INCENTIVE ANALYSIS FOR COOPERATIVE DISTRIBUTION OF INTERACTIVE MULTIVIEW VIDEO

Bo Hu [*], *Gene Cheung* [#] *and H. Vicky Zhao* [*]

[*] University of Alberta, [#] National Institute of Informatics

ABSTRACT

In interactive multiview video streaming (IMVS), users can periodically select one out of many captured views available for observation as video is played back in time. In single-view video streaming, to reduce server's upload burden, cooperative strategies where peers share received packets of the same video have proven to be effective, and incentive mechanisms are designed to stimulate user cooperation. Exploiting user cooperation in high dimensional IMVS, however, is more challenging. First, small number of peers in a local area are likely watching different views among large number of views available, making it difficult for a peer to find partners of the exact same view to cooperate. Second, even if a peer can identify cooperative partners of the same view, they will soon be watching different views after independent view-switching. In this paper, we study the use of a multiview video frame structure for IMVS that facilitates cooperative view switching, where even if peers are observing different views, they can nonetheless help each other. To stimulate user cooperation, we model peers' interaction as an indirect reciprocity game. Using Markov decision process (MDP) as a formalism, each peer makes distributed decisions to maximize his aggregate utilities within his lifetime. Simulation results show that when the cost to help others is much smaller than the utility gained from others' help, users fully cooperate. As the cost-to-gain ratio increases, users tend to behave differently at different views: given peers can predict their future view navigation paths probabilistically, a peer likely to enter a view-switching path not requiring others' help will have less incentive to cooperate. When the cost-to-gain ratio is very large, no users will cooperate.

Index Terms— interactive multiview video, cooperative streaming, incentive mechanisms

1. INTRODUCTION

Multiview video refers to the simultaneous capturing of multiple videos of the same scene of interest by a large array of closely spaced cameras from different viewpoints. In new interactive multiview view streaming (IMVS) services [1], a client can periodically select one out of many captured views available for observation as the video is played back in time. In response, server sends only pre-encoded data for the single requested view (rather than all the captured views) to lower streaming rate.

In single-view video streaming, to ease server's burden to upload the same video to many users, user cooperation [2] has been exploited where peers share received packets of the same video, so that a single server can serve a large number of clients. Incentive mechanisms [3] are designed to stimulate the appropriate amount of cooperation among selfish peers. However, exploiting user cooperation in high dimensional IMVS is more challenging. First, small number of peers in a local area are likely watching different views among large number of views available, making it difficult for a peer to find partners of the exact same view to cooperate. Second, even if a peer can identify cooperative partners of the same view, they will soon be watching different views after independent view-switching.

In the literature, [1] designed frame structures using distributed source coding (DSC) [4] for IMVS to achieve bandwidth-efficient view switching. [5] used DSC for both view-switching and cooperative packet loss recovery in a WWAN multiview video multicast system. In this work, we study the use of a multiview video coding structure based on DSC to facilitate *cooperative view-switching* to reduce upload bandwidth from the server: even if users are in different views, they can nonetheless help each other achieve low-bitrate view-switching. To stimulate user cooperation, we model users' interaction as an indirect reciprocity game [6], where each user is assigned a reputation level. Users that help others will accumulate high reputations, thus more likely to receive help from others. Using Markov decision process (MDP) as a formalism, each peer makes distributed decisions to maximize his aggregate utilities within his lifetime. Simulation results show when the cost of helping others is much smaller than the utility gained from others' help, users fully cooperate, which also helps reduce server's upload bandwidth. As the cost-to-gain ratio increases, users tend to behave differently at different views: given peers can predict their future view navigation paths probabilistically, a peer likely to enter a view-switching path not requiring others' help will also have less incentive to cooperate. When the cost-to-gain ratio is very large, no users will cooperate.

The outline of the paper is as follows. We overview the formulation of the IMVS system in Section 2. We model the optimal decision making using MDP in Section 3. We present simulation results and conclusions in Section 4 and 5, respectively.

