Title	Closest playback-point first: A new peer selection algorithm for P2P VoD systems
Author(s)	Wen, Z; Liu, N; Yeung, KL; Lei, Z
Citation	Globecom - IEEE Global Telecommunications Conference, 2011
Issued Date	2011
URL	http://hdl.handle.net/10722/140255
Rights	©2011 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

Closest Playback-Point First: A New Peer Selection Algorithm for P2P VoD Systems

Zheng Wen, Nianwang Liu, Kwan L. Yeung
Department of Electrical and Electronic Engineering
The University of Hong Kong
Pokfulam, Hong Kong
{wenzheng, nwliu, kyeung}@eee.hku.hk

Zhibin Lei
Applied Science & Technology Research Institute (ASTRI)
Shatin, Hong Kong
lei@astri.org

***Abstract*—Peer-to-peer (P2P) based video-on-demand (VoD) streaming service has been gaining popularity recently. Unlike live streaming, a VoD peer always starts its playback from the beginning of a stored video. The playback-points of different peers, as well as the amount of video contents/pieces they cached, depend on when they join the video session, or their viewing ages. As a result, the upload bandwidth of younger peers tends to be underutilized because older peers are not interested in their cached video pieces. The collaborative piece exchange among peers is undermined due to the unbalanced supply and demand. To address this issue, a playback-point based request peer selection algorithm is proposed in this paper. Specifically, when a peer requests a particular video piece, among the set of potential providers, a request is sent to the peer that has the smallest playback-point difference with itself. We call this request peer selection algorithm closest playback-point first (CPF). With CPF, peers with similar available content can be loosely grouped together for a more balanced collaborative piece exchange. Extensive packet-level simulations show that with CPF, the video playback quality is enhanced and the VoD server load is significantly reduced.**

I. INTRODUCTION

Inspired by the immense success and efficiency of BitTorrent [1] in distributing file contents to large number of users, increasing research as well as engineering efforts have been focusing on adapting the peer-to-peer system to provide large scale video streaming service, including both live streaming and video-on-demand, through the public Internet.

Although, live streaming and VoD streaming inherit common features from P2P file sharing system (such as BitTorrent): self-organized with highly transient population of users; continued scalability without the need of a central control server; and relatively higher control overhead incurred by the data-driven overlay construction, there are also fundamental differences among live streaming, VoD streaming and file sharing.

Firstly, streaming service explicitly imposes a real-time playback constraint of each video data piece, so in-order or deadline-aware piece request scheduling [2][3] is more preferable for live and VoD streaming systems to preserve playback continuity and to decrease startup delay. However, as no explicit time constraint is imposed by a file sharing system, the local rarest first piece scheduling scheme [4] is adopted to diversify the file content availability which is essential to provide incentives for peers to exchange data pieces.

Secondly, peers in live streaming system are synchronous in the sense that their video playback time is bounded by the playback lag with respect to the live streaming server which enhances the content availability in the network and also encourages data piece exchange among peers. As a result, a live streaming peer only needs to maintain a relatively small buffer window. However, in VoD streaming, peers joining the session at different times always watch a video from the beginning, the playback is asynchronous [5] [6]. This implies that peers may often not have pieces that are of interest to others. To be more exact, only older peers (peers joining the network earlier and thus with more video contents) are able to contribute to younger peers, but not vice versa. This asymmetric data flow can handicap peer's ability to help each other, and generates a heavy pressure/load on the VoD server. To alleviate the situation, the VoD streaming system requires each peer to have a much larger buffer window (usually 1GB) [5] than live streaming. In a (video) file sharing system, the playback can not happen until the whole file is received. In order to maximize the content diversity, an object-file-sized buffer window is kept by each peer. Selected file pieces are supplied by or contributed to neighbors until the complete file is received.

Thirdly, different overlay network construction strategies are used. In live streaming, due to the more critical timeliness requirements of disseminating the pieces, peers would normally entertain all piece requests from all its neighbors. In this case, a good overlay network is typically constructed such that each peer has more or less the same number of randomly selected neighbors [7]. In a file sharing system such as BitTorrent, tit-for-tat (T4T) incentive scheme is widely adopted for pairing peers with similar capability together. The direct reciprocity based on T4T not only encourages the collaborative piece exchange but also prevents free-riders. Due to the asynchronous playback and varying content availability in VoD streaming, it is less feasible for younger peers to reciprocate the older ones (because their cached video pieces are a subset of the older ones).. Accordingly, give-to-get [8][9], a modified T4T scheme based on indirect reciprocity, is proposed to jointly consider a neighbor's supply to both the selecting peer and others.

