Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Deadline-aware network coding for video on demand service over P2P networks*

CHI Hui-cheng, ZHANG Qian
(*Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, China*)
E-mail: addison@cs.ust.hk; qianzh@cs.ust.hk

Received Dec. 8, 2005; revision accepted Feb. 20, 2006

Abstract: We are interested in providing Video-on-Demand (VoD) streaming service to a large population of clients using peer-to-peer (P2P) approach. Given the asynchronous demands from multiple clients, continuously changing of the buffered contents, and the continuous video display requirement, how to collaborate with potential partners to get expected data for future content delivery are very important and challenging. In this paper, we develop a novel scheduling algorithm based on deadline-aware network coding (DNC) to fully exploit the network resource for efficient VoD service. DNC generalizes the existing network coding (NC) paradigm, an elegant solution for ubiquitous data distribution. Yet, with deadline awareness, DNC improves the network throughput and meanwhile avoid missing the play deadline in high probability, which is a major deficiency of the conventional NC. Extensive simulation results demonstrated that DNC achieves high streaming continuity even in tight network conditions.

Key words: Video on Demand (VoD), Peer-to-Peer (P2P), Network coding (NC), Deadline-aware network coding (DNC)
doi:10.1631/jzus.2006.A0755 **Document code:** A **CLC number:** TN919.8

INTRODUCTION

With the widespread deployment of broadband access, Video-on-Demand (VoD) streaming on the Internet has received increasing attention recently. In VoD service, video streams are delivered to asynchronous users with low delay and VCR-like operation support (e.g., pause, fast-forward, and rewind). However, streaming to a large population of clients is very challenging due to the limited server capacity and little deployment of IP multicast in today's Internet (Quinn and Almeroth, 2001). In recent years peer-to-peer (P2P) has become a very popular tool for streaming service in the Internet (Chu *et al*., 2000; Cui *et al*., 2004; Do *et al*., 2004; Guo *et al*., 2003; Hefeeda *et al*., 2004; Wang and Liu, 2005; Zhou and Liu, 2005).

Fig.1 depicts the general framework of a typical P2P-based VoD system. In such a system, cooperative peers (peer, client, user and node are used interchangeably in this paper) are organized into an overlay network via unicast tunnels. There are two overlay networks in the VoD system, i.e., the index overlay for locating the users with expected data and the data overlay for cooperative content delivery. The streaming content is split into a sequence of segments, each of which is a smallest playable unit, and the server distributes these segments among clients with asynchronous demands. Each client caches a limited number of segments around its "play offset", which is the sequence number of the playing segment. The client exchanges the available segments with its "partners", which have close play offset and thus can help the client to get the expected data with high probability. For the users in the VoD system, when joining or implement VCR operations, it needs to search for the partners through the index overlay. After locating the partners, the user will collaborate with them to fetch and deliver the content. Therefore, partner search and "multi-partner scheduling" are the

---

* Project (No. DAG05/06.EG05) supported by the Research Grant Council (RGC) of Hong Kong, China

main components in the P2P-based VoD system. How to design the index overlay for efficient partner search had been well studied (Chu *et al*., 2000; Cui *et al*., 2004; Do *et al*., 2004; Guo *et al*., 2003; Hefeeda *et al*., 2004; Wang and Liu, 2005; Zhou and Liu, 2005), but to the best of our knowledge, there are few works on the cooperative multi-partner scheduling in VoD context. In this paper, we focus on the cooperative content delivery among multiple partners in the VoD system.

Multi-partner scheduling is a challenging task. Traditional cooperative schemes, such as smallest-delay (Do *et al*., 2004) and pull-based gossip algorithm (Zhang *et al*., 2005; Zhou and Liu, 2005), assign requests to partners based on the local neighborhood information, e.g., content in the local buffer and the available bandwidth. These schemes suffer from inefficient use of network resource in large and heterogeneous populations. There are some studies trying to leverage network coding to improve the throughput utilization and facilitate the design of optimal solution for static file download applications (Gkantsidis and Rodriguez, 2005; Li *et al*., 2005). With network coding, each peer (including the current node, its partners, and sometimes the server) performs linear combination on available segments and relays the combined segments to the partners. When a node receives enough linearly independent combinations, the original media can be reconstructed. However, it is not trivial to apply network coding in VoD system. First, since media play starts before all the segments are downloaded, some segments may miss the play deadline before being decoded, which causes severe performance degradation. Second, each peer maintains different sliding windows of the segments due to the limited buffer capacity, so it is hard to collaborate among multiple partners considering the different range of encoded segments.

In this work, we develop a novel scheduling algorithm based on deadline-aware network coding (DNC) to address the above challenge. Remembering that media segments have play deadline, instead of encoding all available segments, the DNC scheme adjusts the coding window for each node, which is the number of segments to be encoded, based on its bandwidth and play deadline. DNC improves the network throughput and meanwhile at high probability avoids missing the play deadline.

