Online Learning in BitTorrent Systems

Rafit Izhak-Ratzin, Hyunggon Park, *Member*, *IEEE*, and Mihaela van der Schaar, *Fellow*, *IEEE*

Abstract—We propose a BitTorrent-like protocol based on an online learning (reinforcement learning) mechanism, which can replace the peer selection mechanisms in the regular BitTorrent protocol. We model the peers' interactions in the BitTorrent-like network as a repeated stochastic game, where the strategic behaviors of the peers are explicitly considered. A peer that applies the reinforcement learning (RL)-based mechanism uses the observations on the associated peers' statistical reciprocal behaviors to determine its best responses and estimate the corresponding impact on its expected utility. The policy determines the peer's resource reciprocations such that the peer can maximize its long-term performance. We have implemented the proposed mechanism and incorporated it into an existing BitTorrent client. Our experiments performed on a controlled Planetlab testbed confirm that the proposed protocol 1) promotes fairness and provides incentives to contributed resources, i.e., high capacity peers improve their download completion time by up to 33 percent, 2) improves the system stability and robustness, i.e., reduces the peer selection fluctuations by 57 percent, and (3) discourages free-riding, i.e., peers reduce their uploads to free-riders by 64 percent as compared to the regular BitTorrent protocol.

Index Terms—Peer-to-peer (P2P), BitTorrent, reinforcement learning, foresighted resource reciprocation strategy

1 Introduction

Peer-to-peer (P2P) content sharing protocols dominate the traffic on the Internet, and have become an important part in building scalable Internet applications [1]. The P2P protocols are used by a variety of Internet applications such as content distribution, voice over IP, and streaming multimedia P2P applications [2], [3], [4].

In P2P content distribution systems, fairness among peers is an important factor, as it encourages peers to actively collaborate in disseminating content, which can lead to an improved system performance. However, even BitTorrent [5], one of the most popular protocols used in P2P content distribution, does not provide fair resource reciprocation, particularly for node populations having heterogeneous upload bandwidths [6], [7], [8]. This is because the Tit-for-Tat strategy implemented in BitTorrent only exploits a short-term history for making upload decisions. More specifically, upload decisions are made based on the most recent observations of the resource reciprocation. This also implies that the upload decisions are short backward looking but not forward looking, i.e., the decisions are not *foresighted*. Thus, a peer can keep following the Tit-for-Tat policy only if it continuously uploads pieces of a particular file to its associated peers and as long as it receives pieces of interest in return. However, this is not always feasible, irrespective of peers' willingness to cooperate, as they may not always have pieces of interest to the other peers [9]. Moreover, such behavior is still perceived as a lack of cooperation for interacting peers. In addition, it has been shown that BitTorrent systems do not effectively cope with selfish peers' behaviors such as free-riding [10], [11], [12], because of their built-in optimistic unchoke mechanism. While the optimistic unchoke mechanism enables peers to continuously discover better peers to reciprocate resources, it can provide a major opportunity for selfish peers to obtain data without uploading in return. This mechanism may also lead to unfairness in the system, as it forces high-capacity peers to interact with low-capacity peers.

Unlike the approaches that use short-term observation history, reputation-based schemes have been proposed to overcome the limitations of Tit-for-Tat and optimistic unchoke mechanisms by exploiting global histories (e.g., [13], [14]). However, in order to maintain such a global history across peers, these approaches require a significant amount of information exchange among peers, which increases the communication overhead. Moreover, the reliability of global history can be unclear as peers may exhibit different reciprocation behaviors with different peers. Alternatively, the long-term local (or private) history of upload behaviors with associated peers is used in several other reputation-based approaches such as [9], [15], [16]. While these approaches can reduce the communication overheads, the focus of these systems is still on maximizing the *immediate* utility, which may be less desirable than maximizing the *long-term* utility, as peers can repeatedly interact with each other over a long period of time.

In this paper, we model the peers' interactions in the BitTorrent-like network as a repeated stochastic game—repeated interactions (i.e., reciprocating resources) among several players (i.e., peers) in which a player takes actions (i.e., unchoke peers) so as to maximize long-term reward (i.e., cumulative download rates). The underlying state of the environment changes stochastically and is contingent upon the decisions of the players. In our model, peers can adopt an online learning strategy, specifically, reinforcement learning

- *R. Izhak-Ratzin is with Palo Alto Networks, 1498 S. Mary Avenue, Sunnyvale, CA 94087. E-mail: rratzin@paloaltonetworks.com.*
- *H. Park is with the Department of Electronics Engineering, Ewha Womans University, 52 Ewhayeodae-gil (11-1 Daehyun-dong), Seodaemun-gu, Seoul 120-750, Republic of Korea (South Korea). E-mail: hyunggon.park@ewha.ac.kr.*
- *M. van der Schaar is with the Electrical Engineering Department, University of California (UCLA), 58-109 Engineering IV Building, 420 Westwood Plaza, Los Angeles, CA 90095. E-mail: mihaela@ee.ucla.edu.*

Manuscript received 16 Nov. 2010; revised 30 July 2011; accepted 24 Feb. 2012; published online 9 Mar. 2012.
Recommended for acceptance by Y. Liu.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2010-11-0678.
Digital Object Identifier no. 10.1109/TPDS.2012.90.

(RL) [17], [18], to make their upload decisions. By observing and storing the information about the reciprocation behaviors of their associated peers, peers can explicitly consider the strategic behaviors of peers. The peers applying the RL-based strategy with this information can estimate their future expected rewards, and then can determine their best responses that maximize the long-term future expected utility. Therefore, the RL-based peer selection mechanism can replace both the Tit-for-Tat and the optimistic unchoke mechanisms in the regular BitTorrent protocol. Note that the RL algorithm enables peers to improve their peer selection strategies-based solely on the knowledge of their past interactions, but not based on the knowledge of the complete reciprocation behaviors of the peers in the entire network.

We implemented our proposed protocol on top of an actual BitTorrent client, and performed extensive experiments in a controlled Planetlab testbed. The proposed protocol consists of two main processes, which are the *Learning Process* and the *Policy Finding Process*. The learning process provides updated information about statistical behaviors of the associated peers' resource reciprocation. The policy finding process computes the peer selection policy based on the RL. Then, its subprocess, the *Decision Process*, determines the associated peers that will be unchoked and choked during every rechoke period based on the policy. The proposed algorithm is executed through policy modifications to existing clients with no changes to the BitTorrent protocol. Our protocol does not demand full adoption or sparse adoption of the RL-based peer selection mechanism (as in [6]) and can be run by any number of peers in a BitTorrent-like network.