2. PROBLEM FORMULATION

In this section, we first overview an IMVS system that supports periodic view-switching by users. We then describe an interaction model that captures users' behavior in view-switching, and a multiview video coding structure that facilitates cooperative view-switching among peers. Finally, we propose an indirect reciprocity game to stimulate user cooperation.

2.1. Overview of IMVS system

A scene of interest is captured by a large one-dimensional array of evenly spaced M cameras. A server compresses video of each view into coding segments of K frames each, and provides IMVS service to a group of N users, where $N \ll M$. Once a user selects a view, he remains in this view for K consecutive frames. At the end of this segment, he can switch to another view as the video is played back in time uninterrupted. There is hence a maximum of one segment view-switching delay. When a user switches views, he may request help from other peers (cooperative view-switching), so that the amount of downloaded video data from server can be reduced.

2.2. View Switching Model

Views are divided into two categories: *anchor views* and *normal views*. When seeking interested views, a user first browses views

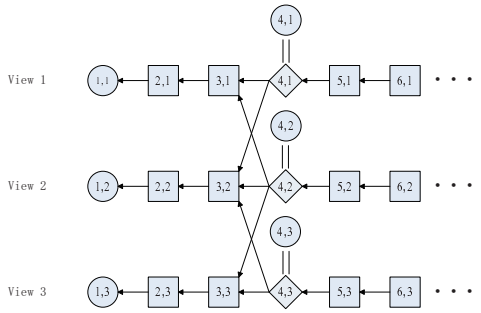

Fig. 1. Example of our multiview video coding structure for $M = 3$ views, segment size $K = 3$. Circles, squares and diamonds denote I-, P- and DSC frames, respectively. Each frame $F_{f,v}$ is labeled by its frame index f and view v.

coarsely through anchor views. Once he reaches an interested anchor view, he can switch to neighboring normal views to refine view selection. In this work, we assume that users switch interested anchor views frequently. After finding an interested anchor view and remaining for one segment, they will likely seek another interested anchor view in the next segment. Thus, anchor views are more frequently selected (more popular) than normal views.

Suppose that there are n_δ anchor views, which evenly divide normal views into $(n_\delta + 1)$ sections of $n_\sigma = (M - n_\delta)/(n_\delta + 1)$ views each. At each view, a user can only switch to his left and right closest anchor views, *and* nearby normal views. Specifically, at an anchor view, a user will switch to the left and right closest anchor views with probability P_δ, and the left and right normal view sections (and current anchor view) with probability $1-P_\delta$. Similarly, at a normal view, a user will switch to the left and right closest anchor views with probability P_σ, and the normal views in the same section with probability $1 - P_\sigma$.

We model transition from view to view using a discrete time Markov chain, and construct a $M \times M$ transition matrix T, where $t_{x,y}$ is the view transition probability of a user selecting view y after viewing x. From earlier discussion on view switching model, if x is an anchor view,

$$t_{x,y} = \begin{cases} P_\delta/|X_\delta| & \text{, if } y \in X_\delta \\ (1-P_\delta)/|X_\sigma| & \text{, if } y \in X_\sigma \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where, X_δ and X_σ denote view x's closest anchor view set, and nearby normal view set, respectively. If x is a normal view, $t_{x,y}$ can be defined similarly.

2.3. Multiview Video Coding Structure and IMVS Service

Fig. 1 shows an example of our proposed frame structure. Each view is encoded into segments of K frames. We encode the first segment using an intra-coded I-frame $I_{1,v}$ with $K-1$ trailing P-frames. For the next segment, for view-switching we encode the first frame $F_{K+1,v}$ into two versions. The first version is an intra-coded I-frame $I_{K+1,v}$, which can be decoded independently. The second version is a *DSC frame* $W_{K+1,v}$ [4]. To encode $W_{K+1,v}$, we use at most three decoded P-frames $P_{K,max(1,v-1)}, \ldots, P_{K,min(M,v+1)}$ as predictors, and use the I-frame $I_{K+1,v}$ as the target. As long as one of the predictor frames is available at the decoder buffer, $W_{K+1,v}$ can be correctly decoded, and the decoded frame is bit-by-bit equivalent to the frame decoded from $I_{K+1,v}$. $W_{K+1,v}$ is followed by $K-1$ trailing P-frames, and each of the following segments has the same structure. Let L_I, L_W, and L_P denote the frame size of an I-frame, DSC frame, and P-frame, respectively, and averagely, $L_I \gg L_W > L_P$.