The rest of the paper is organized as follows. The background and motivation are reviewed in Section 2. The DNC scheduling algorithm is discussed in Section 3. Simulation results are given in Section 4, followed by the conclusions in Section 5.

BACKGROUND AND MOTIVATION

In this section, we briefly review the related works on the VoD service and present a concrete example that motivates our study.

There are many IP multicast proposals for VoD streaming since the birth of Internet, e.g., periodic broadcasting (Guo *et al*., 2002; Hu, 2001), batching (Aggarwal *et al*., 1996; Dan *et al*., 1996), patching (Gao *et al*., 1998; Hua *et al*., 1998), and bandwidth skimming (Eager *et al*., 2000). These proposals exploit the temporal locality of user requests to batch them into different sessions, and repeat the media in a number of multicast channels. However, due to the limited deployment of IP multicast, these protocols are not applicable in the current Internet.

Recently, researchers resort to application-layer solutions, e.g., application-layer-multicast (ALM) and P2P approach. The broadband speed and massive

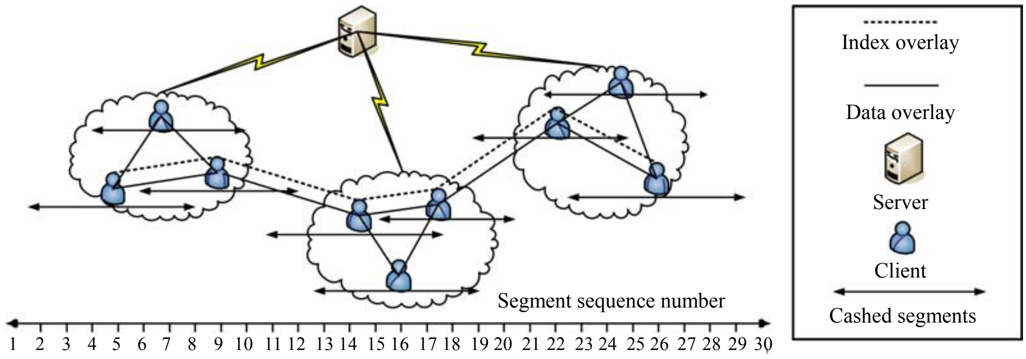

Fig.1 Typical P2P-based VoD system

buffer capacity at client side make these solutions readily deployable. Their applications in live media streaming have been well studied. Originating from IP multicast, previous ALM protocols often organize the overlay nodes into a tree structure. Due to the single-parent nature, tree-based overlay is unbalanced and vulnerable. Thus many P2P proposals exploit multiple parents to balance the load and enhance the robustness, e.g., mesh-based tree [Bullet (Kostic *et al.*, 2003)], multiple-tree overlay [Split-Stream (Castro *et al.*, 2003)], gossip partners [CoolStreaming (Zhang *et al.*, 2005)], and schemes leveraging advanced source coding techniques such as layered coding [PALS (Rejaie and Ortega, 2003)] and multiple description coding [CoopNet (Padmanabhan *et al.*, 2002)]. Our system utilizes multi-partner data delivering and targets to support on-demand streaming, which needs to handle asynchronous requests and more complicated VCR operations.

Cooperative multi-partner scheduling for P2P-based VoD streaming is difficult due to the limited bandwidth and the heterogeneity of content in different partners. Some simple heuristics such as round robin, smallest-delay, pull-based gossip algorithm (PGA) have been proposed to address this issue (Do *et al.*, 2004; Guo *et al.*, 2003; Zhang *et al.*, 2005; Zhou and Liu, 2005). PGA first calculates the number of potential partners for each segment, and then schedules the segments one by one in increasing order of those partners. Among the multiple potential partners, the one with highest bandwidth and enough available time is selected. Fig.2a depicts an example of PGA. In this example, the stream provided by the server is split into 6 segments *a*, *b*, *c*, *d*, *e*, and *f*. Peers *X*, *Y*, and *Z* form partner relationship for cooperative content delivery, and they will not send request to the server unless the partners can not provide the expected data. At Time 0 (*T*0), peer *X* requests *c* from the server; *Y* requests *c* from the server and *b* from *X* simultaneously; *Z* requests *c* from the server, *b* from *X*, and *a* from *Y* at the same time. After that, at time slot *T*1, *T*2, and *T*3, all the three nodes *X*, *Y*, and *Z* have the same content and need to request *d*, *e*, and *f*, respectively, from the server. Overall, PGA consumes 4 time-units and causes 12 traffic-units on the server. Though these heuristics are simple, their local and greedy decisions make inefficient use of network capacities. For example, the bandwidth among peer *X*, *Y* and *Z* has not been utilized since *T*1, while the server has to provide data for the three peers all the time.

