Parent Selection via Reinforcement Learning in Mesh-based P2P Video Streaming*

Muge Fesci Sayit, Yagiz Kaymak, Kemal Deniz Teket, Cihat Cetinkaya, Sercan Demirci, Geylani Kardas
Ege University, International Computer Institute
Izmir, Turkey
muge.fesci@ege.edu.tr, yagiz.kaymak@ege.edu.tr, denizkemal@gmail.com, cihat.cetinkaya@ege.edu.tr,
sercan.demirci@ege.edu.tr, geylani.kardas@ege.edu.tr

Abstract— **There are several successful deployments of peer to peer (P2P) video streaming systems which provide acceptable QoS. Researches on these systems continue to improve the experienced quality by system users. Since received video quality mostly depends on the parent selection, an efficient parent selection algorithm can increase the received video bitrate by peers and provide seamless streaming. In this paper, we propose a novel parent selection method based on reinforcement learning. By the proposed system model, the newly joined peer explores the peers in the system first, and uses this information for its further parent selection actions. We implemented our model on a CoolStreaming-like P2P video streaming system in ns3. Our results indicate that, selected parents by using reinforcement learning approach improve the playback continuity, with respect to parent selection method used by CoolStreaming. Furthermore, reinforcement learning approach helps peers to find more stable parents in case of peer churn.**

Keywords: Parent selection; video streaming; peer to peer networks; reinforcement learning; Q-learning

I. INTRODUCTION

Peer to peer (P2P) video streaming systems are one of the most interesting and challenging multimedia networking systems. Thereby, successful commercial and academic applications related to this area are introduced in recent years.

Commercial P2P video streaming systems such as [1-3] are constructed as mesh-based in which video data are partitioned into smaller sized chunks that are sent between the nodes in the system. Mesh-based systems can be classified into two groups as pure pull and push-pull method, according to their data dissemination algorithm. In pure pull systems, video data are disseminated between peers according to buffer maps exchanging between each other, which shows the available chunks belonging to peer. On the other hand, in push-pull method, video data are partitioned into K different substreams and peers are subscribed to one or more parents in order to receive these partitions and to combine them for constructing video file. Substreams can be constructed by using a single coded video, multiple coded video or a multiple description coding (MDC) scheme. While scalable coded video or MDC is used, each substream received by peers increases the video quality. If single coded video is used, all substreams should be received to play video properly. In the new CoolStreaming, it has been shown that push-pull system improves the playback continuity when it is compared to its earlier pure pull based version [1].

In P2P video streaming systems, in order to improve experienced performance a peer can select more suitable parents to receive video data properly by gathering information such as other peers' upload capacity and churn frequency and by using its previous experience. By reinforcement learning, a learner can gather some information about the environment that it resides in and use this information to guide its further actions and finally receives rewards. If reinforcement learning method is performed in P2P video streaming systems, a peer may learn about system facts in exploration phase and it may take actions according to its knowledge [4]. This approach provides that peer discovers the actions with higher reward over time. Furthermore, if a learning model based on reinforcement learning is used, it is not required to have global underlying topology and to have information about system in advance.

It is well known that the parent selection problem is crucial for P2P video streaming systems and there are remarkable algorithms introduced for this problem in the literature [1], [5-8]. Although this problem has been studied well, there is still some room for improvement. In this paper, we focused on parent selection problem for a push-pull based video streaming system and propose a reinforcement learning model to select parents for each substream. In order to observe the performance of the proposed scheme, we compared the results with a parent selection method introduced in [1] which is one of the pioneer commercial P2P systems, in terms of average received bitrate, continuity index (CI) and parent change count. Although proposed system is developed for push-pull based mesh system, it can be adopted to a different peer selection algorithm designing for pure pull based systems.

The paper is organized as follows: In section two, we give information about related work. We highlight the P2P video streaming system details used in the experiments and explain the proposed learning model in section three. In the next section, the results of the experiments and graphics obtained from ns3 simulations are discussed. Finally, we conclude with including planned future directions.