The proposed protocol provides several advantages compared to the regular BitTorrent protocol.

- It discourages free-riding by limiting uploads to noncooperative peers.
- It promotes cooperative resource reciprocation among high-capacity peers.
- It improves the system robustness by minimizing the impact of free-riding on the contributing peers' performance.
- It improves the stability of the peer selection mechanism.

Note that the scope of our protocol is on the support of streaming (not real-time) data transmission. In this type of protocol, the requested content needs to be completely downloaded before it is displayed. Thus, the download order is not important, but the overall time required for completely downloading the content is important. Note that, however, the proposed protocol can be easily adapted to on-demand media streaming applications using existing techniques such as [19], [20].

For reader's convenience, a summary of notations is provided in Appendix A and related work is studied in Appendix B of the supplemental material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2012.90.

2 Overview of BitTorrent

In this section, we briefly overview the BitTorrent protocol [5]. First, before the content distribution process begins, the content provider divides the possessed data content into multiple *pieces*, or *chunks*. Then, the provider creates a *metainfo file*, which contains information necessary to initiate the content downloading process. The metainfo file includes the address of the *tracker*, which plays the role of coordinator that facilitates peer discovery. A client downloads the metainfo file before joining a *torrent* (or *swarm*)—a group of peers interested in a particular content. Then, it connects the tracker to receive a *peer set*, which consists of randomly selected peers currently exchanging the same content. The peer set may include both *leechers*, peers that are still downloading content pieces, and *seeds*, peers that have the entire content and upload it to other peers. The client can then connect and exchange (or, *reciprocate*) its content pieces with its *associated peers*—the peers in its peer set.

While reciprocating content pieces, each leecher determines a set of leechers among its peer set to whom it can upload its content pieces. The peer selection is determined by *choking mechanisms* which determine the *choking decisions*. BitTorrent leechers adopt two choking mechanisms: the *Tit-for-Tat* resource reciprocation mechanism and the *optimistic unchoke* mechanism. The Tit-for-Tat mechanism prefers the leechers that upload their pieces at the highest rate among the associated leechers. Specifically, every 10 seconds (or *rechoke period*), a leecher checks the current download rates from its associated leechers and selects the leechers that are uploading their pieces at the highest rates. Then, the leecher uploads only to the selected associated leechers, while choking (i.e., blocking downloads to) the rest of them during the rechoke period.

The available upload bandwidth is equally divided into the unchoked peers. The optimistic unchoke mechanism reserves a portion of the available upload bandwidth to provide pieces to peers that are randomly selected. The purpose of this mechanism is to enable the leechers to continuously discover better peers to associate itself with, and bootstrap newly joining leechers into the Tit-for-Tat mechanism. Optimistic unchokes are randomly rotated among the associated peers, typically once every three rechoke periods, allowing enough time for leechers to demonstrate their cooperative behaviors. The number of unchoked peers (slots) may vary depending on specific implementation, and it can be fixed or dynamically changed as a function of the available upload bandwidth.

Seeds deploy different choking mechanisms, as they have already completed the content download. The most common implementation is based on a round-robin schedule, which aims to distribute data uniformly. The round-robin schedule is also deployed in our implementation. More details can be found in [21].

3 RL for Resource Reciprocation in BitTorrent Systems

3.1 Overview

Peers in BitTorrent-like systems often make repeated decisions to select unchoked peers in their dynamically changing environment. Thus, we model the evolution of the peers' interactions across the various rechoke periods as a repeated stochastic game, which is played over a long period

of time. In each time slot (i.e., rechoke period), a peer can observe the download rates from its associated peers, which determine the *state* of the peer. The total download rates in the state thus can thus be computed, which is the *reward* for the state. The peer then decides its own upload bandwidth allocations to its selected associated peers independently and simultaneously in each rechoke period. This is the *action* selection of the peer. The upload bandwidth decision of the peer triggers new upload bandwidth allocations of its associated peers, which determines the next state of the peer. Thus, the reward and state transition are contingent upon other peers' states and actions. These reciprocal interactions are observed and stored by each peer and are used in the proposed RL-based approach.

3.2 Formulation of RL-Based Approach

The resource reciprocation among peers can be formally expressed as a tuple, $\langle \mathbf{I}, \mathcal{S}, \mathcal{A}, P, R \rangle$, where $\mathbf{I}$, $\mathcal{S}$, $\mathcal{A}$, P, and R denote the set of peers, the set of state profiles, the set of action spaces, a state transition probability function, and a reward function, respectively. For M peers in the game, i.e., $\mathbf{I} = \{1, \ldots, M\}$, $\mathcal{S} = \mathbf{S}_1 \times \cdots \times \mathbf{S}_M$ and $\mathcal{A} = \mathbf{A}_1 \times \cdots \times \mathbf{A}_M$, where $\mathbf{S}_j$ and $\mathbf{A}_j$ are the state space and action space of peer j, respectively. A state transition probability, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$, maps a state profile $\mathcal{S}(t) \in \mathcal{S}$ at time t to the next state profile $\mathcal{S}(t+1) \in \mathcal{S}$ at time $t+1$ given the joint action $\mathcal{A}(t) \in \mathcal{A}$. Finally, a reward function, $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}_+^M$, maps a state profile $\mathcal{S}(t) \in \mathcal{S}$ at time t, and the joint action $\mathcal{A}(t) \in \mathcal{A}$, to a vector with each element being the reward to a particular peer. All the previous states, actions and rewards of the peers in the network are available to the peers for the RL-based approach. This formulation can be found in our prior work [22].

3.2.1 State Space of Peer j ($\mathbf{S}_j$)

The state of peer j represents the set of resources received (i.e., the uploading behavior) from the peers associating with peer j (denoted by C_j). Thus, this captures the uploading behavior of its associated peers. The upload rates from peer $i \in C_j$ to peer j at time t are denoted by $L_{ij}(t)$. In our implementation, an uploading rate of peer i observed by peer j is represented by $s_{ij} \in \{0, 1\}$, where $s_{ij} = 1$ if $L_{ij} > \theta_j$ and $s_{ij} = 0$ otherwise. θ_j is a predetermined threshold of peer j.[1] Thus, $\mathbf{S}_j$ with N associated peers in C_j is expressed as

$$\mathbf{S}_j = \big\{(s_{1j}, \ldots, s_{Nj}) \big| s_{ij} \in \{0, 1\} \text{ for all } i \in C_j\big\}.$$

3.2.2 Action Space of Peer j ($\mathbf{A}_j$)

The action of peer j represents the set of decisions on its peer selection. An action of peer j to peer i, a_{ji}, at time t is defined as $a_{ji}(t) \in \{0, 1\}$, where $a_{ji}(t) = 0$ if peer j chokes peer i and $a_{ji}(t) = 1$ otherwise. Thus, the corresponding action space of peer j is

$$\mathbf{A}_j = \big\{(a_{j1}, \ldots, a_{jN}) \big| a_{ji} \in \{0, 1\} \text{ for all } i \in C_j\big\},$$

where $\mathbf{A}_j(t) \in \mathbf{A}_j$ denotes an action space at time t. In our implementation, $N_u (\leq N)$ peers are simultaneously unchoked and the upload bandwidth B_j of peer j is equally divided into its associated peers, i.e., an unchoked peer i obtains $L_{ji}(t) = B_j / N_u$ from peer j.