Recently, network coding (NC) technology has brought fundamentally new insights and efficient solutions to the cooperative data scheduling problem (Gkantsidis and Rodriguez, 2005; Li *et al.*, 2005). With NC, each node linearly combines all received segments with random coefficients. When the number of received combinations is no less than the size of coding window (CW), which is the media size here, the media can be fully decoded with very high probability. In (Wang and Liu, 2005), it is proposed to directly apply linear NC into VoD streaming. As an example shown in Fig.2b, at time *T*0, *X* gets a linear combination among missing segments *c*, *d*, *e*, and *f*, denoted as *c~f*, from the server; *Y* gets *b~f* from the server and *b* from *X* simultaneously; *Z* gets *a~f* from the server, *a~b* from *X*, and *a* from *Y* at the same time. At time *T*1, as *Y* and *Z* have got linear combinations containing segment *c* to *f*, *X* can benefit from *Y* and *Z* besides the server. Similarly, both *Y* and *Z* also get three combinations at time *T*1. At time *T*2, *X* has four combinations plus two available segments *a*, *b*, so the original media can be reconstructed; similarly, both *Y* and *Z* can decode the entire media successfully. Overall, in the best condition, NC only consumes 2 time-units and generates 6 traffic-units on the server. Though NC improves the resource utilization and minimizes the total schedule time, some segments may miss the play deadline since they are not available till all the linear combinations are received and decoded. For instance, at time *T*1, segment *c* is missed for node *X*, and this segment-missing problem will be further aggravated when the number of segments increases. Thus, the coding window for NC should be carefully determined instead of simply covering the entire available media at each node.

Our proposed Deadline-aware Network Coding (DNC) algorithm addresses this segment-missing issue. DNC adopts the network coding method to improve throughput, and in the meanwhile dynamically adjusts the coding window by taking the play deadline into consideration. A simple and intuitive way to estimate the coding window for each peer is to set CW as the product of the number of partners that contains expected data and the remaining time-units before the most emergent play deadline. Fig.2c depicts the case of DNC using this simple estimation. At