*This work is funded by the Scientific and Technological Research Council of Turkey (TUBITAK) Electric, Electronic and Informatics Research Group (EEEAG) under grant 111E022.

II. RELATED WORK

To construct an efficient P2P video streaming system, a good partner selection strategy should be established among peers. In most systems, partner selection strategy is based on availability of video chunks [1, 2, 7, 8]. In other systems, partner selection can be done by considering the stability and bandwidth [5, 6] of candidate partners, the distance in terms of delay [9] or path loss rate [10]. It is also seen that there is a tendency of improving partner selection algorithms which provides incentive mechanisms [6, 8]. Although these algorithms improve the performance of P2P systems, the performance may also be increased if peers keep and use historical data about their partner selection decisions.

Besides, there are remarkable studies implementing learning paradigms to P2P video streaming applications. In [11], the authors present "superiority index", a parameter defining a peer's estimated contribution. Superiority index is obtained by multiplying two parameters, session duration and upload bandwidth of the peer, which are calculated by using regression models. In [12], a model based on clustering with linear regression and neural networks has been proposed for determining optimal server bandwidth allocation to channels and detecting system anomalies, respectively. Both studies are tested on UUSee [3] and test results show that proposed learning methods improve streaming quality. A group of studies are interested in peer behaviors such as stability or bandwidth contribution in P2P systems with machine learning and estimation techniques [13, 14]. However, these systems need different global parameters such as arrival/departure rates of nodes, content popularity or content access patterns in order to implement their learning model. Reinforcement learning methods can also be used in P2P applications, since it is not required for a user to have some prior knowledge about system to be able to implement learning methodology. From this point of view, a peer selection method using reinforcement learning framework for Bittorent-like file sharing systems is proposed in [15]. The model proposed in [15] provides an improvement for P2P file sharing system. But the learning model may not be directly performed to real-time P2P video streaming systems because of its complexity. In our previous work [16] a different reinforcement learning model for P2P video streaming applications is proposed. However proposed model in [16] has high space complexity which is difficult to implement for real-time systems consisting of a large number of peers. In the model, the states are constructed as ordered tuples of parents for each node, and state space grows fast with increasing number of combinations of ordered tuples.

In this paper, we propose a learning model, in which peers can keep the previous data about its actions and use this information to improve the experienced performance in a P2P video streaming system.

III. SYSTEM DESIGN

In this section, we provide the details of our proposed system, the overlay construction scheme used in the experiments, the fundamentals of CoolStreaming-like parent selection and learning model based on Q-learning technique implemented to the parent selection approach.

A. Overlay Construction

Bootstrapping is similar to that of mesh-based P2P video streaming applications; a node starts joining procedure to a P2P system by communicating with the tracker. In addition to standard bootstrapping process, a timestamp is added to bootstrap messages in order to reveal exact login time of the node. This information will further be used by gossip protocol in order to detect up-to-date gossip messages.

After obtaining a list of available nodes, the node selects a set of them to construct membership list and establishes mutual partnership with some nodes from the subset of membership list. This is similar to the approach presented in [1]. The underlying gossip protocol is responsible for disseminating data about nodes in the system since node churn causes frequent changes in the set of nodes in the P2P system. A node receiving a gossip message compares the timestamp value in the message with the timestamp value of sender node in order to detect whether the message is recent and discard the message if is not. Each node periodically exchanges the information about joining and leaving nodes with the nodes in the membership list.