3.2.3 State Transition Probability of Peer j

A state transition probability represents the probability that an action $\mathbf{A}_j(t) \in \mathbf{A}_j$ of peer j in state $\mathbf{S}_j(t) \in \mathbf{S}_j$ at time t will lead to another state $\mathbf{S}_j(t+1) \in \mathbf{S}_j$ at time $t+1$. This can be expressed as

$$P_{\mathbf{A}_j(t)}(\mathbf{S}_j(t), \mathbf{S}_j(t+1)) = \Pr(\mathbf{S}_j(t+1) | \mathbf{S}_j(t), \mathbf{A}_j(t)).$$

The state transition probability functions can be estimated based on past interactions, which are stored in a transition table. While we deploy an empirical frequency based algorithm to estimate the state transition probability function (presented in Section 4.2), alternative algorithms (e.g., [23], [24]) can also be deployed depending on application-specific implementation constraints such as implementation complexity, estimation time, estimation accuracy, etc.

3.2.4 The Reward of Peer j (R_j)

The reward of a peer in a state is the sum of the estimated download rates from all of its associated peers. The reward of peer j from state $\mathbf{S}_j(t) \in \mathbf{S}_j$ can thus be expressed as

$$R_j(\mathbf{S}_j(t)) = \sum_{i \in C_j} L_{ij}.$$

3.2.5 RL-Based Policy (π_j)

The policy π_j obtained by RL can provide actions $\mathbf{A}_j(t)$ for peer j in state $\mathbf{S}_j(t)$ at time t, i.e., $\pi_j : \mathcal{S}_j \rightarrow \mathcal{A}_j$, defined as $\mathbf{A}_j(t) = \pi_j(\mathbf{S}_j(t))$. These actions enable peer j to maximize its cumulative discounted expected reward. This is defined for a peer j in state $\mathbf{S}_j(t)$ at time $t = t_c$ given a discount factor γ_j as

$$R_j^f(\mathbf{S}_j(t_c)) \triangleq \sum_{t=t_c+1}^{\infty} \gamma_j^{(t-(t_c+1))} \cdot R_j(\mathbf{S}_j(t)). \quad (1)$$

The value of γ_j may represent the importance of the future expected rewards and can be determined based on the peers' past experiences, reputation of their associated peers, network conditions, etc. [23], [25]. Therefore, the policy is deployed as a peer selection algorithm in our implementation. While the policy π_j can be obtained using well-known methods such as value iteration and policy iteration [18], the environment dynamics keep changing in practice, and thus, the policy needs to be updated frequently. This may require high computational complexity. Hence, it is important to reduce the complexity of finding the policy, such that the proposed algorithm can be efficiently deployed.

4 Protocol Design and Implementation

4.1 Overview

The protocol consists of two main processes running in parallel, which are the *learning process* and the *policy finding*

1. While $s_{ij} \in \{0, 1\}$ in our implementation for low computational complexity, the granularity of the state can be easily extended.

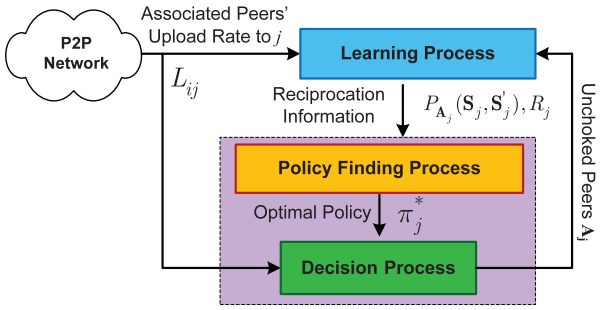

Fig. 1. Main processes in the proposed protocol design.

process. The learning process gathers online information about statistical behaviors of the associated peers' resource reciprocation. As the peers' reciprocation behaviors are not foretold, peers are required to construct the transition function and the rewards of the associated peers. The other main process is the policy finding process that computes the policy based on RL and determines the choking decisions. This process needs to be running during the entire downloading process as the changes in peers' reciprocation behaviors (identified by the learning process) can result in the policies obtained in the previous time slots becoming outdated. Its subprocess, the *decision process*, determines the decisions on peer selection in each rechoke period based on the policy and the observed state. The protocol design is depicted in Fig. 1.

The proposed RL-based client is implemented on top of the *Enhanced CTorrent* client, version 3.2 [26]. We enhance the original client such that our client can operate in *RL-enhanced mode*, where it reciprocates its resources using the proposed RL-based mechanism, or in *regular mode*, where it reciprocates its resources based on the regular BitTorrent peer selection mechanism.

4.2 The Learning Process

Due to the unannounced information, network scalability constraints, time-varying network dynamics, etc., each peer needs to estimate (or *learn*) the other peers' states, its rewards and the state transition probabilities using the past observations of its competing peers. In the proposed protocol, RL is deployed. Thus, each peer needs to update the above information regularly throughout the learning process, while downloading content from its associated peers.

The learning process consists of two main methods that compute the estimated reward and the state transition probability, which is depicted in Fig. 2.

4.2.1 Computing Reward

The reward of peer j is the total estimated download rates from the peers associating with peer j. Based on the availability of information about past resource reciprocation history, different reward calculation methods are applied.

If peers have reciprocated their resources with peer j, referred to as *peers with reciprocation history*, peer j estimates their upload rates based on the weighted average of the past samples of upload rates. Specifically, peer j estimates the upload rates L_{ij} of peer $i \in C_j$ based on recently observed resource reciprocation L^o_{ij} as

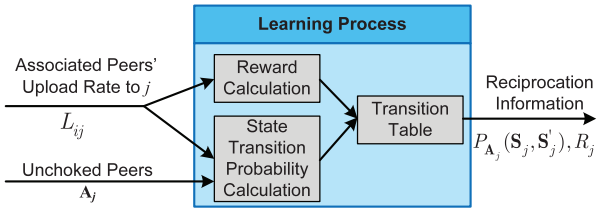

Fig. 2. The learning process.

$$L_{ij}(t+1) \leftarrow \alpha_j \cdot L^o_{ij}(t+1) + (1-\alpha_j) L_{ij}(t), \tag{2}$$

where α_j denotes the weight for most recent resource reciprocation, and is set as $\alpha_j = 0.5$ (for more details of α_j and weighted average of the sample in (2), see Appendix C of the online supplemental material).