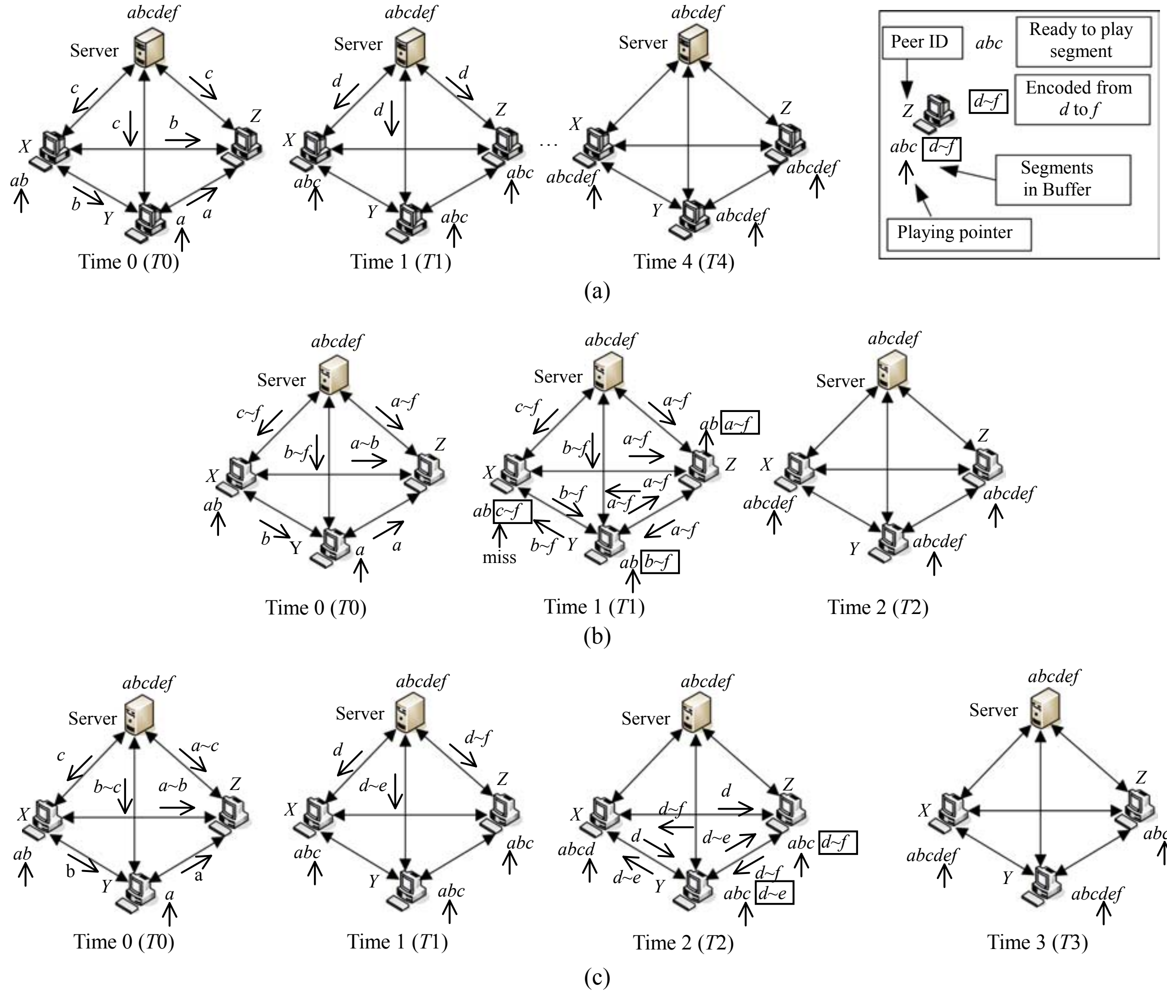

Fig.2 Pull-based gossip, network coding and deadline-aware network coding. (a) PGA; (b) NC; (c) DNC

time *T*0, *X* has one partner and one time-unit before *c*'s deadline, so the CW size for *X* is set as 1 and *X* gets *c* from the server; similarly, *Y* calculates its CW size as 2 and gets *b*~*c* from the server and *b* from *X* simultaneously; *Z* gets *a*~*c* from the server, *a*~*b* from *X* and *a* from *Y*. At time *T*1, *X* continuously gets *d* from the server; After decoding *b* and *c*, *Y* has only one partner, and the remaining time units before *d*'s deadline is 2, so it gets *d*~*e* from the server; similarly *Z* gets *d*~*f* from the server. At time *T*2, *X*, *Y*, *Z* become a partner of each other and exchange their data. At time *T*3, peer *X* reconstructs the original media from *a*, *b*, *c*, *d* and *d*~*e*, *d*~*f*; similarly *Y* and *Z* decode all segments. In summary, DNC only uses 3 time-units for downloading and costs 6 traffic-units on the server. The important thing here is that with deadline constrained in mind, using DNC no segment missing occurs. Thus DNC not only reduces the schedule time and server overhead, but also avoids the situation where segments miss the play deadline as in the previous NC case. The simple CW estimation we used in the above example may over-estimate the coding window when some partner cannot utilize full bandwidth due to its limited contents, so we will introduce a more intelligent way for CW estimation in the next section.

DEADLINE-AWARE NETWORK CODING (DNC) SCHEDULING

After locating the partners, a peer will collaborate with them to dynamically exchange/schedule data transmission together with them according to the content in their buffer and the corresponding bandwidth condition among them. In this section, we propose a Deadline-aware Network Coding (DNC) based approach for the multi-partner scheduling. We first introduce the DNC scheme and formulate the scheduling problem using DNC. We then present an optimal solution for such a problem and a distributed protocol accordingly.

DNC scheme and problem formulation

DNC considers the play deadline of data segments when doing network coding so that it can improve the network throughput and meanwhile satisfy the deadline requirement.

Assume there are k expected segments $\{m_i\}$ such as $m_1, m_2, \dots, m_k$. A random linear coding on a w-size subset $\{m_i'\}$ ($|\{m_i'\}|=w$, both are used interchangeably) of $\{m_i\}$ is a vector $\boldsymbol{c}_j=\sum_{i=1}^{w}\boldsymbol{\alpha}_{ji}m_i'$, where w is called the coding window (CW), and the coefficient vector $\boldsymbol{\alpha}_{ji}$ is randomly generated in a finite field $\boldsymbol{F}_q$ of size q. If all $\boldsymbol{c}_j$ are linearly independent, once a node has a subset of $\boldsymbol{c}_j$ that spans w, it can recover the w expected segments in $\{m_i'\}$ by solving a set of linear equations. In conventional network coding, w is always the same as the number of expected segments, i.e., $w=k$, and thus the coding range covers all expected segments. Though a larger w makes better utilization of resource, it causes higher possibility of segments missing upon the play deadline due to the longer waiting time before decoding. Thus the DNC scheme tries to use an as large as possible w for better coding efficiency while controlling w so that no segment will miss its play deadline.

DNC estimates w to be the maximum number of segments that can be retrieved from the partners before the most emergent deadline. Thus we formulate w assignment problem with DNC as follows. The segment size is assumed to be 1 unit for simplicity. At time t, suppose a peer has k data segments $\{m_i\}$ missing in the buffer each with a play deadline t_i, $i=1, 2, \dots, k$, and s partners each with a subset of data segments D_i and available bandwidth b_i, $i=1, 2, \dots, s$, and the problem then is to find an assignment $A=\{(i,j)\}$ for retrieving segment j from partner i, such that the number of received segments $\{m_i'\}$ is maximized before the most emergent deadline t_1 (Fig.3). We have two rules on the assignment.

