Journal of Software Engineering

ISSN 1819-4311

Academic Journals Inc.

www.academicjournals.com

Journal of Software Engineering 7 (4): 121-132, 2013
ISSN 1819-4311 / DOI: 10.3923/jse.2013.121.132

Simulative Software Architecture of P-CDN System with Extendable Interface and Loading Balance Mechanism

Huang Yongsheng, Li Yuguang and Du Huamei
Tangshan Key Laboratory of Informationization Technologies and Engineering Control, School of Management, Hebei United University

Corresponding Author: Huang Yongsheng, Tangshan Key Laboratory of Informationization Technologies and Engineering Control, School of Management, Hebei United University

ABSTRACT

The simulative software is the important tool for demonstrating the performances of the P-CDN. Accompanying with the increasing complexity of the P-CDN, extendibility and distributed processing based on multiple machines become the basic requirements of the simulation of the P-CDN. In this study, the simulative software architecture with hierarchical extendable interfaces is put forward to meet the need of diverse extending. Meanwhile, the multiple machines based distributed processing is implemented with the Markov Chain model involved to balance the loading of the machines. Experimental results indicate the proposed software architecture can achieve better performance even when the scale of peers is larger.

Key words: P2P, CDN, simulative software architecture, loading balance, extendibility

INTRODUCTION

With the development of the network technologies, the content sharing is becoming the mainstream application (Ganjam *et al*., 2010). In the research field of content sharing, the Peer to Peer (P2P) network and the Content Delivery Network (CDN) are widely accepted for content storage and dispatch (Fujita *et al*., 2008). In order to conform to the diverse requirement in network circumstances, the hybrid structure, P2P CDN (P-CDN) is generally taken as an excellent topology in this field (Hu *et al*., 2012; Huang *et al*., 2012). As the P-CDN usually consists of several servers and a great many clients, it is not practical that the efficiency of a new proposed P-CDN model is testified by the actual network. Accordingly, the simulative software is necessary for the research on the field of P-CDN (Fortino and Mastroianni, 2008; Xu *et al*., 2006).

In the research field of P2P, a lot of simulative structures were proposed accompanying with the updated protocols, topologies and performance optimization schemes for P2P network (Pogkas *et al*., 2009; Qian *et al*., 2011). For the sake of testifying the new architecture with a large-scale on-demand media streaming service (Hefeeda *et al*., 2004) put forward an extensive simulation structure in which the service capacity, client-side buffering and patterns proposed can be simulated integrally. To the research on CDN, the simulators are the tools being necessarily chosen to indicate the achievement. To demonstrate the performance of combating bandwidth fluctuation on the Internet, a simulator is designed by Nguyen *et al*. (2010). In the simulative software, the network coding techniques is implemented with the TCP being integrated for testifying the bandwidth saving of the proposed model.

In contrast with P2P network or CDN, the simulation to the P-CDN is more complex as its diverse constitution. In the study (Baccaglini *et al*., 2012), the PlanetLab network is used to encompass real network behaviors by the simulation of the peers and CDN according to the proposed NextShare model. To show the efficiency of the proposed model based on the predictive control scheme, the stream of multimedia and interactive grid data are considered in the simulative architecture (Caviglione and Cervellera, 2011).

To the P-CDN model with complicated network management and computation in the servers and the peers, the simulative software specially developed for designated application is usually a time-cost work and may be useless or unusable for further work (Shen and Zhu, 2013; Wahlisch *et al*., 2011). Additionally, the simulative software running on single machine is usually not comfort to the requirement of the simulation to the P-CDN with large scale of peers (Lin and Lee, 2010). Accordingly, the simulative software architecture with extendable interfaces based on multiple machines becomes the demand of the simulation to P-CDN in the aftertime. In this study, the simulative software architecture is proposed. To conform to the need of the extendibility, a structure with multiple hierarchies is designed. From the bottom to the top, the functions of the simulation are implemented in consideration of full extendibility. On the other hand, the integrated control is accomplished from the top to the bottom according to the integrality of the functions.

The simulation of the peer: As the control module is necessary to any simulative system and the functions of the central server are also tasks of system control, the simulation to the central server can be imbedded into the control module (Guo *et al*., 2006). The basic assignments of the simulation to P-CDN are implementing the simulation of the peer and the surrogate server.