If peers have not reciprocated yet their resources with peer j, which are referred to as *peers without resource reciprocation history*, peer j initializes the information about such peers by optimistically estimating that they will reciprocate their resources with high probability and high upload rate. Specifically, the initial estimated upload rate is set to be the highest upload rate L^{max}_{ij} that is predetermined in the P2P network, i.e., $L_{ij} \leftarrow L^{max}_{ij}$, and the probability of reciprocation with j is initiated to 1. This enables peer j to efficiently discover additional peers and bootstrap newly joining peers, which can lead to a system performance improvement.

Peer j needs to continue updating the initially assumed reward in every nonreciprocated event (i.e., peer j uploads resources to peer i while peer i does not upload resources to peer j). This provides j with more confidence that the particular peer may not actively reciprocate its data. This also prevents the associated peers from taking advantage of a peer through optimistic initialization and possible free riding. In our implementation, when peer j estimates the reward for peer i, peer j can assume for the expected download rates $\hat{L}_{ij}$ that

i) $\hat{L}_{ij}$ satisfies $\hat{L}_{ij}(n-1)/\hat{L}_{ij}(n) < \hat{L}_{ij}(n)/\hat{L}_{ij}(n+1)$ where n denotes the number of nonreciprocated events, and
ii) $\hat{L}_{ij}(n)$ decreases exponentially such that it approaches 0 after several attempts.

Assumption i) means that the ratio of the estimated rate of two consecutive events is an increasing function of n. This also implies an increasing uncertainty about peer i's reciprocation behavior. Assumption ii) is required to prevent nonreciprocating behavior including free-riding (see Appendix D of the online supplemental material for an implemented function satisfying properties i) and ii)).

Note that white washing [27] is not possible in our design, because peers are identified by their IP addresses.

4.2.2 Finding State Transition Probability

The state transition probabilities are updated every rechoke period, and thus, each peer can capture the time-varying resource reciprocation behaviors of its associated peers. Every rechoke period at $t+1$, peer j stores 3-bit triplets for its associated peer i, $(s_{ij}(t), a_{ji}(t), s_{ij}(t+1))$. Peer j stores the triplets for its associated peers in its *reduced peer set*, which will be discussed later in this section, or peers that uploaded to peer j at time t or $t+1$. In our design, we compute the

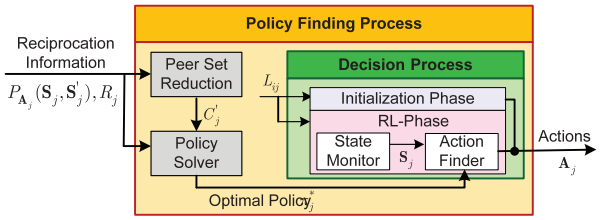

Fig. 3. A policy finding process.

state transition probability functions based on the empirical frequency, and assume that the state transitions of each peer are independent. Thus, the state transition probability $P_{\mathbf{A}_j(t)}(\mathbf{S}_j(t), \mathbf{S}_j(t+1))$ from $\mathbf{S}_j(t) = (s_{1j}(t), \ldots, s_{Nj}(t))$ to $\mathbf{S}_j(t+1) = (s_{1j}(t+1), \ldots, s_{Nj}(t+1))$ given an action $\mathbf{A}_j(t) = (a_{j1}(t), \ldots, a_{jN}(t))$ can be expressed as

$$P_{\mathbf{A}_j(t)}(\mathbf{S}_j(t), \mathbf{S}_j(t+1)) = \prod_{i=1}^{N} \Pr(s_{ij}(t+1) | s_{ij}(t), a_{ji}(t)).$$

4.3 The Policy Finding Process

The policy finding process runs in parallel with the learning process, while using the information obtained from the learning process. This process is depicted in Fig. 3.

The first part of the policy finding process is to find the peer selection policy. Finding the RL-based policy frequently may require high computational complexity, if the number of the associated peers becomes large. Specifically, the number of states and the number of actions for peer j are both 2^{N-1} for N associated peers, i.e., $|\mathbf{A}_j| = |\mathbf{S}_j| = 2^{N-1}$. Thus, the state transition probability storage magnitude is $O(4^N)$. Moreover, it is known that the computation complexity required for converging to an optimal policy increases in the order of $O(|\mathbf{A}_j||\mathbf{S}_j|^2 = 8^N)$ [28]. Hence, in order to practically implement the proposed algorithm, it is critical to reduce the number of peers that a peer considers for reciprocation (see Section 3). Therefore, this process begins with reducing the set of associated peers, and then, finds policy π_j that maximizes the cumulative discounted expected reward (i.e., in (1)) in the reduced peer set.

The goal of reducing the associated peer set is to select peers that can reciprocate their resources with higher probability and with higher upload rates in the reduced peer set. Specifically, peer j computes the expected rewards (or download rates) $\hat{L}_{ij}$ from each peer $i \in C_j$, defined as

$$\hat{L}_{ij}(t+1) = L_{ij}(t) \times \Pr(i \Rightarrow j),$$

where $i \Rightarrow j$ means that peer i serves peer j and it is computed by

$$\Pr(i \Rightarrow j) = \sum_{t \in T} a_{ji}(t) \Big/ \sum_{t \in T, a_{ji}(t)=1} s_{ij}(t+1).$$

Based on the computed $\hat{L}_{ij}$, peer j reduces its associated peer set by iteratively eliminating the peers with the smallest $\hat{L}_{ij}$ in its associated peer set. Based on our experiments, we have observed that if the reduced peer set exceeds seven peers, finding the RL-based policy becomes significantly slow, leading to severe performance degradation for the RL-enhanced client. Thus, in our implementation, we set the size of the reduced peer set to seven (i.e., $T = 7$ in the algorithm presented in Appendix E of the online supplemental material).

The computed policy holds for up to three additional rechoke periods, which is determined by considering the tradeoff between the time for enough reciprocation and the time for capturing the network dynamics. However, this duration can be adapted. For example, the updating frequency can be easy reduced to less than 3 rechoke periods if major changes in the network such as peers joining or leaving are experienced. Alternatively, the duration can be more than 3 rechoke periods if there are limited changes experienced.