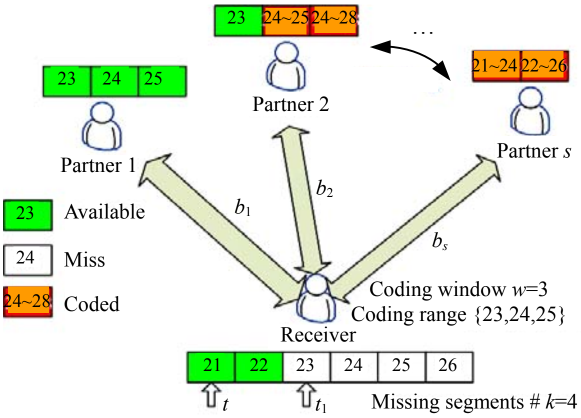

Fig.3 Analysis of CW assignment problem

(1) Any valid assignment does not assign the same segment request to more than one partner, since it makes no sense in receiving multiple copies of the same segment.

(2) The number of segments assigned to each partner cannot exceed the quantity that can be achieved by the partner, i.e., $f(i)=\lfloor b_i\times(t_1-t)\rfloor$.

The formulation is shown as follows:

INPUT: s subsets D_i of a finite set $\{m_i\}$, with $|\{m_i\}|=k$;

SOLUTION: A set $\{m_i'\}\subseteq\{m_i\}$ that consists of at most $f(i)=\lfloor b_i\times(t_1-t)\rfloor$ elements from each subset D_i;

GOAL: Maximize the cardinality of $\{m_i'\}$, i.e., $|\{m_i'\}|$.

Max-flow based optimal solution

In this subsection, we will construct a flow network $G=(V, E)$ to transform the DNC problem to the max-flow problem. Nodes S and T are the source and sink, respectively. Each subset D_i and each element m_j corresponds to node D_i and node m_j, respectively. The capacities of edges between S and D_i are $f(i)$. Between D_i and m_j, there is an edge with capacity 1 only if $m_j\in D_i$. The capacities of edges between T and m_j are all 1. Then the complexity of the node number and edge number is $|V|=s+k+2=O(s+k)$ and $|E|=s+k+\sum_{i=1}^{s}|D_i|=O(s\times k)$, respectively. Fig.4 depicts the flow network G. For a flow g in G, we denote the flow value of g as $|g|$, and the flow value from node x to node y as $g(x, y)$.

Theorem 1 If A is a valid assignment in the coding window problem, then there is a flow g in G with

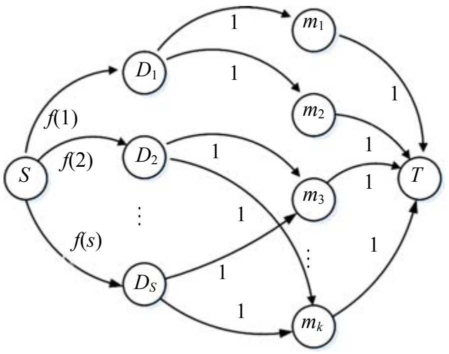

Fig.4 Constructed flow network *G*

$|g|=|\{m_i'\}|$. Similarly if g is a flow in G, then there is a valid assignment A with $\{m_i'\}=|g|$.

Proof For any valid assignment $A=\{(i,j)\}$, we can construct a flow g in this way. If $(i,j)\in A$, then let $g(D_i, m_j)=1$. The two rules presented in the last subsection ensure that this constructed flow g does not exceed the limits on the edges between S and D_i and those between T and m_j. Since each segment m_j assigned in A corresponds to a value-1 flow from node m_j to T, we have $|g|=|\{m_i'\}|$. A similar proof can be given in the reverse direction.

Thus our coding window assignment problem can be mapped to the max-flow problem in G. We can use the well-known max-flow algorithm to compute the coding window w and corresponding coding elements $\{m_i'\}$. Since the fastest known max-flow algorithm has time complexity of $O[|V||E|\log(|V|^2/|E|)]$ (Goldberg and Tarjan, 1988), the running time of our approach for coding window assignment is also $O\{s^2k\log[(s+k)^2/(sk)]\}$.

Distributed protocol

Each peer uses the max-flow approach to calculate the coding window w and corresponding coding elements $\{m_i'\}$, and then sends this request to all the partners. After decoding all requested segments in $\{m_i'\}$, the peer starts a new round of computation of the coding window and corresponding coding elements. The DNC scheduling algorithm is shown in Fig.5.