This structure can support cooperative view switching as follows. Suppose that a peer switches from view v to v'. If v' is adjacent to v, he can ask the server to transmit the DSC frame $W_{iK+1,v'}$ and the following P-frames to reconstruct the video in view v'. However, if v' is not adjacent to v, he has to either receive help from other peers or request from the server the I-frame in view v'. Using Fig. 1 as an example, suppose that the peer i switches from view 1 to view 3 after the first segment. If another peer is watching view 2 and would like to share the reconstructed frame $F_{3,2}$, then i only needs to ask the server for the DSC frame $W_{4,3}$ and the following P-frames to reconstruct the video in view 3. If he cannot get help, he has to request the I-frame $I_{4,3}$ from the server.

2.4. Game Formulation: Indirect Reciprocity Game

In this work, we assume that the upload bandwidth of the server is limited and expensive. Thus, it charges virtual currency from peers that pull video data from it to compensate its cost, and b denotes the price for the transmission of each single bit from the server. As discussed in section 2.3, when a peer switches to a non-adjacent view, if he can get help from others, he will download the reconstructed frame of the last segment from the helper for free, and will only download a DSC frame from the server instead of an I-frame. Thus, he can gain an utility $g = b(L_I - L_W)$ for paying less to the server. However, uploading a video frame will incur a cost c to the helper due to the consumed bandwidth and CPU time, etc. Since users are selfish, they want to receive others' help, but do not want to cooperate and upload video packets. In this work, to stimulate user cooperation, we design a reputation-based mechanism, where peers that keep helping others will accumulate good/high reputations, and peers that have good reputations also tend to receive others' help. In this mechanism, peer i helping peer j is not because j directly helped i previously, but j helped someone else. Thus, it is an indirect reciprocity game.

2.4.1. Peer Reputation and Interaction

In this system, each peer i is assigned a discrete reputation value $r_i \in \mathcal{R} = \{1, ..., L\}$. A larger value of r_i means a better reputation. Reputation values change as peers interact with each other. When a peer needs help for cooperative view-switching, he first needs view information of other peers to find a suitable helper. To implement this, we can either let peers exchange their view information and seek help in a distributed way, or have a central controller that tracks peers' up-to-date view information and assigns helpers to peers that need help. For simplicity, we assume here that there is a trustworthy local agent closed to the N peers, which tracks peers' view switching, helps each peer find helpers, observes their interactions, and updates their reputations. Our work can also be extended to a distributed system. Thus, in this centralized system,when peer j needs help, the local agent randomly selects i from peers that can help, and sends a help request. Upon receiving a help request, i takes an action $a_i \in \mathcal{A} = \{1, \ldots, L+1\}$. The action a_i is not a direct answer of whether to help or not, but a *reputation threshold*. i sends the action a_i back to the local agent, who then compares j's reputation r_j with the threshold a_i. If $r_j \geq a_i$, the local agent informs i to upload the needed video data to j. Otherwise, the local agent informs j to pull the I-frame from the streaming server. Thus, if $a_i = L + 1$, i will not cooperate regardless of j's reputation.