4.4 The Decision Process

The decision process is a subprocess that is included in the policy finding process, and it determines the choking decisions. The decision process includes two phases, which are the initialization phase and the RL phase.

4.4.1 Initialization Phase

Since no information about associated peers is available for a newly joining peer j, peer j begins with adopting the regular BitTorrent mechanisms (i.e., the Tit-for-Tat mechanism and the optimistic unchoke mechanism) in the initialization phase. This enables the peer to collect information such as the rewards and the state transition probabilities with respect to its associated peers. During this phase, peer j discovers new peers, i.e., downloads from peers for the first time. Additional details of how to decide the duration of the initialization phase are provided in Appendix F of the online supplemental material.

In our implementation, peer j counts the number of peers without reciprocation history within every rechoke period. Once the count is reduced by one in a duration of three rechoke periods over two consecutive durations (i.e., six rechoke periods), peer j switches to the continuous phase and begins to adopt the RL-based strategy. Based on our experiments, peers switch from the initialization phase to the continuous phase after approximately 60 rechoke periods in the flash-crowd scenarios and after approximately 36 rechoke periods in the steady-state scenarios. However, different network settings may lead to different durations of the initialization phase.

4.4.2 RL Phase

Once a peer switches to the RL phase, it stays in this phase until the end of the downloading process. In this phase, peer j determines the decisions on peer selection based on the policy obtained from the policy finding process in every rechoke period. Peer j determines its current state $\mathbf{S}_j$ and the corresponding action $\mathbf{A}_j$ based on the policy π_j, i.e., $\mathbf{A}_j = \pi_j(\mathbf{S}_j)$. Note that $\mathbf{A}_j$ is a set of decisions on peer selection of peer j, i.e., either to choke or to unchoke. In our implementation, the peer selection decisions are made based on the RL policy every 10 seconds (as in regular BitTorrent). The selected peers will be unchoked for a rechoke period of ten seconds. The minimum number of unchoked peers is four. The number of unchoked peers can increase if either the upload capacity of a peer that makes the peer selection decisions is not saturated, or the upload

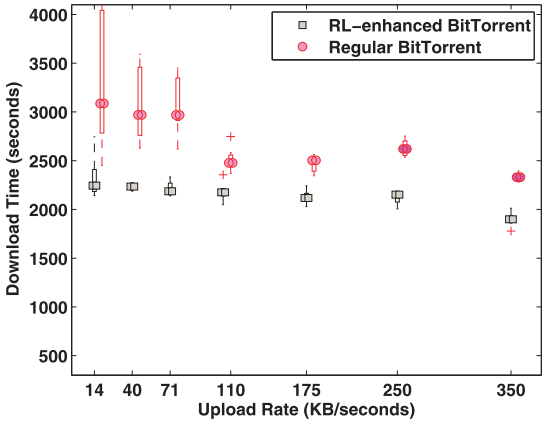

Fig. 4. Leecher's download time.

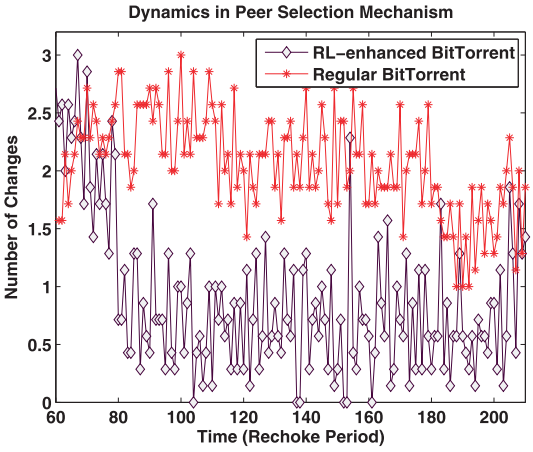

Fig. 5. Peer selection mechanism dynamics.

bandwidth of a peer that makes the peer selection decisions is higher than that of the most peers it interacts with.

5 Experimental Results

5.1 Methodology

All of our experiments are performed on the Planetlab experiment platform [29], which utilizes nodes (machines) located across the globe. Unless otherwise specified, the default implementations of leecher and seed in regular BitTorrent systems are deployed.

The upload capacities of the nodes are set according to the bandwidth distribution of typical BitTorrent leechers [6]. This distribution was estimated based on empirical measurements of BitTorrent swarms including more than 300,000 unique BitTorrent IPs. Since several nodes may be incapable of matching the target upload capacities determined by the bandwidth distribution, we scale the upload capacity and other relevant experiment parameters such as file size by 1/20th. However, there is no limitation on the download bandwidth.

All peers begin the download process simultaneously, which emulates a flash crowd scenario. The initial seeds stay and are connected throughout the entire experiment. In order to provide synthetic churn with constant capacity, once leechers finish downloading an entire file, they immediately disconnect their connections from their associated peers. Then, they rejoin the network as new comers while requesting the entire file. This enables our experiments to have the same upload bandwidth distribution for the duration of the experiment.

Unless otherwise specified, our experiments host 104 Planetlab nodes, 100 leechers, and 4 seeds with a combined capacity of 128 KB/s, serving a 99 MB file. While the discount factor is set to 0.8 for the RL-enhanced mode, this value can be adaptively chosen as illustrated in [23], [25]. In order to avoid a potentially negative impact which can be incurred by the unavailability of peers located in the Planetlab during the experiment, the availability and unavailability of peers are monitored by recording the information about their availability in log files maintained by each peer during the running time of all the experiments.

5.2 Single RL Leecher

In this experiment, only a single leecher adopts the RL-enhanced protocol, while the rest of the leechers in the network run with the regular BitTorrent. We assume that there is no free-rider in the network.[2] Fig. 4 shows the comparison of the download time of a single leecher that is adopting the RL-enhanced protocol and the other leechers that are adopting the regular BitTorrent protocol, as a function of their upload capacities over seven trials.

In Fig. 4, separate boxplots[3] are depicted for the different scenarios. The results in Fig. 4 provide several insights into the operation of our RL-based protocol. High and low capacity leechers benefit from the *RL-enhanced* with 12-27 percent improvement of their download time performance as indicated by the median. This improvement provides leechers with an incentive to deploy the proposed protocol. Moreover, the RL-based strategy does not only improve performance; it also provides more consistent performance across multiple trials. By unchoking peers based on the information about the reciprocal behaviors, the proposed protocol can prevent random behaviors, which may appear in the regular BitTorrent Tit-for-Tat and optimistic unchoke implementations, resulting in unstable peer selections and correspondingly slow convergence.