Upon receiving new segments, the peer examines whether the received combinations can be decoded. If they are decodable, they are reconstructed and the buffer is updated immediately. If the w expected segments decided in the last schedule have been fully recovered, the coding window and coding range are recomputed. And then the node requests combinations spanning the w expected segments in the coding range from each idle partner that contains unavailable data. Upon receiving a request from some neighbor, the node checks whether it is able to respond to an encoded segment computed from its available segments or received encoded segments or both. There are three cases that the node is able to provide useful segment for the receiver (see Fig.3 for the following examples). First, there are some available segments within the requested window range, e.g., partner 1 has the expected segment 23, 24, 25. Second, the node has some encoded segments whose coding range is within the requested window range, e.g., partner 2 has a coded segment 24~25 that is within the requested window range {23, 24, 25}. Third, there are some encoded segments that have some portion of coding range within the requested window range and the other portion corresponding to available segments in the requested node, e.g., 21~24 on partner 3.

```
for each node $n$ in the system
{
  if (any received combinations can be decoded)
      decode and update buffer;
  if (all the $w$ expected segments in $\{m_i'\}$ are decoded)
      recalculate the coding window $w$ and $\{m_i'\}$ ;
  for each idle partner $s_i$ with expected data in $\{m_i'\}$
      request length-$w$ combinations in $\{m_i'\}$ from $s_i$;
  if (any requests arrive) and
     ($n$ contains useful segments for the requests)
      responds the requests with encoded segments;
}
```

Fig.5 DNC scheduling algorithm

In summary, DNC generalizes traditional network coding in providing a dynamic coding window for each peer to avoid segment missing upon the play deadline, which is inherent in conventional network coding.

SIMULATION RESULTS

We evaluate the performance of the proposed

VoD system by comparing it with some existing systems.

Simulation configuration

We used the Ebone ISP topology collected by Rocketfuel engine for system setup (Spring *et al.*, 2002). It consists of 111 backbone nodes and 61 access nodes. The default bandwidth settings between two backbone nodes, a backbone and an access node, two access nodes are 10, 5, 5 Mbps, respectively.

In our simulation, the server and users are located at randomly selected access nodes. The communication path between any two nodes follows the shortest path. The bit rate of the streaming media is 500 kbps and its length is 720 s. The length of a segment is 1 s, and the default capacity of the user buffer can accommodate 20 segments. There is no user in the system at the beginning, and users join the system following a Poisson process with mean inter-arrival time of 2 s. The start offset of each user is evenly distributed between 0 and 720 s. The users leave the system once their play offsets reach the end of the streaming media. Fig.6 plots the number of online users in the system during a 1000-second simulation. It is seen that the system keeps increasing to about 180 users during the first 720 s, i.e., the streaming length. After that, the system size kept relatively stable, since the number of new users is closer to that of the leaving users.

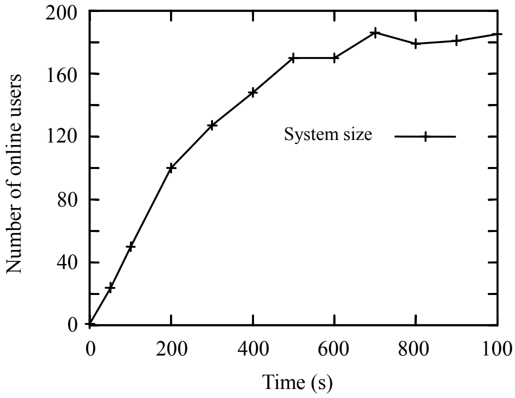

Fig.6 System size during 1000-second simulation

Streaming quality comparison

In this subsection, we compare the streaming quality of the VoD system using different partner scheduling algorithms. The start-up delay, which is the waiting time before the first playable segment is ready, and Segment Missing Ratio (SMR), which is the average ratio of segments that are not available upon their play deadline among all users, as well as system throughput, which is the total data that has been received at the receiver side, are used as the performance evaluation metrics. Three scheduling schemes, pull-based gossip algorithm (PGA), pure network coding (NC), and deadline-aware network coding (DNC), are implemented for comparison. In pure NC scheme, the coding window size is set to the default user buffer length, instead of the whole media length.

Fig.7 plots the start-up delay for VoD systems based on the three scheduling methods, PGA, NC and DNC, under the same network configuration and user setting. The start-up delay for NC is obviously larger than those of PGA and DNC because the first segment could not be reconstructed till enough coded segments are received. In the dynamic changing network, PGA may suffer from the congestion due to the greedy and local decisions, so the start-up delay is not stable. In contrast, the start-up delay in DNC is smaller and more consistent because the coding window is adaptive to the network condition and thus small upon peer arrival, and the network throughput utilization is improved with network coding.