In this paper, we focus on enhancing the performance of a VoD streaming system through request peer selection: when a peer decides to retrieve/pull a specific missing video piece, the peer needs to select a neighbor to make a request. Note that

978-1-4244-9268-8/11/$26.00 ©2011 IEEE

each VoD peer has a prefetching window starting from its current playback-point. A peer is only interested in getting the video pieces inside its prefetching window. If the prefetching windows of two peers do not overlap, only the older peer can provide data to the younger one. As a result, the upload bandwidth of the younger peer tends to be underutilized [10], and the bidirectional collaborative data exchange tends to be undermined. Recent measurements have shown that up to 70% of video pieces may still need to be supplied by the VoD server [11]. To address this problem, we propose to select the peer that has the closest playback-point with the request sender to make piece requests. We call this request peer selection algorithm closest playback-point first (CPF). The idea of CPF is that the peer with the closest playback-point tends to have the largest prefetching window overlap with the request sender, which renders a higher probability for the two peers to mutually benefit from each other's cached video pieces. Besides, young peers have limited number of cached pieces for uploading to others. With CPF, their otherwise wasted upload bandwidth can be quickly utilized for supplying pieces to the younger peers. When the uplink bandwidth of all peers is better utilized, their reliance on VoD server is reduced therefore the VoD server load can also be significantly reduced.

The rest of the paper is organized as follows. In Section II, we present a brief summary of related work on adapting BitTorrent-like system for VoD streaming service. In Section III, our playback-point based request peer selection algorithm CPF is introduced. In Section IV, we present our packet-level simulation results to illustrate the superiority of our algorithm. Finally, we conclude the paper in Section V.

II. RELATED WORK

Research work on adapting the general P2P framework of BitTorrent to provide large scale video streaming service mainly focused on two issues: addressing the piece selection issue to meet the timeliness playback requirement; and refining the overlay network construction strategy that are robust to the varying content availability. As for the first category, existing work (e.g., [2][3], to name a few) all focused on leverage the tradeoff between random or rarest first algorithms that maximize the system content availability and in-order or deadline-aware schemes that cater to fluent in-order playback. Despite the unique character of each proposal, they all share a common feature that is to make a compromise and strategically combine the two different piece scheduling algorithms together. Due to the asynchronous playback feature and the unbalanced piece supply pattern, T4T based incentive scheme used in BitTorrent is not applicable for VoD system. New Approaches that address this issue are proposed in [8] [9] [10]. The general idea is to adopt indirect reciprocity which takes a peer's supply to the network rather than to a specific peer into consideration.

Besides the above mentioned issues, one open yet fundamental question is, how to select a neighbor peer to make piece request for a specific piece among potential providers, which we refer to as *request peer selection*. Since in VoD streaming system bidirectional data pieces supply is largely undermined due to asynchronous playback and the varying content availability, a simple random request peer selection is likely to stress the older peers and waste network bandwidth, which is briefly discussed in [12]. A tracker assisted solution is proposed in [13] where the tracker sorts peers in the network according to their arrival times and reply a peer list request with a group of peers that have closest arrival times to the request sender. A more complicated enhancement is to let the tracker know each peer's buffering progress (through sending periodical update message) and help the peer to find those with similar buffering progress. In such way, peers with similar age or buffering progress are grouped together to alleviate the impact of uni-directional data piece supply pattern. But, this is at the cost of a much higher control and processing overhead at the peer and tracker side, respectively. A more detailed simulation based analysis together with a distributed solution, LLP-P, is proposed in [6]. The general idea of LLP-P is to ask peers to report its request queue size to its neighbors periodically and send the piece request to the least loaded peer (peers with shortest queue size) that have the needed piece. Since, our CPF is also distributed in nature; we shall compare our algorithm with LLP-P by simulations in Section IV.

