}(j,t) \geq 0$, i agrees to send some extra, but at most $D_{max}^{(i)}(j,t)$ chunks to j without getting instant payback. Meanwhile, unlike acting fully cooperatively, the extra number of chunks that i forwards to j is bounded to limit the possible damages when j plays non-cooperatively or maliciously.

Player i's goal of setting the credit line is to avoid helping player j much more than player j helps i in long term's view, and vice versa, since neither of i, j has incentive to send more chunks than the other does. Meanwhile, due to the dynamically changing network conditions, the request rates between i and j may vary from time to time. In this case, the credit line has to be large enough since a small credit line will refuse some requests even when the long-term average request rates between i and j are equal. The ultimate goal of setting the credit line is to make sure that player i and j send asymptotically equal number of unpolluted chunks to each other, and

$$\lim_{t\to\infty} Cc^{(i)}(j,t) = \lim_{t\to\infty} Cc^{(i)}(j,t). \quad (6)$$

Combining the definition of $D_{max}^{(i)}(j,t)$ with (6), $D_{max}^{(i)}(j,t)$ must satisfy

$$\lim_{t\to\infty} \frac{D_{max}^{(i)}(j,t)}{Cr^{(i)}(t)} = 0, \quad (7)$$

which also implies that arbitrarily increasing credit lines cannot always increase the number of accepted requests. (7) provides an asymptotic upper bound for $D_{max}^{(i)}(j,t)$. Based on the above analysis, to stimulate cooperation in the first few rounds, $D_{max}^{(i)}(j,t)$ should be large enough in the first few cooperating rounds between user i and j. On the other hand, $D_{max}^{(i)}(j,t)$/[total number of rounds after time t] should be closed to 0 to prevent decreasing the utility of user i. Therefore, when choosing $D_{max}^{(i)}(j,t)$, user i should first estimate the number of remaining rounds for the live streaming, and choose a relatively small number D_{temp}. Then compare D_{temp} with the reciprocal of P_{ij}, so that $D_{max}^{(i)}(j,t)$ should be larger than $1/P_{ij}$ to stimulate the cooperation. A simple solution to this is to set the credit lines to be reasonably large positive constants, as in our simulations in Section **??**.

3.0.2. Malicious User Detection

Malicious attacks, such as the incomplete chunk attack and the pollution attack, exhaust other peers' resources and cause damages to the P2P live streaming system. Thus, it is of critical importance to implement a monitoring system to detect and identify misbehaving users, and a challenging issue is to differentiate "innocent" misbehavior (due to erroneous and congested networks) from "intentional" ones (for example, intentional pollution attacks).

If the credit line is set properly to satisfy (7), the damage of the pollution attack can be controlled to 0 asymptotically. Since the pollution attack will not effect honest users' utility by the credit line mechanism, in this section, we propose a malicious user detection algorithm that can differentiate the incomplete information attack to ensure the attack-resistance of the P2P live streaming social network.

P_{ij} is the probability of successful transmitting one chunk within round period τ. Hence when player $i$decides to send a chunk to player j, with probability $1 - P_{ij}$, this chunk transmission cannot be completed within one round because of packet dropping or delay caused by high traffic internet. That is, we use a Bernoulli random process to model the unsuccessful transmission of a chunk due to high traffic internet connection. Recall that that $Cu^{(j)}(i,t)$ denote the number of chunks that i has requested from j and j has agreed by time t, and $Cs^{(i)}(j,t)$ is the number of chunks that peer i successfully receives from j in one round. Given the Bernoulli random process, if user j does not intentionally deploy the incomplete chunk attack, based on the Central Limit Theorem [**?**], for any positive real number x, we can have

$$\lim_{Cu^{(j)}(i,t)\to\infty} Prob\left(\frac{Cs^{(i)}(j,t) - Cu^{(j)}(i,t)P_{ji}}{\sqrt{Cu^{(j)}(i,t)Pji(1-P_{ji})}} \geq -x\right) = \Phi(x), \quad (8)$$

where $\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}dt$ is the Gauss tail function. If user j does not intentionally sends incomplete chunks, (8) indicates that when the peer-to-peer live streaming game keeps going and $Cu^{(j)}(i,t)$ is large enough, then $Cs^{(i)}(j,t) - P_{ji}C_u^{(j)}(i,t)$ can be approximated by a Gaussian random variable with zero mean and variance $Cu^{(j)}(i,t)P_{ji}(1-P_{ji})$, that is,

$$Cs^{(i)}(j,t) - Cu^{(j)}(i,t)P_{ji} \sim \mathcal{N}\left(0, Cu^{(j)}(i,t)P'_{ji}(1-P_{ji})\right). \quad (9)$$

Therefore, based on (9), given a predetermined threshold $h > 0$, every selfish peer i can identify peer j as a malicious user

by thresholding $Cs^{(i)}(j,t) - Cu^{(j)}(i,t)P_{ji}$ as follows:

$$\begin{aligned} j \in N_m^{(i)}(t) \quad \text{if and only if} \quad Cs^{(i)}(j,t) - Cu^{(j)}(i,t)P_{ji} &\leq -h\sqrt{Cu^{(j)}(i,t)P_{ji}(1-P_{ji})}, \\ \text{and} \quad j \in N_s^{(i)}(t) \quad \text{if and only if} \quad Cs^{(i)}(j,t) - Cu^{(j)}(i,t)P_{ji} &> -h\sqrt{Cu^{(j)}(i,t)P_{ji}(1-P_{ji})} \end{aligned} \tag{10}$$

In (10), $N_m^{(i)}(t)$ is the set of peers that are marked as malicious by peer i at time t, and $N_s^{(i)}(t)$ is the set of peers that are marked as selfish by peer i at time t. Based on (10), if the malicious user is always sending incomplete chunks to other users, then the probability of correctly identify the malicious user (P_d) and the probability of falsely accusing a nonmalicious user as malicious (P_{fa}) can be formulate as

$$P_d = 1 - \Phi(h), \quad \text{and } P_{fa} = \Phi(h). \tag{11}$$

4. REFERENCES

[1] "The software is available at http://www.pplive.com/en/index.html," .

[2] G. Owen, *Game Theory*, Academic Press, 3rd edition, 2007.

[3] Chiranjeeb Buragohain, Divyakant Agrawal, and Subhash Suri, "A game theoretic framework for incentives in p2p systems," *In Proceeding of the International Conference on Peer-to-Peer Computing*, pp. 48–56, Sep 2003.

[4] Zhengye Liu, Yanming Shen, Shivendra Panwar, Keith Ross, and Yao Wang, "Using layered video to provide incentives in p2p live streaming," *ACM Special Interest Group on Data Communication*, August 2007.

[5] Ahsan Habib and John Chuang, "Incentive mechanism for peer-to-peer media streaming," *International Workshop on Quality of Service (IWQoS)*, pp. 171–180, June 2004.