The simulation of the peer consists of the data simulation and the activity simulation which are addressed by the upper portion and the lower portion of the Fig. 1, respectively. The data simulation takes charge of the information management of the peer and the activity simulation is used to implement the core functions of the peer in the P-CDN system. From another point of view, the simulative content of a peer can be divided into two categories: file-related management and connection related management which are denoted by the left portion and the right portion of the Fig. 1, respectively. The components in the peer simulation are addressed as follows.

File Information List (FIL): The information of files stored in the peer is listed by FIL. The information of each file consists of the file ID, the file name and the file size etc.

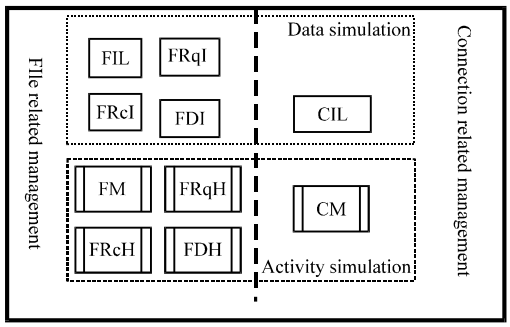

Fig. 1: The simulative structure of the peer

File Request Information (FRqI): When a file is required by the peer, the file request is initialized and sent to the peers connected to the peer. Before the procedure of the file request is finished, the information about the file request which consists of the file request ID, the file ID and the other necessary data is stored in FRqI.

File Receiving Information (FRcI): When a requesting file is found and then the file will be transferred from other peers, the information about the procedure of the file receiving is stored in FRcI.

File Dispatching Information (FDI): When a file requested by another peer is found in the peer, the file will be dispatched. The information about the procedure of the file dispatching will be stored in FDI.

Connection Information List (CIL): In the P-CDN system, a peer will selected a certain number of surrogate servers and peers according to the designated rule. The information about the surrogate servers and the peers connected to the server is listed in CIL.

File Management (FM): The FM takes charge of simulating the file adding, deleting and updating of the file. And meanwhile, the FIL will be updated.

File Request handling (FRqH): FRqH is used to simulation of initializing and relaying file request. As the peer requires a file and needs to search the file in the P-CDN, a file request is initialized by FRqH and sent to other peers. Besides, a relay of a fie request will be executed by FRqH when the peer receives a file request and the file request should be relayed according to the designated rule.

File Receiving Handling (FRcH): As a file required is found, the FRcH is started and takes charge of cooperating with the peers sending the file to the peer.

File Dispatch Handling (FDH): As a file should be dispatched to another peer, the FDH is executed to control the dispatching procedure.

Connection Management (CM): During the running of the P-CDN, the connection between the peer and the surrogate server or another peer is built or deleted by the CM.

The simulation of the surrogate server: In the simulation system, the function of the simulative surrogate server is simplified. Corresponding to the designated rules, part of the available files will be stored into a surrogate server by simulative format. As a result, adding file and deleting file is its basic simulative functions. It is not necessary for the surrogate server to search for the peers proactively and so the surrogate server should not manage the connection with the peers. However, the simulative surrogate server should handle the activity of file dispatch in order to demonstrate the information about file dispatch according to a certain simulative requirement. These simulations are the same to those of the peer.

The control module of the simulative CDN system: To the simulative system, the control module takes charge of initializing, operating and terminating the simulative system. The

information for the control to the simulative system should be input from external (Wu and Starobinski, 2008). Consequently, the control module can be divided into four sub-modules: the module of external interface, the module for initialization, the module for operation and the module for termination. The simulative functions of the control module are listed below.

Module of External Interface (MEI): The initial structure and the file distribution of the simulative system are formed according to external commands. The categories, quantity and the formats of the external commands are defined in MEI. The command interpretation is another function of the MEI. Thus, MEI is the intermediate between the simulative system and the external.

Module of Initialization (MI): The initialization of simulative system consists of System Available Resource Initialization (SARI) and the Initial Structure and File Distribution Initialization (ISFDI). The configuration to the usable machines (microcomputers and/or servers) and the network interconnecting the machines is managed by SARI. ISFDI takes charge of allocating tasks to the usable machine beforehand.

Module of Operation (MO): In the running of the simulative system, the activities occurring can be divided into two categories. The first one consists of the activities started by the external commands and the activities started by other activities comprise the second one. For the activities of the first category, MO transfers the external commands to the activities. To the activities of the second category, MO takes charge of bringing the objects related to the activities under control.