III. CLOSEST PLAYBACK-POINT FIRST

As mentioned in previous sections, the most distinctive feature of P2P VoD streaming system is its asynchronous playback. It implies that even though peers buffer all the watched pieces, it is still infeasible for younger peers to reciprocate the older ones, as younger peers lack pieces that are of interest to the elders. An example is shown in Fig.1, where P2 can only feed P3 but not P1, as P1 has already buffered all the content P2 has thus has no interest in retrieving any content from P2. The varying content availability and the unbalanced piece flow result in a waste of large portion of network upload capacity. To compensate, the server is unavoidably stressed.

But, one insight is that bi-directional data piece exchange among peers of similar content availability is not handicapped as their local buffer windows overlap to a large extent with each other. In other words, those peers have common interests in trading their cached pieces which could maximize the peers upload bandwidth utilization. As in VoD streaming system a peer's local buffer window is implicitly bounded by its playback-point, we propose to use peer's playback-point as a guideline to interpret the correlation of content availability among peers and to regulate the piece flow so as to maximize the upload bandwidth utilization from the system's point of view.

Specifically, each peer caches the watched pieces ranging from the beginning to the playback-point in buffer window. And for the sake of continuous playback, it also retrieves additional pieces in the prefetching window, which normally covers the pieces from the one next to the current playback-point to those to play in a short future. In other words, a closer playback-point implies the likelihood of a larger prefetching window overlap and thus a higher probability of bidirectional piece exchange. The system can exploit peer uplink capacity if it encourages peers to collaboratively exchange pieces with those of close playback-point. For example, in Fig.1, P3 and P4 can better utilize their uplink capacities by exchanging pieces that are of interest to each other instead of retrieving them from P1 or P2. In such a way, piece flows from the older to the younger ones (i.e., pieces flow from P2 to P3), deliver rare

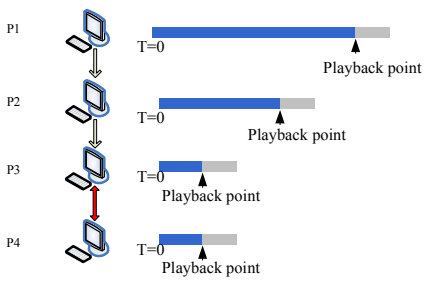

Figure 1. Example of playback point based clustering

pieces, and sometimes donate additional bandwidth when the aggregate upload bandwidth is less than that of the supply among younger peers. With such a regulated piece flow pattern, the server bandwidth consumption is also decreased. It may only be used to supply fresh pieces to peers with largest playback-point (older peers) and to push the initial pieces to peers just join the network as long as the aggregate bandwidth supply meets or exceeds network bandwidth demand.

To implement, we integrate this playback-point based scheduling strategy into the request peer selection algorithm and call it closest playback-point first (CPF). Specifically, peers in the network periodically exchange their playback-point with neighbors by e.g. piggybacking onto the buffermap update message. Once a video piece is selected for retrieving (e.g. based on the scheme depicted in Fig. 2), the request sender identifies all its neighbors with the selected piece as potential provides. Among the potential providers, a piece request is sent to the peer that has the smallest difference in playback point with the request sender. In the case that more than one potential providers are with the same playback offset, the request sender would randomly pick up one to send the piece request. And to avoid overwhelming a specific peer, a limit of the maximum number of piece requests sending to the same peer should be imposed (which is set to 10 in the simulation).

A salient point of our design is that we fix the playback-point of the VoD server at the beginning of the video. In other words, the server's playback-point will always remain at the first piece of the video (whereas the playback-points of other peers will move with the playback time). The rationale is to ensure that the server is always available for providing initial pieces to a new peer (assuming the server is in the peer list returned by the tracker) because its playback-point is closest to/same as the new peer. Getting initial pieces directly from server is highly desirable because the server has a higher uplink bandwidth that can "jump-start" a new peer. The playback-point of the new peer grows with its age in the system, but not the server. The new peer will then gracefully "move away" from the server due to the growing gap between their playback-points. In doing so, the new peer gradually shifts its piece requests to other regular peers (with closest playback-points) in the system. The loading at the server is then reduced – another desirable feature which frees up the server to help those in greater needs, e.g. uploading a data piece that cannot be found from existing peers in the system (due to peer churn). Indeed, the "speed" of shifting the load from the server to other regular peers can be controlled with a weighting factor on the gap between the playback-points of the server and the new peer. We leave that for future research.