2.4.2. Social Norm and Reputation Updating

Based on the previous observed interaction between peer i and j, the local agent updates i' reputation, since he is the decision maker, while j' reputation remains the same. The reputation is updated following a *social norm* Q [6], which effects changes in a peer's reputation after an interaction:

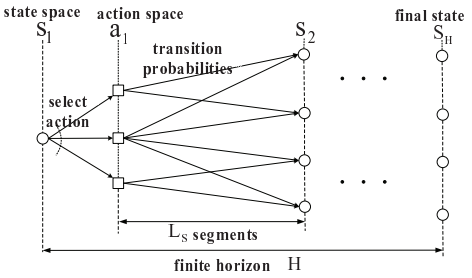

Fig. 2. An example of MDP.

$$Q = \begin{matrix} a_i \le r_j, \text{ uploading} \\ a_i > r_j, \text{ not uploading} \end{matrix} \begin{matrix} r_j \ge r_i & r_j < r_i \\ \left(\begin{matrix} L & r_i \\ 1 & r_i \end{matrix} \right), \end{matrix} \quad (2)$$

where the rows of Q denote the results of this interaction of whether i uploads the requested frame to j. The columns of Q denote j's reputation being larger or smaller than i's reputation. When $r_j \ge r_i$, i will gain an immediate reputation L by helping j, or be punished with an immediate reputation 1 by not helping. When $r_j < r_i$, no matter whether i helps j, i's reputation remains the same. With this social norm, peers are encouraged to help those whose reputations are larger than or equal to their own, while they are discouraged to cooperate with others that have smaller reputations, since cooperating does not improve their own reputations.

Let $\psi(a_i, r_j)$ and $\phi(r_i, r_j)$ be functions that return the row and column indices of Q, respectively. If $r_j \ge a_i$, $\psi(a_i, r_j) = 1$; otherwise $\psi(a_i, r_j) = 2$. If $r_j \ge r_i$, $\phi(r_i, r_j) = 1$; otherwise $\phi(r_i, r_j) = 2$. Then, i' reputation is updated by,

$$r_i(t) = round\left[\lambda r_i(t-1) + (1-\lambda) Q_{\psi(a_i, r_j), \phi(r_i, r_j)}\right], \quad (3)$$

where $round[\cdot]$ is the rounding function, and λ is a parameter weighting the past versus immediate reputation values.

Given this reputation system, helpers need reputation information of users that need help to decide appropriate actions. In this work, we consider that they also take the future interactions into consideration. Since they do not know who they will interact with at a later time, they need peers' reputation distribution to assist their decision making. Given that the local agent has the record of all peers' reputations at different time instances, it can calculate the probability mass function $P_r(l)$ of each reputation value, $l \in \mathcal{R}$, appearing in the peer population since the beginning of the game,

$$P_r(l) = \frac{\sum_{t=1}^{T_c} \sum_{i=1}^{N} I[r_i(t) = l]}{N T_c}, \quad (4)$$

where T_c is the current segment index, and $I[\cdot]$ is the indicator function. Let $\mathcal{D} = \{P_r(1), P_r(2), ..., P_r(L)\}$. The local agent broadcasts $\mathcal{D}$ to all peers periodically to assist their decision making.

3. OPTIMAL ACTION SELECTION WITH MARKOV DECISION PROCESS

In this work, we formalize peers' distributed decision making process using MDP, which is used in previous work [7]. Each peer considers actions taking future utility into consideration, and finds the optimal actions to maximize his utility within his lifetime.

3.1. State Space, Action Space and State Transition Probability

For a peer i, MDP is a recursion with finite levels into the future, as shown in Fig. 2, where each level t in the future is marked by its states s_i^t's and actions $a_{s_i}^t$'s. In this work, a state $s_i^t(r_i, v_i)$ represents reputation r_i and view v_i at the moment t peer i receives a help request. Hence the state space is denoted as $\mathcal{S} = \mathcal{R} \times \mathcal{V}$, where $\mathcal{V} = \{1, ..., M\}$ and is the view space. At state s_i^t, he responses to the help request by choosing action $a_{s_i}^t$ from the action space $\mathcal{A}$. From state-action pair $(s_i^t, a_{s_i}^t)$, he transits to a new state $s_i^{t+1}(r_i', v_i')$ in the next level with probability $P_{s_i^t \to s_i^{t+1}}(a_{s_i}^t)$, when he receives another help request after playback of L_s video segments. Here, L_s can be learned from his past history. To derive the state transition probabilities, we first derive its reputation and view transition probabilities, respectively.