B. CoolStreaming-like Parent Selection

In push-pull based mesh streaming, the video data is partitioned into K substreams and each node subscribes to one or more parents to receive these K substreams. In CoolStreaming, the nodes in the system periodically exchange their buffer maps which show buffer position for each substream between them and their partners. Parents are selected from the set of partners according to their buffer maps. During streaming, in order to detect insufficient video retrieval for substream j, buffer maps related to node, node's parents and partners are periodically controlled by using the inequalities given in (1) and (2) [1]. The notations used in the inequalities are listed in Table 1.

$$\max\{|\ B_{P_j} - B_i\ |, i \le K\} < T_s \tag{1}$$

$$\begin{aligned} &A = \max\{B_{Q_i}, i \le K, Q \in\ partners\} \\ &A - B_{P_j} < T_p \end{aligned} \tag{2}$$

TABLE I. THE EXPLANATION OF NOTATIONS USED IN THE INEQUALITIES

Notation	Explanation
P_j	The parent of the *j.* substream received
B_i	Buffer map of the *i.* substream of its own
B_{P_j}	Buffer map of the *j.* substream recevied from P_j
T_s	The threshold related to buffer maps of a substream of the node and its parents
T_p	The threshold related to buffer maps of a substream of the node's parent and partners
K	Total number of substreams

Note that these inequalities given in (1) and (2) provide node to detect congestion or inadequate bandwidth of a link in the path between source and itself for substream *j*. The parents fail to meet these requirements are changed with any node from the partners which buffer map for requested substream is consistent with the node's buffer map [1]. To provide this consistency, candidate parents are required to meet inequalities given (2) and (3) which is slightly different version of the expression (1). In the expression, *j* refers to the substream that is not received at adequate bitrate.

$$B_{Q_j} - \max\{B_i, i \le K\} \quad < T_s \tag{3}$$

In CoolStreaming, if more than one node meet these criteria given in (2) and (3), then one of them is randomly selected as new parent. In our proposed scheme, instead of this randomization in parent reselection, we use a method based on Q-learning model which is discussed in the next subsection. We also highlight and elaborate some of the details related to the methods used in CoolStreaming. In this context, a mechanism for a peer which cannot find a suitable parent in the parent selection process is developed. If there is no candidate that holds the inequality (2) and (3), the peer which does not have a parent for at least one substream, starts a timer to repeat the parent selection process in order to shorten the time to connect a new parent. After two unsuccessful trials, peer removes some of its partners from the partnership list and sends partnership requests to the peers in the membership list. An additional method is implemented in case of a decrease in the number of partners in the partnership list. To overcome this situation we identify a threshold for partnership list size. If the number of the partners in the list is smaller than threshold value (50% of max list size) the peer sends partnership requests to peers in the membership list.

C. Learning Model

In this study, we propose a novel model based on reinforcement learning for parent selection in a P2P video streaming system. In reinforcement learning, based on its underlying Markov process, a node in a state s_t at time t takes an action $a_t \in A(s_t)$, where $A(s_t)$ is the set of available actions in state s_t and moves to next state s_{t+1}. The node also receives a reward (r_{t+1}) determined by the reward function of the model related to selected action, a_t. If state transition probabilities are not known, Q-learning technique [17] can be used. In Q-learning, for s=s_t, the obtained reward due to the selection of an action a=a_t is added to Q(s,a) value according to the update function given in (4). The selection probability of any action in state s_t is determined based on these Q-values. In the equation, η and γ represents the learning rate and the discount rate, respectively.

$$\hat{Q}(s_t,a_t) = \hat{Q}(s_t,a_t) + \eta(r_{t+1} + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1},a_{t+1}) - Q(s_t,a_t)) \tag{4}$$

In our model, the state space (S) equals the Cartesian product of the number of substreams (K) and the number of possible status υ_m, the indicator of the aforementioned inequalities related to the parent providing m^{th} substream. That is because S = K x υ_m, state space is limited by a value of 2x*K*. υ_m is defined in (5).

$$\upsilon_m = \begin{cases} 0, \textit{ if the parent providing } m^{th} \textit{ substream holds} \\ \qquad \textit{inequalities given in (1) and (2)} \\ 1, \textit{ otherwise}. \end{cases} \tag{5}$$

Selecting a new parent for the related substream *m* is defined as an action and the set of nodes constructing A, i.e. all possible nodes, that can be selected as new parents, correspond to the set of nodes which passes T_s, T_p related inequivalencies. Even if the all partners meet these criteria and take place in the set of A, the number of the actions is limited by the size of partnership table. The upper bound of the time complexity is limited by the size of the partnership table.