Module of termination (MT): When the tasks of the simulative P-CDN are fulfilled, the resources occupied by the simulative system should be released and the simulative results should be output by a proper format. The MT inspects the resource occupation of the simulative system before termination and release the resources unreleased. Finally, the simulative system is terminated by MO.

The overall architecture of the simulative software: In order to comfort to the extendable requirement, the over architecture of the simulative software is divided into four layers. The first layer is the interface layer in which the sketch of the functions is formulated by interfaces. In the second layer, the uniform abstract classes are designed to implement the common functions and define the extendable functions. For the third layer, an implemented scheme to the abstract classes is provided. In the forth layer, the components of an integrated simulative system can be directly deployed to the distributed processing environment are proposed. The components in the four layers are addressed in detail by Fig. 2 as following.

Interface for Control (IC): IC uniformly defines the methods extendable and waiting for implementation in order to be used uniformly for simulative system control in higher layers.

Interface for Activity (IA): IA uniformly defines the methods extendable and waiting for implementation in order to be used uniformly for cooperation among the entities of the simulative system.

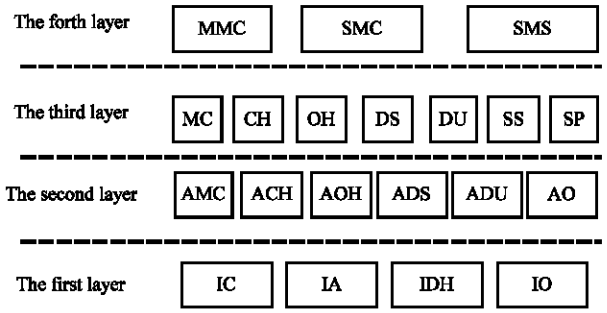

Fig. 2: The hierarchical structure of the simulative software for P-CDN

Interface for Data Handling (IDH): IDH is responsible for designing the abstract methods which handle the common data of the simulative system.

Interface for Object (IO): By IO, the uniform abstract methods of the objects in the simulative system are designed to handle the common methods uniformly.

Abstract class for Machine Control (AMC): AMC is in charge of designing/implementing the methods control the using approaches of the available machines and the network resources. In the methods, the abstract ones are waiting to further implement and the others can be used directly and be rewritten for peculiar purposes.

Abstract class for Control Handling (ACH): According to the requirement of operating control for common and extendable usage, the attributes of ACH is designed and the activities (methods) fall into abstract methods, final methods and rewritable methods.

Abstract class for Data storage (ADS): ADS answers for designing the formats of data storage in the objects defined in the four layers and extended objects. Additionally, the data storage approaches delimit the calling methods and procedures about the key data in the simulative system.

Abstract class for Data Update (ADU): The data needing to update related to an activity is commonly distributed to a number of objects which may be also to many a machine in the simulative system. Consequently, it is necessary that the handling strategies should be drawn out by ADU utilizing abstract methods and rewritable methods.

Abstract class for Object (AO): The uniform entity about the simulative server and the simulative peer is formulated in order to being uniformly used in simulative system, especially when the objects are extended. Consequently, the common data and methods accompanying the calling procedures of them should be drawn out in AO.

Class for Machine Control (MC): MC is an implemented scheme for the control of machines in simulative system.

Class for Control Handling (CH): CH is an implemented scheme for the control of the active objects such as surrogate servers, peers and other extended objects when the simulative system is running.

Class for Data storage (DS): DS is an implemented class for the ADS, in which the abstract methods are implemented and the rewritable methods are rewritten according to designated purposes.

Class for Data Update (DU): According to general purpose of the simulation to P-CDN, DU is an implemented scheme for ADU. In DU, the method corresponding to ADU is implemented or rewritten conformed to general usage.

Class for Surrogate Server (SS): The simulative surrogate server is implemented corresponding to AO according to the requirements of surrogate servers in P-CDN.

Class for Simulative Peer (SP): The simulative peer is implemented according to AO conformed to the requirements of the peers in P-CDN.

Master Machine Control (MMC): For the simulative P-CDN, a machine should be selected to be the master machine to control the simulative system. MMC is implemented by the relative control functions being incorporated into and the abstract methods defined in the other layers for control being implemented.