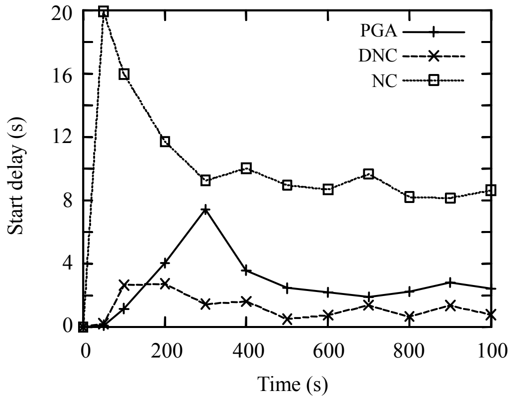

Fig.7 Start delay for PGA, NC and DNC

Fig.8 plots SMRs for PGA, NC and DNC systems during the first 700-second period of the 1000-second simulation, when the network condition is most dynamic. From no user at time 0, the system expanded to about 180 users (Fig.6). To emulate local bandwidth fluctuations, the available bandwidth at each link varied with time and the average value remained at 90% of the base setting. From Fig.8, we see SMR increased dramatically in the first 100 s. Since the server can serve at most 10 (5 Mbps/500 kbps) simultaneous connections and there are at most 36 (streaming-length/buffer-size) non-overlapping online users, the users are congested at the server side at the early stage when the system has more than 10 non-overlapping users and there are few users filled with segments. Compared with PGA, DNC and NC quickly reduce the SMRs to reasonable degree due to the power of network coding in improving the network throughput. With content play deadline in mind, DNC can obtain the smallest SMR.

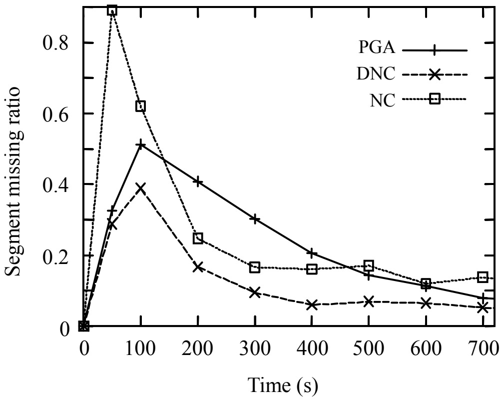

Fig.8 SMRs for PGA, NC and DNC

Fig.9 depicts the network throughput comparison for these three schemes. The throughput metric is the average number of segments a peer received per second. From Fig.9, the throughput of DNC and NC outperforms PGA by more than 20%. The NC always gets high throughput, but due to the changing and large coding window, many segments can not be decoded when they are required to be played and thus even though the segments eventually can be received, some of them are wasted. DNC leverages the throughput improvement and also considers the deadline to change the coding window to make as many segments as possible decodable upon their deadline. So even though DNC gets uncomfortable SMR at the initial stage, it can adaptively reduce SMR into around 5% soon and such loss can be effectively recovered by applying Forward Error Correction (FEC) (Frossard and Verscheure, 2001).

In summary, compared with PGA and NC, DNC not only provides small start-up delay, but also provides good streaming quality by adapting to the changing network conditions.

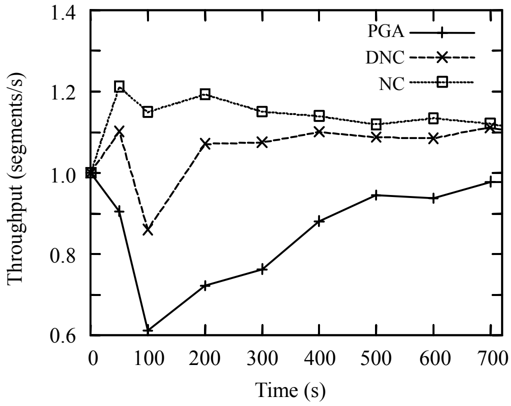

Fig.9 Throughput comparison

CONCLUSION

In this paper, we have presented one novel scheme, deadline-aware networking coding (DNC), to improve the data schedule efficiency for VoD system. DNC uses network coding to improve the network throughput utilization and keep high streaming continuity by keeping in mind media play deadline. Our simulation results showed that our VoD system outperforms the existing systems in terms of start-up delay, segment missing ratio, and also the effective throughput. Our future work is to run the experiments on the planet-lab test bed and build a VoD system prototype.

ACKNOWLEDGEMENT

We sincerely thank Dr. Xueyan Tang for his helpful suggestions on the optimization problem.

References

Aggarwal, C.C., Wolf, J.L., Yu, P.S., 1996. On Optimal Batching Policies for Video-on-Demand Storage Servers. Proceedings of the International Conference on Multimedia Computing and Systems. Hiroshima, Japan, p.253-258.