After parent selection, the received reward via selected new parent is calculated by the function given in (6). This calculation is done when parent check timer expires, and the related Q(s,a) value is updated by using cumulative reward obtaining until the time when new parent is selected for the same substream. New parent selection may be done if the current parent fails to meet T_s, T_p related inequalities or if the current parent leads to experience unacceptable playback lag. In the proposed system, quality of received video and playback lag depend on bitrate and then latency. Since the quality of video is more crucial than the playback lag, the rewards are calculated by considering cumulative received bitrate. The value of α in (6) has been chosen to smooth the value of received bitrate.

$$r = \alpha * (\textit{Cumulative received bitrate}) \tag{6}$$

We use the softmax function [18] given in (7) to determine the probability of selecting an action in state*s*.

$$P(a \mid s) = \frac{\exp[Q(s,a)/T]}{\sum_{b \in A} \exp[Q(s,b)/T]} \tag{7}$$

When a node joins to the system, Q-values are initially set to zero since it has no prior knowledge about the system. Each node starts to gain experience by selecting parents for the substreams and updates related Q-values. In the exploration phase, *T* value given in (7) is large enough. Thus, the node selects a new parent among candidate partners whose P values are likely equal, although the Q-values can vary in a large interval. T value is gradually decreased over time in order to ensure that the node selects parents with higher Q-values.

IV. SIMULATIONS AND PERFORMANCE EVALUATIONS

In this section, the test results running over Network Simulator 3 (ns3) [19] environment of the proposed model explained in the previous section are given.

A. Simulation Setup

The proposed P2P system is tested in ns3 environment over a network topology generated by BRITE Barabasi model [20]. We run the simulations for different size of

network, and obtain performance results such as received video bitrate and playback continuity. In the simulations, there is only one source with upload capacity of 3 Mbps. The bandwidth distribution of other nodes is as follows: 50% of the nodes have an upload capacity lower than 500 Kbps, 90% of the nodes have an upload capacity lower than 800 Kbps and all of the nodes have an upload capacity lower than 3000 Kbps.

In order to implement peer churn, nodes in the system join and leave the system according to on/off model [1]. Based on this model, time samples are obtained from the exponential distribution with the mean value of 1000 seconds and 400 seconds for on and off period, respectively. In each simulation, nearly 10% of the nodes in the system are selected as robust nodes which tend to stay longer in the system. Some of these nodes may not leave after joining the system. All simulations run over 1800 seconds.

After simulation starts, each node joins to the system in every 5 msec. Each node subscribes to the set of parents in order to obtain K substreams. In all simulations, K equals to 4 and each substream bitrate equals to 72 Kbps. All video and control packets are sent over TCP. Buffer maps are sent to the nodes in the partnership table in every 5 sec. Video packets received after their playout time are discarded. In every 20 sec, parent check timer expires to control the buffer map for each stream in order to detect any problems with the substream retrieval. T_s and T_p values explained in the previous section equal to 500 packets. The values of parent check timer period, T_s and T_p are selected according to the values given in [1].

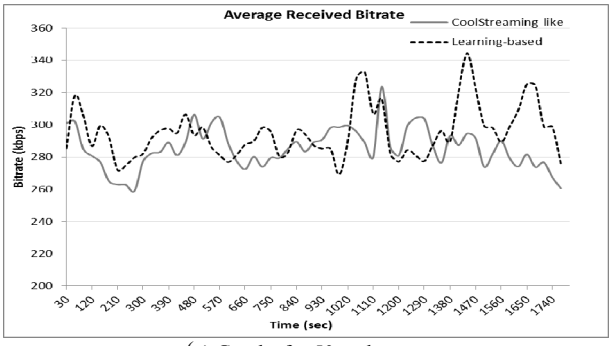
If a node cannot find a new parent after disconnecting from its parent, it removes some nodes from its partnership table and adds a new set of partners. This procedure is repeatedly executed in every 5 sec until the node connects a new parent. If the problem continues for 20 seconds, the node resets itself and joins the system again. The size of membership table equals to 10, the size of partnership table equals to 5 for network sizes of 50 and 100. For the networks consisting of more than 100 nodes, the size membership table and partnership table equals to 15 and 10, respectively.