This shows that the proposed protocol can be selectively deployed by peers (i.e., not all peers in the network need to adopt the proposed protocol) and also that by adopting the proposed protocol the performance of these peers is improved.

We further study the stability of the peer selection mechanism. The stability of the peer selection mechanism is closely related to the performance of the system. This is because when a peer starts to upload to another peer, it takes time for the peer to reach its full capacity. Thus, it is preferable to minimize the peer selection fluctuations. The peer selection fluctuations are measured by comparing the peer selection decisions during two consecutive rechoke periods and the difference between the two decisions. Fig. 5 shows the average number of peer selection changes over several rechoke periods (time) for a single peer. It shows that the average number of peer selection changes in the

2. This is a common scenario tested by other protocols (e.g., [6], [30]).

3. The top and the bottom of the boxes represent the 75th and the 25th percentile samples, respectively, over all seven experiments. The markers inside the boxes represent the median, while the vertical lines extending above and below the boxes represent the maximum and minimum of samples within the ranges of 1.5 times the box height from the box boarder. Outliers are marked individually with "+" mark.

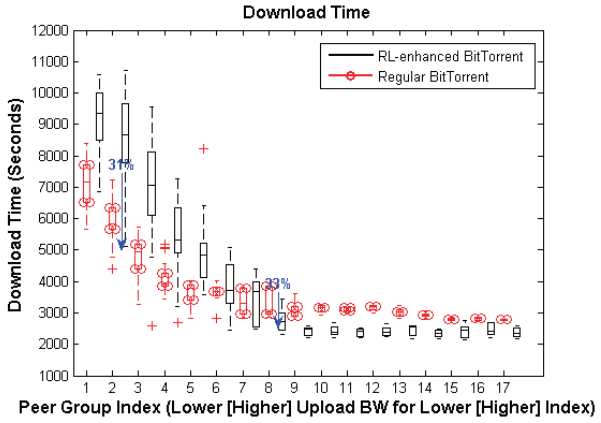

Fig. 6. Download completion time for leechers.

RL-enhanced mode is smaller than that in the regular BitTorrent mode for the majority of the time. More specifically, there are 2.1 changes on average in the regular BitTorrent mode, while there are 0.9 changes on average in the RL-enhanced mode. Thus, the RL-enhanced peer selection mechanism is more stable than the peer selection mechanism in the regular BitTorrent mode, reducing the fluctuations by 57 percent on average. Note that the improvement in stability of the peer selection mechanism mainly stems from replacing the Tit-for-Tat mechanism of the BitTorrent network with the RL-enhanced mechanism that performs foresighted unchoking decisions.

5.3 Performance of Leechers without Free-Riders

In this experiment, we compare the performance of the systems that consist of the leechers, where all of them adopt either the regular BitTorrent protocol or RL-enhanced mechanism. In order to highlight the performance of different systems, we assume that there are no free-riders in the networks and 50 leechers are hosted. Fig. 6 shows the download completion time of leechers. For each group of leechers having the same upload capacity, separate box-plots are depicted for the different scenarios.

The results show a clear performance difference among high-capacity leechers (i.e., fastest 20 percent leechers) and low-capacity leechers (slowest 80 percent leechers). High-capacity leechers can significantly improve their download completion time. For example, leechers having the upload capacity of at least 18 KB/sec (Peer Group Index 8) improve their download completion time by up to 33 percent in median. Unlike the leechers in the regular BitTorrent system, the RL-enhanced leechers determine their peer selection decisions based on the long-term history. This enables the leechers to estimate the behaviors of their associated peers more accurately. Moreover, due to the random peer selection in the regular BitTorrent, there is a high probability that high capacity leechers need to reciprocate with the low-capacity leechers [6]. However, because the random decisions are made only in the initialization phase or in order to collect the reciprocation history of newly joined peers, the number of random peer selection is significantly reduced in the RL-enhanced mode. As a result, the high capacity leechers increase their probability to reciprocate resources with other high capacity leechers. This is confirmed in Fig. 7, which shows the unchoking percentage among the 20 percent high capacity leechers.

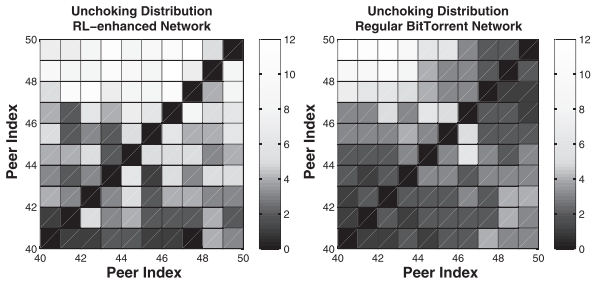

Fig. 7. Unchokes among the 20 percent fastest peers.

It is clearly observed that the collaboration among high capacity leechers improves when the leechers are in the RL-enhanced mode. Thus, we can conclude that the RL-based strategy further improves the incentive mechanisms compared to the regular BitTorrent system.

We further study the fairness improvement of the RL-based approach. Recent studies [6], [7], [15] show that the regular BitTorrent protocol suffers from unfairness particularly for high capacity leechers. Fig. 8 shows the upload rates and the average download rates of the leechers. It implies that the fairness is improved in the RL-enhanced network, since high-capacity leechers can increase their download rates as much as their upload rates despite the restriction of limited seeds' upload rates. On the other hand, in the RL-enhanced network, the download rates of low-capacity leechers decrease by at most 36 percent, compared to the regular BitTorrent system. However, all the peers that are slowed down by the RL-based strategy still download faster than their upload rates.

5.4 Performance of Leechers with Free-Riders

Unlike the above sections, we investigate how effectively the proposed protocol can prevent selfish behaviors such as free-riding in this section. Hence, our focus is on studying how the free-riders are punished because of their selfish behaviors. Fig. 9 shows the number of free-riders and time required for free-riders to complete downloading a 99 MB file in a network consisting of 50 contributing leechers. Fig. 9 confirms that in the RL-enhanced network, the leechers are able to effectively penalize the free-riders, as it takes longer time for the free-riders to complete their downloads (requires 8-20 percent more time in the median compared to the regular BitTorrent protocol). This is because the RL-enhanced leechers can efficiently capture the selfish

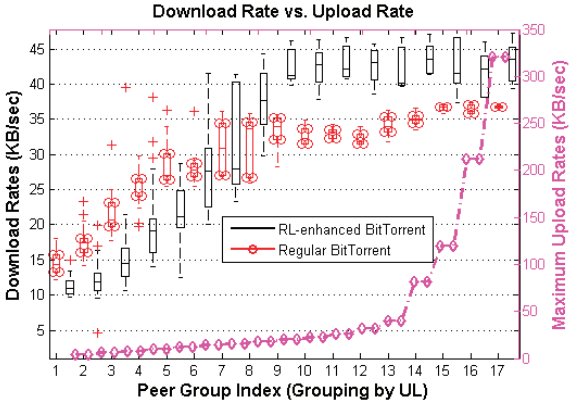

Fig. 8. Download rates for different upload rates.