IV. PERFORMANCE EVALUATION

A. Performance Metrics

The continuity index is adopted by [6] [7] [9] as a measure for video quality, but it is noted that this index does not capture the burstiness of the missing packets [6] and its associated impact on the frozen time. The frozen time is the time that a peer suspends its playback to wait for the missing video piece to arrive. In our simulations, we use two metrics to measure the playback continuity performance: the *total number of stops* due to lack of piece to playback; and the *total caching time* which sums up all the frozen time as discussed in [14]. Note that these two metrics are not necessarily correlated to each other, since a peer may stop playback only once but with a long frozen time. Besides, as some VoD implementations allow peers to cache or keep the full-sized video file in hard-disk while streaming [5], we also monitor the time duration for a peer to finish downloading the whole video file, which we refer to as *download time*. We also measure the *server load* at every 5 seconds, which shows the upload bandwidth consumption to serve the piece requests from all the peers. The *startup delay* that measures the initial time duration for a new peer to retrieve sufficient pieces before playback is also collected. Since our CPF is designed for request peer selection in this paper, it is expected that its impact on the startup delay will be minimal.

To illustrate the strength of our CPF, we compare it with two other algorithms, namely, the *random* request peer selection [12] where request sender randomly select a peer among the potential piece providers; and the *least loaded* request peer selection (referred to as LLP-P in [6]), in which peers periodically update neighbors its request queue size information and make the piece request to the one with shortest queue size. For a fair comparison, the queue size information used in LLP-P and the playback-point information used in our algorithm are all piggybacked to the buffermap exchange packet which is frequently sent out whenever a complete piece is received or the age of the buffermap is greater than 3 seconds.

B. Simulation Setup

We evaluate the proposed CPF though simulations using an event-driven P2P simulator provided by ASTRI[15] . The simulator is built on top of NS2 and carries out the detailed packet-level simulations of a P2P VoD streaming system. It is well tested and is extensively used for both research and pre-deployment simulations [16].

To focus only on the request peer selection strategy, unless otherwise stated, the following generalized system framework is used throughout the simulation: (1) One server that holds the complete video file stays in the network for the duration of the simulation. The video file is segmented into multiple pieces each contains the data for one second playback;(2) Each peer in the network keeps a list of neighbors with which it periodically exchanges their buffer maps; (3) A simple *section based* piece

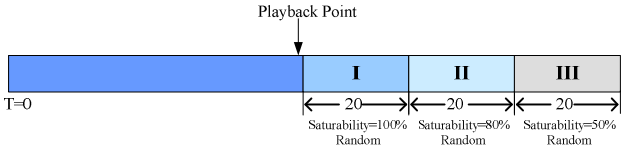

Figure 2. Piece selection strategy used in simulation

TABLE I. SIMULATION PARAMETERS

Simulation Time	1000 sec.
Avg. inter-arrival Time	4 sec.
Number of Peers	100
Number of Pieces	300
Piece Size	25KB(200Kbps)
Peer Node bandwidth (Down/Up)	3/0.5 Mbps
Server Bandwidth (Down/Up)	10/10 Mps
Peer Node Max Upload Connections (P)	20
Server Max Upload Connections	40
Max # Simultaneous Requests to a Specific Peer	10
Max. # simultaneous Requests (N)	60

selection strategy is used where the prefetching window, with a size of 60 pieces/seconds, is divided into three equal sections each with different saturability, as shown in Fig. 2. A peer randomly selects a piece from a lower numbered section to request until the saturability of that section is satisfied before retrieving a piece from the next section. A peer never requests a piece beyond the request window; (4) At any time, each peer can have up to N pending/on-going piece requests and among them, no more than D requests to the same peer; (5) At any time, a peer can serve up to P peers whose piece requests are served on their order of arrivals, and the server is able to entertain the piece requests from all peers(6) Each peer starts video playback after receiving the first five pieces and resumes the playback as long as the piece for the next second is available, otherwise the playback will be suspended until the next five pieces are retrieved; (7) Peers join the VoD session with an exponentially distributed inter-arrival time of 4 seconds until there are 100 peers in the system. They would not leave the network throughout the simulation. Other parameters used in the simulation are summarized in Table I.