B. Simulation Results

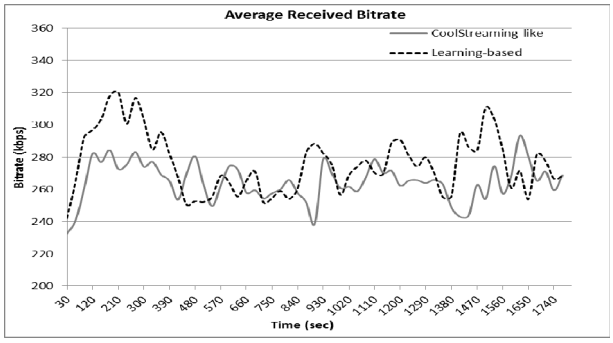
In order to observe the performance of the proposed learning model, we run simulations for both systems with CoolStreaming-like parent selection and with learning-based parent selection and give comparative results. All parameters for both systems were set the same for fair comparison. Since higher reward is received by selecting parents providing sufficient bitrate due to the reward function, simulations show that the learning model improves average video bitrate received by peers. Average received bitrate values given in Fig. 1 are obtained by calculating bitrate received by each online peer in every 30 seconds as soon as the node starts to receive video. Note that this value represents the received packet payload which is buffered at peers. In the figure, it is seen that learning-based model provides some improvement for all simulations having different size of network topology. The highest bitrate observed in the simulations are obtained from the systems consisting of 50 nodes, since almost 20% of peers in these systems are connected the server directly and the most part of the other nodes are connected to robust nodes. However, with the growing network size, we observe decrease in bitrates since it is more challenging to find parents which can send video properly. But especially for the systems consisting of more than 100 nodes, learning-based parent selection model gives better results when compared to CoolStreaming-like parent selection model.

Although we generally observe an improvement in received bitrate with the learning-based model, some degradation in bitrates can also be observed with this model. For instance, received bitrates between the 1560th seconds and 1660th seconds in Fig. 1.b can be an example for such degradation. This is because there may not be any good candidate parents in the partnership table of a node, so the node may not find any suitable parent to connect even it has sufficient experience about the nodes in the system. Moreover, the seesaw pattern of the graphics in Figure 1 depends on the peer churn activity in the system.

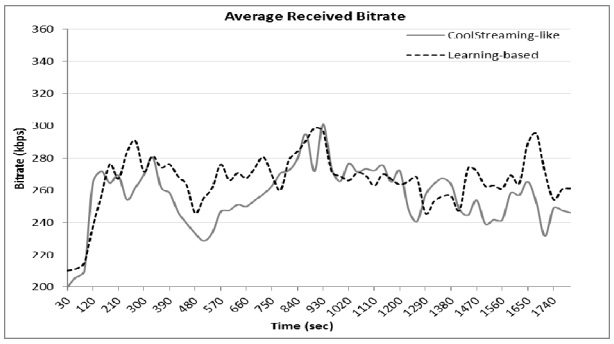
In Fig. 2, average continuity indexes measured by each node in the system are summarized in the CDF graphs. Continuity index measurement starts after buffering period is completed. Even though it is correlated to average received bitrates given in Fig. 1, graphs in Fig. 2 does not contain the results of buffering time. As it is observed from the previous figure, the highest continuity indexes are obtained in the network with 50 nodes. We observe that nodes in the system with learning-based approach experience higher continuity index than the nodes in the system with CoolStreaming-like approach, especially in large networks with more than 100 nodes. In Fig. 2.c, it is seen that 70% of the nodes have the continuity index value of higher than 0.85 with learning-based approach while only 35% of the nodes in the system have the same continuity index value with CoolStreaming-like approach. In Fig. 2.d, it is observed that 50% of the nodes have the continuity index values between 0.7 and 0.8 without learning, while the percentage of nodes equal to zero with learning-based approach. These graphs show that learning-based approach in parent selection can significantly improve the performance of P2P video streaming applications.