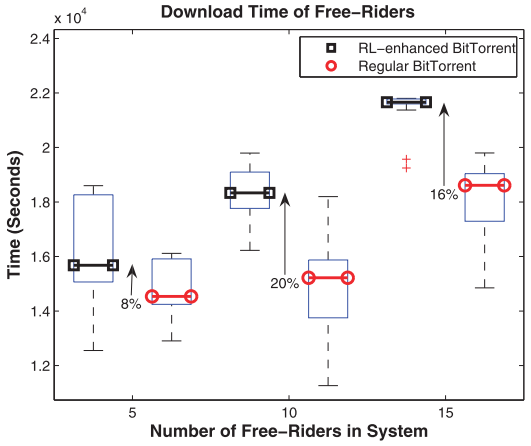

Fig. 9. Download completion time for free-riders.

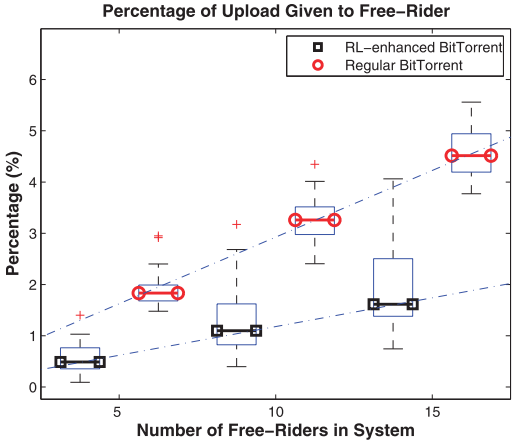

Fig. 10. Percentage of free-riders' downloads.

behaviors of the free-riders. Hence, the free-riders download their content mainly from seeds and not from the leechers. This is also confirmed from the results shown in Fig. 10, which shows that the leechers in the regular BitTorrent network upload approximately 2.8-3.7 times more data to the free-riders compared to the RL-enhanced network. Fig. 10 also shows that the RL-enhanced network is more robust against the selfish behaviors of peers than the network with the regular BitTorrent protocol. For example, in the network with 15 free-riders, the leechers in the regular BitTorrent systems upload 4.5 percent of their total upload capacity to free-riders, while they only upload 1.6 percent of their total upload capacity in the RL-enhanced network. Thus, the leechers can reduce the amount of upload capacity to free-riders by 64 percent

In summary, our experimental results confirm that the proposed RL-based protocol improves the peer's download rates, improves the stability of the peer selection mechanism, improves collaboration among high capacity peers, improves fairness in the system, and discourages noncooperative behaviors such as free-riding.

6 Conclusions and Future Work

In this paper, we propose a protocol that can replace the Tit-for-Tat and the optimistic unchoke peer selection mechanisms in the regular BitTorrent protocol. In the proposed protocol, the evolution of the peers' interactions across the various rechoke periods is modeled as repeated interactions in a game. The peers adopt RL to make their upload decisions while explicitly considering the reciprocation behaviors of their associated peers. Their decisions enable the peers to maximize the long-term future expected utility. Our experimental results based on actual implementation confirm that the proposed protocol provides several improvements in terms of the stability of the peer selection mechanism, collaboration among high capacity peers, system fairness, the robustness of the network against noncooperative behaviors, and downloading rates.

As for interesting research directions, the adopted RL approach can additionally consider application-specific information, which may improve the performance of the proposed protocol. For example, the proposed protocol can be modified such that it supports efficient delivery of delay-sensitive data (e.g., multimedia streaming services). In this case, additional information such as delay deadlines (playback deadline), video encoding structure, rate-distortion optimized scheduling, and reciprocation can be taken into account. The RL can also be deployed to seeders with modified reward functions as well as state transition probabilities, which may improve system performance. Moreover, if a global history in the system is available, the learning process can be boosted up. Finally, the study on the robustness of the RL-based approach against strategic manipulation of associated peers is an important topic of future research.

Acknowledgments

The material in this paper was presented in part at the Thirtieth IEEE International Conference on Computer Communications (IEEE INFOCOM 2011), Shanghai, China, April 2011. This work was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012-0002917), in part by the MKE (Ministry of Knowledge Economy) under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2012-H0301-12-1008, NIPA-2012-H0301-12-4004), in part by the Korea Meteorological Administration Research and Development Program under grant CATER 2012-3064, and in part by NSF grant CNS 0831549. The corresponding author is H. Park. This work was mainly performed while the R. Izhak-Ratzin and the H. Park were with UCLA.

References

[1] IPOQUE, "IPOQUE Internet Measurements 2008-2009," http://www.ipoque.com/, 2012.
[2] BitTorrent, "BitTorrent," http://www.bittorrent.com/, 2012.
[3] "Skype," http://skype.com, 2012.
[4] "PPLive," http://pplive.com, 2012.
[5] B. Cohen, "Incentives Build Robustness in BitTorrent," *Proc. Workshop Economics of Peer-to-Peer Systems (P2PEcon)*, 2003.
[6] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "Do Incentives Build Robustness in BitTorrent?" *Proc. Fourth USENIX Symp. Networked Systems Design and Implementation (NSDI)*, 2007.
[7] A. Bharambe, C. Herley, and V. Padmanabhan, "Analyzing and Improving a BitTorrent Network's Performance Mechanisms," *Proc. IEEE INFOCOM*, 2006.