For action $a_{s_i}^t$ taken at time t, peer i's reputation is updated using (3). Suppose that the user that needs this help is j with reputation r_j. In (3), $Q_{\psi(a_{s_i}^t, r_j), \phi(r_i, r_j)}$ can be one of three values: 1, r_i or L. Thus, the updated reputation of i can also only be one of three possible values. Let $r_i'(1)$, $r_i'(r_i)$ and $r_i'(L)$ be the updated reputation value when $Q_{\psi(a_{s_i}^t, r_j), \phi(r_i, r_j)} = 1$, r_i or L, respectively. Since i does not know the exact value of r_j, he assumes that r_j follows the reputation distribution $\mathcal{D}$. Thus, the probability that the reputation value updates to $r_i' = r_i'(1)$ is

$$\begin{aligned} P_{r_i \to r_i'(1)}(a_{s_i}^t) &= P(Q_{\psi(a_{s_i}^t, r_j), \phi(r_i, r_j)} = 1) = P(r_i \le r_j < a_{s_i}^t) \\ &= \sum_{l:\ r_i \le l < a_{s_i}^t} \mathcal{D}(l). \end{aligned} \quad (5)$$

Similarly, we can derive $P_{r_i \to r_i'(r_i)}(a_{s_i}^t) = \sum_{l:\ l < r_i} \mathcal{D}(l)$, and $P_{r_i \to r_i'(L)}(a_{s_i}^t) = 1 - P_{r_i \to r_i'(1)}(a_{s_i}^t) - P_{r_i \to r_i'(r_i)}(a_{s_i}^t)$. Once updated to r_i', i's reputation stays the same at r_i', till the next instant he receives another help request at state $s_i^{t+1}(r_i', v_i')$.

The view transition probability follows the view transition matrix T and is independent of the reputation transition probability. From state s_i^t, there are L_s segments and view transitions before i receives another help request at state s_i^{t+1}. Let $T_{v_i, v_i'}^{L_s}$ denotes the entry of row v_i and column v_i' of T^{L_s}. Thus, the probability of switching from v_i to v_i' is $P_{v_i \to v_i'} = T_{v_i, v_i'}^{L_s}$. Therefore, given the above discussion on reputation and view transition probabilities, the state transition probability is $P_{s_i^t \to s_i^{t+1}}(a_{s_i}^t) = P_{r_i \to r_i'}(a_{s_i}^t) P_{v_i \to v_i'}$.

3.2. Utility Function

In this subsection, we derive peer i's utility based on his state and action. We first discuss his utility cost for helping others upload video packets. Then, we study the utility gain he receives from others' help. Finally, we derive the utility of his life time since the current state s_i^t. As discussed previously, helping one neighbor upload video data will incur a cost c, and the user that receives this help gains an utility g. When i is in state s_i^t and take the action $a_{s_i}^t$, he assumes the reputation of the user that requests this help follows $\mathcal{D}$, and therefore, he will upload video packet with probability

$$P_u(a_{s_i}^t) = \sum_{l:\ l \ge a_{s_i}^t} \mathcal{D}(l). \quad (6)$$

Thus, the cost incurred by the action $a_{s_i}^t$ is $cP_u(a_{s_i}^t)$.