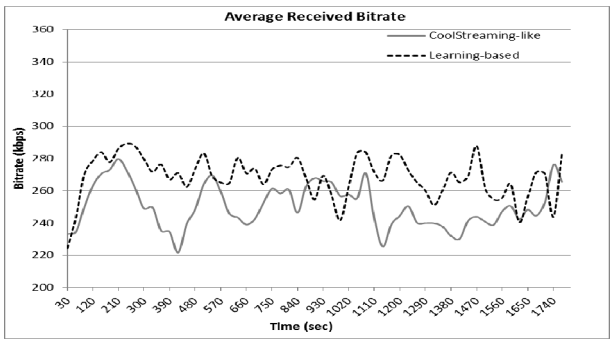
(a) Graphs for 50 nodes



(b) Graphs for 100 nodes



(c) Graphs for 150 nodes



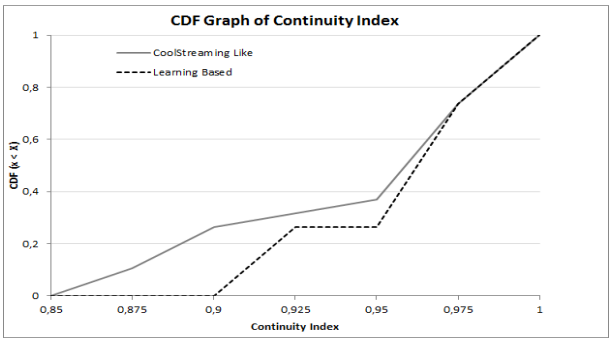
(d) Graphs for 200 nodes

Figure 1. Average received bitrate values of different size of networks.

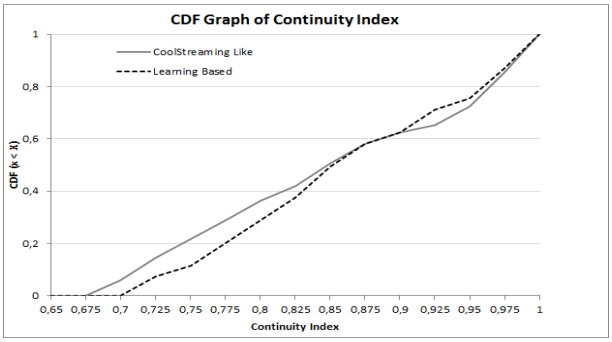
Moreover, we observe the number of parent changes for all simulations. While this parameter is similar for small networks (e.g. networks with 50 and 100 nodes), there is a decrease in number of parent changes with learning for larger networks. The reason is that when network becomes larger, the probability of finding suitable parents decreases with randomized approach. The average parent change counts per node for different size of networks are listed in Table 2.

TABLE II. AVERAGE PARENT CHANGE COUNTS PER NODE

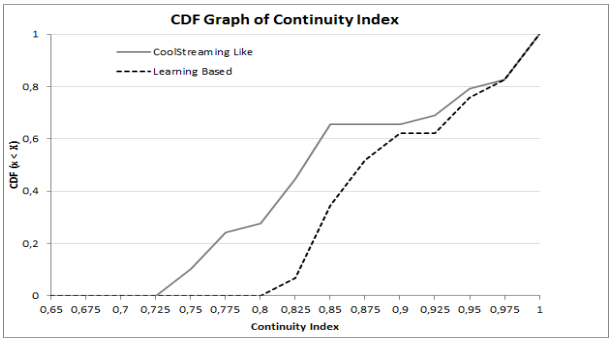
Network Size	Parent Change Count	
	CoolStreaming-like	***Learning-based***
50	4.78	5
100	21.52	21.58
150	37.19	31.71
200	34.51	29.52