Castro, M., Druschel, P., Kermarrec, A., Nandi, A., Rowstron, A., Singh, A., 2003. Split-Stream: High-Bandwidth Multicast in Cooperative Environments. Proceedings of the

19th ACM SOSP. Bolton Landing, NY.

Chu, Y., Rao, S., Zhang, H., 2000. A Case for End System Multicast. Proceedings of ACM SIGMETRICS. Santa Clara, CA, USA.

Cui, Y., Li, B.C., Nahrstedt, K., 2004. oStream: asynchronous streaming multicast. *IEEE Journal on Selected Areas in Communications*, **22**(1):91-106. [doi:10.1109/JSAC.2003.818799]

Dan, A., Sitaram, D., Shahabuddin, P., 1996. Dynamic batching policies for an on-demand video server. *Multimedia Systems*, **4**(3):112-121. [doi:10.1007/s005300050016]

Do, T., Hua, K.A., Tantaoui, M., 2004. P2VoD: Providing Fault Tolerant Video-on-Demand Streaming in Peer-to-Peer Environment. Proc. IEEE ICC'04. Paris.

Eager, D., Vernon, M., Zahorjan, J., 2000. Bandwidth Skimming: A Technique for Cost-effective Video-on-Demand. Proceedings of SPIE/ACM Conference on Multimedia Computing and Networking (MMCN). San Jose, CA, USA.

Frossard, P., Verscheure, O., 2001. Joint source/FEC rate selection for quality-optimal MPEG-2 video delivery. *IEEE Trans. on Image Processing*, **10**(12):1815-1825. [doi:10.1109/83.974566]

Gao, L., Towsley, D., Kurose, J., 1998. Efficient Schemes for Broadcasting Popular Videos. Proceedings of the 8th ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV). Cambridge, UK.

Gkantsidis, C., Rodriguez, P., 2005. Network Coding for Large Scale Content Distribution. INFOCOM 2005. Miami.

Goldberg, A.V., Tarjan, R.E., 1988. A new approach to the maximum-flow problem. *Journal of the ACM*, **35**(4):921-940. [doi:10.1145/48014.61051]

Guo, Y., Gao, L., Towsley, D., Sen, S., 2002. Seamless Workload Adaptive Broadcast. Proceedings of IEEE International Packet video Workshop. Pittsburgh, PA, USA.

Guo, Y., Suh, K., Kurose, J., Towsley, D., 2003. P2Cast: Peer-to-Peer Patching Scheme for VoD Service. Proceedings of the 12th World Wide Web Conference (WWW-03). Budapest, Hungary.

Hefeeda, M., Bhargava, B., Yau, D., 2004. A hybrid architecture for cost effective on demand media streaming. *Journal of Computer Networks*, **44**(3):353-382. [doi:10.1016/j.comnet.2003.10.002]

Hu, A., 2001. Video-on-Demand Broadcasting Protocols: A Comprehensive Study. Proceedings of IEEE INFOCOM, Anchorage. AK, USA.

Hua, K., Cai, Y., Sheu, S., 1998. Patching: A Multicast Technique for True Video-on-Demand Services. Proceedings of the 6th ACM International Conference on Multimedia (MM). Bristol, England.

Kostic, D., Rodriguez, A., Albrecht, J., Vahdat, A., 2003. Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh. Proc. of the 19th ACM Symposium on Operating Systems Principles (SOSP).

Li, Z., Li, B., Jiang, D., Lau, L.C., 2005. On Achieving Optimal Throughput with Network Coding. INFOCOM 2005. Miami.

Padmanabhan, V., Wang, H., Chou, P., Sripanidkulchai, K., 2002. Distributing Streaming Media Content Using Co-operative Networking. Proceedings of ACM NOSSDAV.

Quinn, B., Almeroth, K., 2001. IP Multicast Applications: Challenges and Solutions. Internet Engineering Task Force (IETF) Internet Draft, work in progress.

Rejaie, R., Ortega, A., 2003. PALS: Peer-to-Peer Adaptive Layered Streaming. Proceedings of ACM NOSSDAV.

Spring, N., Mahajan, R., Wetherall, D., 2002. Measuring ISP Topologies with Rocketfuel. Proceedings of ACM SIGCOMM'02, p.133-145.

Wang, D., Liu, J., 2005. A Dynamic Skip List Based Overlay Network for On-Demand Media Streaming with VCR Interactions. Technical Report.

Zhang, X., Liu, J., Li, B., Yum, T.S.P., 2005. DONet/Cool-Streaming: A Data-Driven Overlay Network for Live Media Streaming. IEEE INFOCOM'05. Miami, USA.

Zhou, M., Liu, J., 2005. Tree-Assisted Gossiping for Overlay Video Distribution. Kluwer Multimedia Tools and Applications.