C. Simulation Results

Fig. 3 depicts the server load comparison. We observe an overlap of the three curves during the initial phase which could be interpreted as server bandwidth consumption of using different algorithms is about the same when there are few peers join the network. Since very limited number of pieces is available in the neighborhood, all peers tend to retrieve pieces from the server. Due to the donation of peers' uploading bandwidth, the server load gradually decreases as the network grows up and peers begin playback during which the performance difference emerges. Noticeably, our CPF outperforms the other two algorithms. It saves about 60% and 18% of server upload bandwidth compared to the random request peer selection algorithm and the LLP-P, respectively. This significant server load reduction largely results from the highly peer upload bandwidth utilization in our CPF. Since

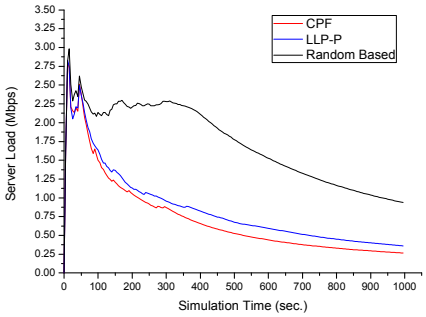

Figure 3. Server load comparison

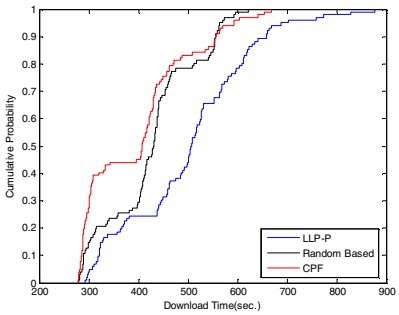

Figure 4. Download time comparison,

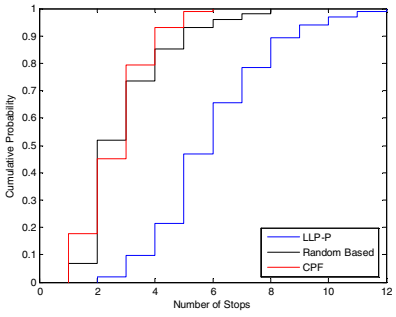

Figure 5. Number of stops comparison

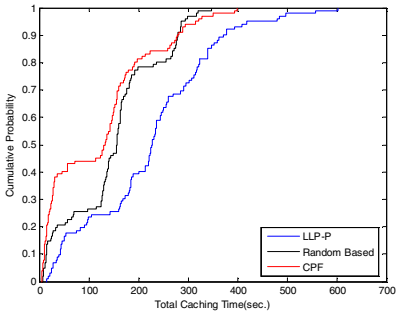

Figure 6. Total caching time comparison

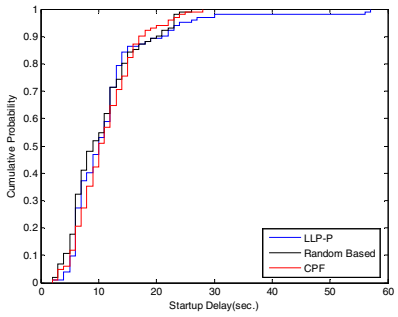

Figure 7. Startup delay comparison

TABLE II. AVERAGE VALUE COMPARISON

	CPF	LLP-P	Random Based
Download Time (sec.)	396.35873	500.9334	425.0706
Startup Delay (sec.)	11.10784	11.56863	10.40196
Total Caching Time (sec.)	120.98039	222.6863	148.6961
Number of Stops	2.65686	5.96078	2.95098

peers with similar content availability are inclined to exploit their upload capacity through piece exchange rather than stress the server. LLP-P is also able to reduce the server load as compared to the random based request peer selection algorithm due to the better load balance scheme it adopts. However, it is less efficient than our CPF. Because LLP-P does not distinguish the server from normal peers, peers in LLP-P are likely to rely more on the server. Fig. 4-7 shows the quality of service comparison of the three algorithms. The cumulative distribution function (CDF) of different metrics is used to demonstrate performance difference and the corresponding average values are summarized in Table II. As compared to the other two schemes, our CPF shows a better performance in the sense that large portion of peers in the network experience a shorter download time, less number of stops, competitive startup delay and shrinked total caching time. These quality-of-service improvements can also be attributed to the well regulated piece supply pattern with CPF request scheduling. It is also noted that LLP-P shows poor performance as compared to the random based request peer selection algorithm, which is inconsistent the results presented in [6]. We doubt this could due to the different playback policies adopted. (In [6], peers begin playback after waiting a fixed amount of time and never stop even if the next piece to play is missing. Instead, it counts the piece missing ratio as the performance metrics).