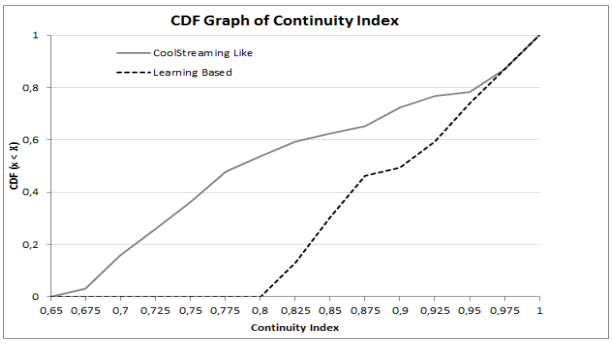
(a) Graphs for 50 nodes



(b) Graphs for 100 nodes



(c) Graphs for 150 nodes



(d) Graphs for 200 nodes

Figure 2. CDF graph of continuity index for different size of networks.

V. DISCUSSIONS

One of the main challenges of P2P systems is to provide scalability. The proposed learning model can be easily implemented for a medium sized network. For larger networks, i.e. the number of peers reaching to thousands within the system, peers can be clustered as in [1] and Q-tables can be constructed for the peers within the same clusters and this approach limits the size of Q-tables to the size of a cluster.

For push-pull systems, the number of substreams is a crucial parameter that directly affects received video bitrate. Clearly, increasing substream count causes increase in message complexity since each substream introduces additional control messages. In [21], it is reported that if the number of substreams is less than 6 then the continuity index falls under 70% for heterogeneous networks. However, we obtain continuity index values higher that 70% which shows that our improvements increase the performance of the system remarkably even if there is no super peers used in the experiments. Besides, the performance of the proposed system can be raised by increasing the substream count.

In the experiments, it is seen that the performance obtained by the learning model is higher for large sized networks consisting 150 and 200 nodes than that of small sized networks. The possibility of selecting a good parent in CoolStreaming-like parent selection is $1/\varepsilon$, where ε is the number of all partners that can be selected. But if a partner is stable and has high upload bandwidth, the possibility of selecting this partner is higher than $1/\varepsilon$. If network size increases, ε also increases which causes the decrease in selecting good parents with CoolStreaming-like parent selection, while this possibility is high in our learning model.

VI. CONCLUSION

In this paper, we propose a parent selection method using Q-learning technique for P2P video streaming applications. The aim of the proposed Q-learning model is to select an efficient and a good parent by having no prior knowledge of the network topology and without the need of any centralized mechanism. The proposed learning model is efficient and easy to implement. Since the space and time complexity are limited, it is suitable for real-time live video streaming applications. With a large number of simulations running on ns3, our model improves received bitrate by peers in the system when compared to CoolStreaming-like parent selection method.

Application of the proposed Q-learning based technique and the increase in node amount provided that both bitrate and continuity index values were increased comparing to CoolStreaming. It has been examined that parent selection by Q-learning increased bitrate values by around 3% for both 50 and 100 nodes and around 4% and 7% for 150 and 200 nodes respectively. Increase percentage in CI is around 1% for both 50 and 100 nodes and 4% and 9% for 150 and 200 nodes respectively. Furthermore, limitation of the state space (S) with 2*K (where K is the number of substreams in the system) in the proposed Q-learning based technique demonstrates how the proposed technique is appropriate for large-scale P2P video streaming applications.

As a future direction, we plan to implement our learning model to the partner selection process for push-pull systems. If the nodes in the system use the previous experiences and data about other peers when constructing partnership table, it is expected that more improvement can be observed.