After this interaction, his reputation changes to r_i' and he is in view v_i. He will keep the reputation r_i' for L_s segments till the next time he receives another help request at state s_i^{t+1}. During these L_s segments, he may require others' help, and the utility he/she can gain from others' help depends on his reputation r_i', the view v_i he currently watches, and other peers' actions. Let $P_v^{l_s}$ denote the probability that he needs help when transferring to view v at the l_sth segment after the current state s_i^t, where $v \in \mathcal{V}$ and $1 \le l_s \le L_s$, and let P_h^v denote the probability that a helper k can be found to help in view v. The helper k is in state $s_k(l, v_k)$ with probability $\mathcal{D}(l) P_{v_k}$, where $l \in \mathcal{R}$ and $v_k \in \mathcal{V}$. At s_k, k will take action a_{s_k} following his action policy, and only when $r_i' \ge a_{s_k}$, i can receive help from k. Let $U(r_i', v_i)$ denote the utility he can gain from others' help during the L_s segments, and we have,

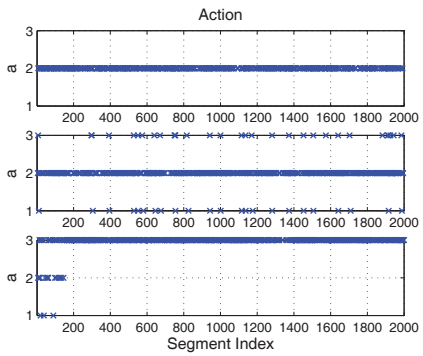

Fig. 3. User 1' actions at each segment. (Top): $c/g = 0.3$, (middle): $c/g = 0.6$, and (bottom): $c/g = 0.9$.

$$U(r'_i, v_i) = g \sum_{l_s=1}^{L_s} \left\{ \sum_{v=1}^{M} P_v^{l_s} P_h^v \left[\sum_{l=1}^{L} \sum_{v_k=1}^{M} \mathcal{D}(l) P_{v_k} I[r'_i \geq a_{s_k}] \right] \right\} \quad (7)$$

Given above analysis, we can derive the utility function for peer i's lifetime since the current state as $W(s_i^t)$,

$$W(s_i^t(r_i, v_i)) = -P_u(a_{s_i}^t)c + \sum_{q \in \{1, r_i, L\}} P_{r_i \to r'_i(q)}(a_{s_i}^t) U(r'_i(q), v_i)$$
$$+\eta \sum_{q \in \{1, r_i, L\}} \sum_{v'_i=1}^{M} P_{r_i \to r'_i(q)}(a_{s_i}^t) P_{v_i \to v'_i} W(s_i^{t+1}(r'_i(q), v'_i)) \quad (8)$$

where η is the discounting factor. In (8), we derive $W(s_i^t)$ recursively. The first term is the utility cost for helping others. The second term is the utility gain he receives through others' help in the L_s segments. The last term is his life time utility since the next state s_i^{t+1}. i wants to find the optimal action at each state to maximize the utility for his life time. To achieve this, we use dynamic programming, and let the life time utility at the Hth state after s_i^t, $W(s_i^{t+H}) = 0$, if $\eta^H < 0.1$, to avoid the infinite recursion.

4. EXPERIMENTATION

This section evaluates the system performance by simulations. In the simulation setup, the server provides IMVS with $M = 11$ views to a group of $N = 3$ users. The video segment size is $K = 10$. The average sizes of I- and DSC frame are 5 and 1.5 packets, respectively. For the view switch model, the 4th and 8th views are anchor views, which divide the rest normal views into 3 sections with $n_\sigma = 3$ views per section. If a user is in a normal view, the probability of switching to the closest anchor views is $P_\sigma = 0.7$. If he/she is in an anchor view, the probability of switching to the closest anchor views is $P_\delta = 0.3$. In the reputation system, $L = 2$ and $\lambda = 0.3$. The discounting factor $\eta = 0.9$. We test the system for 2000 segments.