In a word, the above simulation results have shown the superiority of CPF in significantly reducing server load as well as providing better playback performance in all performance metrics over random and LLP-P algorithms.

V. CONCLUTION

In this paper, we proposed a playback-point based request peer selection algorithm, CPF, for P2P VoD streaming system. In CPF, piece requests are sent to the potential provider that has the smallest playback point difference with respect to the piece requester. In such a way, peers with similar content availability can better utilize their upload capacity and also offload the video server from the system's point of view. Our packet-level simulation results illustrate the superiority of the proposed algorithm in reducing the server bandwidth consumption. At the meantime, it preserves better quality of service in terms of number of stops, total caching time and file downloading time.

Note that the concept of closest playback-point first can be generalized to overlay network construction. For example, when a new peer joins, the tracker can return a list of peers with the closest playback points. For an on-going peer, it can continuously refine its neighbor list by adding peers with closer playback points, which can be learned by neighbor lists exchange.

REFERENCES

[1] BitTorrent: http://www.bittorrent.com/.

[2] Y. Zhou, D.-M. Chiu, and J. Lui, "A simple model for analyzing p2p streaming protocols," in *ICNP*, 2007.

[3] K. W. Hwang, V. Misra, and D. Rubenstein, "Stored media streaming in bittorrent-like p2p networks," *Tech Report, Columbia University, NY*,no. cucs-024-08, 2008.

[4] R. S. Dongyu Qiu, "Modeling and performance analysis of bittorrentlike peer-to-peer networks," in *ACM SIGCOMM*, 2004..

[5] Y. Huang, T. Z. Fu, D. M. Chiu, J. C. Lui, and C. Huang, "Challenges, design and analysis of a large-scale p2p-vod system," *ACM SIGCOMM*, 2008.

[6] Y.Yang, A.L.H.Chow, L. Golubchik, and D. Bragg, "Improving QoS in bittorrent-like VoD systems," In *INFOCOM*, 2010

[7] X. Zhang, J. Liu, B. Li, and T. Yum, "Coolstreaming/donet: A datadriven overlay network for efficient live media streaming," in *INFOCOM*, 2005.

[8] J J D Mol, J A Pouwelse, M Meulpolder, D H J Epema, H J Sips, "Give-to-get: Free-riding-resilient video-on-demand in p2p systems," in SPIE 2008.

[9] L.D'Acunto, N. Andrade, J. Pouwelse and H. Sips, "Peer selection strategies for improved QoS in heterogeneous bittorrent-like vod systems," in IEEE International Symposium on Multimedia, 2010.

[10] C. Huang, J. Li, and K.W.Ross, "Can internet video-on-demand be profitable?" in SIGCOMM, 2007.

[11] B. Cheng, X. Liu, Z. Zhang and H. Jin, "A measurement study of a peer-to-peer video-on-demand system," in *IPTPS*, 2007

[12] N. Parvez, C. Williamson, A. Mahanti, and N. Carlsson, "Analysis of bittorrent-like protocols for on-demand stored media streaming," in *SIGMETRICS*, 2008.

[13] C. Liang, Z. Fu, Y. Liu, and C. W. Wu, "Ipass: Incentivized peer-assisted system for asynchronous streaming," in *INFOCOM Mini Conf.*, 2009.

[14] Z.J. Fu, D.M Chiu and Z.B Lei, "Designing QoE experiments to evaluate Peer-to-Peer streaming applications", Visual Communications and Image Processing 2010,

[15] ASTRI: http:// www. astri. org/.

[16] J. Huang, G.Cheng, J.Liu, and D.M. Chiu et. all, "A simulation tool for the design and provisioning of p2p asisted content distribution platforms" under review. A copy is available at http://personal.ie.cuhk.edu.hk/~dmchiu/references/P2P_Simulation.pdf