REFERENCES

[1] S. Xie, B. Li, G.Y. Keung, and X. Zhang, “CoolStreaming: Design, Theory, and Practice,” IEEE Transactions on Multimedia 9, pp. 1661-1671.

[2] PPLive, http://www.pptv.com/

[3] Z. Liu, C. Wu, B. Li, and Shuqiao Zhao, “UUSee: large-scale operational on-demand streaming with random network coding,” in Proc. INFOCOM, 2010, NJ, USA, pp. 2070-2078.

[4] C. Szepesvári, Algorithms for Reinforcement Learning, Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

[5] M. Adler, R. Kumar, K.W. Ross, D. Rubenstein, T. Suel, and D.D. Yao, "Optimal peer selection for P2P downloading and streaming", in Proc. INFOCOM, 2005, pp. 1538-1549.

[6] A. Habib and J. Chuang, “Service differentiated peer selection: an incentive mechanism for peer-to-peer media streaming,” IEEE Trans. Multi. vol. 8(3), pp. 610-621, September 2006.

[7] S.S. Savas, A.M. Tekalp, and C.G. Gurler, “Adaptive multi-view video streaming over P2P networks considering quality of experience,” in Proc. of ACM SBNMA, 2011, NY, USA, pp. 53-58.

[8] Z. Liu, Y. Shen, S.S. Panwar, K.W. Ross, and Y. Wang, "P2P Video Live Streaming with MDC: Providing Incentives for Redistribution", in Proc. ICME, 2007, pp. 48-51.

[9] Y.T.H. Li, D. Ren, S.G. Chan, and A.C. Begen, "Low-delay mesh with peer churns for peer-to-peer streaming," ICME, 2009, pp. 1546-1547.

[10] Y. Xu, C. Zhu, W. Zeng, and X.J. Li, “Multiple description coded video streaming in peer-to-peer networks,” Image Commun. vol. 27(5), pp. 412-429, May 2012.

[11] Z. Liu, C. Wu, B. Li, and S. Zhao, "Distilling Superior Peers in Large-Scale P2P Streaming Systems," INFOCOM, 2009, pp. 82-90.

[12] D. Niu, B. Li, and S. Zhao, "Self-diagnostic peer-assisted video streaming through a learning framework," in Proc. ACM Multimedia, 2010, pp. 73-82.

[13] I. Ullah, G. Doyen, G. Bonnet, and D. Gaïti, “A Bayesian approach for user aware peer-to-peer video streaming systems,” Image Commun. 27(5), pp. 438-456, May 2012

[14] H. Yu, D. Zheng, B.Y. Zhao, W. Zheng, “Understanding user behavior in large-scale video-on-demand systems”, ACM SIGOPS Operating Systems Review, vol. 40(4), pp. 333–344, 2006.

[15] R. Izhak-Ratzin, H. Park, and M.V.D. Schaar, "Reinforcement learning in BitTorrent systems," in Proc. INFOCOM, 2011, pp. 406-410.

[16] M. Sayit, and O. Sonmez, “Reinforcement learning for peer to peer video streaming applications,” in Proc. SIU, 2012, pp. 1-4.

[17] D.P. Bertsekas, Dynamic Programming and Optimal Control, Vol. II (3rd ed.), Athena Scientific, 2007.

[18] E. Alpaydın, Introduction to Machine Learning, Machine Learning, The MIT Press, 2004.

[19] “The NS-3 Network Simulator,” http://www.nsnam.org/

[20] A.L. Barabasi and R. Albert, “Emergence of Scaling in Random Networks,” Science 286, pp. 509–512, October 1999.

[21] B. Li, S. Xie, Y. Qu, G. Y. Keung, C. Lin, J. Liu, X. Zhang, Inside the New Coolstreaming: Principles, Measurements and Performance Implications, in Proc. INFOCOM, 2008, pp. 1031-1039.