Fig. 3 shows user 1's actions in each segment, and for other users we have similar observation. The top figure shows his/her actions when the cost to gain ratio $c/g = 0.3$, where he always play $a = 2$ at the steady state, and we observe that users reputations are all 2. Since $a = 2$ means he will cooperate with others whose reputation is larger or equal to 2, he fully cooperates. This is because the cost is comparatively low, and cooperating help him receive high utility. At the bottom figure when $c/g = 0.9$, the cost is high, and he will receive negative utility, if he helps upload video packets. Thus, he tends to not help and plays the action $a = 3$ at the steady state, which means he does not cooperate regardless of others' reputations. For the case when $c/g = 0.6$, we observe that sometimes he plays action $a = 3$ and do not cooperate, but sometimes he plays action $a = 1$ or 2 and cooperate. To understand this better, we study his actions at different views. Table 1 shows the percentages of his actions from

	v_1	v_2	v_3	v_4	v_5	v_6
$a = 1$	11%	12%	0	8%	5%	5%
$a = 2$	89%	88%	0	92%	94%	95%
$a = 3$	0	0	100%	0	1%	0

Table 1. Percentages of users' actions at different views.

v_1 to v_6. Views from v_7 to v_{11} are symmetric to views from v_5 to v_1, which have similar results and are omitted in this table. From this table, we observe that when he receives requests at v_3, most of his responses are $a = 3$ and he does not cooperate. When he is in the other views, he uses $a = 1$ or 2 and cooperates. This is because v_3 is adjacent to an popular anchor view v_4 that is also the only popular view around v_3. Thus, a user in v_3 probably switches to v_4 in the next segment, and this view switching does not need others' help. Therefore, he also has less incentive to cooperate and maintain a high reputation.

We also evaluate the reduction of bandwidth consumption at the server side due to user cooperation as $C_d = \frac{(L_I - L_W)n_h}{L_I n_n}$, where n_h denotes the number of times that the 3 users get help from others, and n_n denotes the number of times that the 3 users switch to non-adjacent views. When $c/g = 0.3$, users fully cooperate with each other and $C_d = 37\%$. When $c/g = 0.6$, although users do not cooperate at view 3 and 9, they cooperate at other views and $C_d = 33\%$. Thus, our indirect reciprocity scheme can simulate user cooperation and reduce bandwidth consumption.

5. CONCLUSION

In this work, we propose an IMVS system that supports cooperative view-switching. To stimulate user cooperation, we model users' interaction as an indirect reciprocity game. Using MDP as a formalism, each peer makes distributed decisions to maximize his aggregate utilities within his lifetime. Simulation results show that when the cost of helping others is much smaller than the utility gain from others' help, users fully cooperate, which also helps reduce server's upload bandwidth. As the cost-to-gain ratio increases, users tend to behave differently at different views: given peers can predict their future view navigation paths probabilistically, a peer likely to enter a view-switching path not requiring others' help will also have less incentive to cooperate. When the cost-to-gain ratio is very large, no users will cooperate.

6. REFERENCES

[1] G. Cheung, A. Ortega, and N. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 744–761, March 2011.

[2] H. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "Insights into pplive: A measurement study of a large-scale P2P IPTV system," in *IPTV workshop in conjunction with WWW2006*, May 2006.

[3] W. Lin, H. V. Zhao, and K. J. R. Liu, "Incentive cooperation strategies for Peer-to-Peer live streaming social networks," *IEEE Transaction on Multimedia*, vol. 11, no. 3, pp. 396–412, April 2009.

[4] N.-M. Cheung, A. Ortega, and G. Cheung, "Distributed source coding techniques for interactive multiview video streaming," in *27th Picture Coding Symposium*, Chicago, IL, May 2009.

[5] Z. Liu, G Cheung, and Y. Ji, "Distributed source coding for WWAN multiview video multicast with cooperative peer-to-peer repair," in *IEEE International Conference on Communications*, Kyoto, Japan, June 2011.

[6] Y. Chen and K. J. R. Liu, "Indirect reciprocity game modelling for cooperation stimulation in cognitive networks," *IEEE Transaction on Communications*, vol. 59, no. 1, pp. 159–168, Jan. 2011.

[7] H. Park and M. van der Schaar, "A framework for foresighted resource reciprocation in P2P networks," in *IEEE Transactions on Multimedia*, January 2009, vol. 11, no.1, pp. 101–116.