[8] A. Legout, N. Liogkas, E. Kohler, and L. Zhang, "Clustering and Sharing Incentives in BitTorrent Systems," *SIGMETRICS Performance Evaluation Rev.*, vol. 35, no. 1, pp. 301-312, 2007.
[9] M. Piatek, T. Isdal, A. Krishnamurthy, and T. Anderson, "One Hop Reputations for P2P File Sharing Workloads," *Proc. Fifth USENIX Symp. Networked Systems Design and Implementation (NSDI)*, 2008.
[10] N. Liogkas, R. Nelson, E. Kohler, and L. Zhang, "Exploiting BitTorrent for Fun (But Not Profit)," *Proc. Int'l Workshop Peer-To-Peer Systems (IPTPS)*, 2006.
[11] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer, "Free Riding in BitTorrent is Cheap," *Proc. Fifth Workshop Hot Topics in Networks (HotNets-V)*, 2006.
[12] M. Sirivianos, J.H. Park, R. Chen, and X. Yang, "Free-Riding in BitTorrent Networks with the Large View Exploit," *Proc. Int'l Workshop Peer-To-Peer Systems (IPTPS)*, 2007.
[13] S. Buchegger and J.-Y. le Boudec, "A Robust Reputation System for P2P and Mobile Ad-Hoc Networks," *Proc. Second Workshop Economics of P2P Systems*, 2004.
[14] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for P2P Electronic Communities," *IEEE Trans. Knowledge and Data Eng.*, vol. 16, no. 7, pp. 843-857, July 2004.
[15] R. Izhak-Razin, "Collaboration in BitTorrent Systems," *Proc. Eight Int'l IFIP-TC 6 Networking Conf.*, 2009.
[16] R. Izhak-Razin, N. Liogkas, and R. Majumdar, "Team Incentives in BitTorrent Systems," *Proc. 18th Int'l Conf. Computer Comm. and Networks (ICCCN)*, 2009.
[17] J. Hu and P. Wellman, "Multiagent Reinforcement Learning: Theorectical Framework and an Algorithm," *Proc. Int'l Conf. Machine Learning*, 1998.
[18] R.S. Sutton and A.G. Barto, *Reinforcement Learning: An Introduction.* MIT Press, 1998.
[19] A. Vlavianos, M. Iliofotou, and M. Faloutsos, "BiToS: Enhancing BitTorrent for Supporting Streaming Applications," *Proc. IEEE INFOCOM*, 2006.
[20] Y.R. Choe, D.L. Schuff, J.M. Dyaberi, and V.S. Pai, "Improving VoD Server Efficiency with BitTorrent," *Proc. Int'l Conf. Multimedia*, 2007.
[21] H. Park, R.I. Ratzin, and M. van der Schaar, "Peer-to-Peer Networks - Protocols, Cooperation and Competition," *Streaming Media Architectures, Techniques, and Applications: Recent Advances*, C. Zhu, Y. Li, and X. Niu, eds. IGI Global, 2011.
[22] R. Izhak-Ratzin, H. Park, and M. van der Schaar, "Reinforcement Learning in BitTorrent Systems," *Proc. IEEE INFOCOM*, pp. 406-410, Apr. 2011.
[23] H. Park and M. van der Schaar, "A Framework for Foresighted Resource Reciprocation in P2P Networks," *IEEE Trans. Multimedia*, vol. 11, no. 1, pp. 101-116, Jan. 2009.
[24] E.J. Wegman, "Maximum Likelihood Estimation of a Probability Density Function," *The Indian J. Statistics*, vol. 37, pp. 211-224, 1975.
[25] H. Park and M. van der Schaar, "Evolution of Resource Reciprocation Strategies in P2P Networks," *IEEE Trans. Signal Processing*, vol. 58, no. 3, pp. 1205-1218, Mar. 2010.
[26] "Enhanced-CTorrent," http://www.rahul.net/dholmes/ctorrent, 2012.
[27] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica, "Free-Riding and Whitewashing in Peer-to-Peer Systems," *Proc. ACM SIGCOMM Workshop Practice and Theory of Incentives in Networked Systems (PINS)*, pp. 228-236, Aug. 2004.
[28] M.L. Littman, T.L. Dean, and L.P. Kaelbling, "On the Complexity of Solving Markov Decision Problems," *Proc. Conf. Uncertainty in Artificial Intelligence*, May 1995.
[29] A. Bavier, M. Bowman, B. Chun, D. Culler, S. Karlin, S. Muir, L. Peterson, T. Roscoe, T. Spalink, and M. Wawrzoniak, "Operating System Support for Planetary-Scale Network Services," *Proc. First Conf. Symp. Networked Systems Design and Implementation (NSDI)*, 2004.
[30] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee, "BitTorrent is an Auction: Analyzing and Improving BitTorrent's Incentives," *Proc. SIGCOMM*, 2008.

Rafit Izhak Ratzin received the BS (magna cum laude) degree in communication engineering from Ben Gurion University, Israel, the MS degree in electrical engineering from Tel Aviv University, Israel, and the MS and PhD degrees in computer science from the University of California, Los Angeles (UCLA), in 2000, 2003, 2005, and 2010, respectively. Currently, she is working for Palo-Alto Networks, CA. Her research interests include distributed systems and networks with emphasis on applying game theoretical methods as solution to practical networking and distributed systems problems.

Hyunggon Park received the BS degree in electronics and electrical engineering from the Pohang University of Science and Technology (POSTECH), Pohang, Korea, and the MS and PhD degrees in electrical engineering from the University of California, Los Angeles (UCLA), in 2004, 2006 and 2008, respectively. Currently, he is an assistant professor at the Department of Electronics Engineering, Ewha Womans University, Seoul, Korea. In 2008, he was an intern at IBM T.J.Watson Research Center, Hawthorne, NY, and was a senior researcher at the Signal Processing Laboratory (LTS4), Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 2009-2010. His research interests include game theoretic approaches for distributed resource management (resource reciprocation and resource allocation) strategies for multiuser systems and multiuser transmission over wireless/wired/peer-to-peer (P2P) networks. He was a recipient of the Graduate Study Abroad Scholarship from the Korea Science and Engineering Foundation during 2004-2006 and a recipient of the Electrical Engineering Department Fellowship at UCLA in 2008. He is a member of the IEEE.

Mihaela van der Schaar is chancellor's professor of electrical engineering at University of California, Los Angeles (UCLA). She is a distinguished lecturer of the Communications Society for 2011-2012, the editor-in-chief of *IEEE Transactions on Multimedia* and a member of the Editorial Board of the IEEE *Journal on Selected Topics in Signal Processing*. Her research interests include dynamic multiuser networks and system designs, online learning, network economics, and game theory, multimedia networking, communication, processing, and systems, and multimedia stream mining. She received an NSF CAREER Award (2004), the Best Paper Award from *IEEE Transactions on Circuits and Systems for Video Technology*(2005), the Okawa Foundation Award (2006), the IBM Faculty Award (2005, 2007, 2008), the Most Cited Paper Award from EURASIP: *Image Communications Journal* (2006), the Gamenets Conference Best Paper Award (2011) and the 2011 IEEE Circuits and Systems Society Darlington Award Best Paper Award. She received three ISO Awards for her contributions to the MPEG video compression and streaming international standardization activities, and holds 33 granted US patents. She is a fellow of the IEEE.