Single Machine Control (SMC): To the simulative P-CDN running on more than one machine, SMC is implemented to manage the communications among machines.

Single Machine Simulator (SMS): SMS is in charge of building the objects should be simulated in a designated machine. And meanwhile, creating and maintaining the relations among the objects with the simulative system running.

The control and balancing of machine loading: To the simulative P-CDN, the purpose that the simulative objects are distributed to multiple machines can solve the problem that only one machine usually can not bare the simulation about the P-CDN systems with large scale of simulative objects (Laoutaris *et al*., 2007). However, the cost of the network resources and the expenditure for collaboration among machine raise with the number of the machines used increasing. Consequently, it is more proper that the simulation is based on single machine when the P-CDN being simulated has smaller scale of objects comparatively. Only when the P-CDN being simulated maybe have larger scale of objects, the simulation software based on multiple machines is necessary. To the P-CDN system with the scale of objects varying from small to large enough during the running period, there are two options which be selected from when the scale of the objects is small. To the first one, the simulation is based on one machine of the available machines when the scale of the objects is small. When the scale of the objects becomes large enough, the other machine will be utilized. To the other option, the simulative objects are distributed to the available machines evenly whenever the scale of the system is small or large. In the proposed simulative software, the two options are selectable by the corresponding design in MMC.

In the proposed simulative system, balancing the peers loaded in the available machines is the key point which is determinative to the balancing of the usable resources of the system. For simulation, the time expenditure of the simulative procedure is finite. Consequently, the duration of the simulative can be denoted as a sequence of time periods $TS = (t_1, t_2,..., t_n)$ in which each of the

Table 1: State transfer of the peer

t_i	t_{i+1}
0	1
0	0
1	0
1	1

time periods is a steady duration of time according to the requirement of loading balance. Let $PR = \{pr_1, pr_2,..., pr_s\}$ denote the set of peers which can be appeared in P-CDN and then the state of a peer can be expressed by the Eq. 1:

$$state(pr_g) = \begin{cases} 1, & \text{if the } pr_g \text{ is active} \\ 0, & \text{if the } pr_g \text{ is not active} \end{cases} \quad (1)$$

When the state of a peer is 0, the peer is not active and is not included in the P-CDN and on the contrary, the peer is active and included in the P-CDN when the state of a peer is 1. The state transfer of a peer can be addressed by the Table 1 when the time period is changed from t_i to t_{i+1}.

By the definition about, the state of the peer in t_{i+1} is only relative to t_i and not relative to the time periods from t_1 to t_{i-1} which is the attributes of the Markov Chain. In order to balance the resource loading of the system, the predictive model of Markov Chain is taken to involve the control procedure (Melnik, 2008).

To conveniently describe the number of active peers in a machine, the set $M = \{m_1, m_2,..., m_s\}$ is selected to represent the machines in the simulative system. The probability to the state transfer is expressed as the Eq. 2:

$$P_{xy}(pr_s) = a_{xy}, \quad 0 \le a_{xy} \le 1, \quad x, y \in \{0, 1\} \quad (2)$$

According to the definition above, the conclusion can be made by the Eq. 3 in the following:

$$\sum_{y=0}^{1}\sum_{x=0}^{1} P_{xy}(pr_g) = \sum_{y=0}^{1}\sum_{x=0}^{1} a_{xy} = 1 \quad (3)$$

In order to balance the count of the active peers, all active peers at the initialing time are allocated to the usable machines equably. Consequently, the number of the active peers is approximately equal in the time period t_1. According to the requirement of the Markov Chain with stationary distribution, the matrix of transfer probability can be represented as follows (Yeh *et al.*, 2010):

$$\begin{array}{c} \\ s_0 \\ s_1 \end{array} \begin{array}{c} \begin{array}{cc} s_0 & s_1 \end{array} \\ \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \end{array}$$

where, $[s_0, s_1]$ is the initial probability distribution. For the Markov Chain with stationary distribution, the Eq. 4 and 5 are tenable at the same time:

$$[s_0, s_1]\begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} = [s_0, s_1] \tag{4}$$

$$s_0 + s_1 = 1 \tag{5}$$

The number of the active peers at the time period t_1 can be calculated by the Eq. 6. In the equation $|PR|$ is the number of elements in the set PR:

$$sum_1 = |PR| \times s_1 \tag{6}$$

To sum_1, the count of the active peers at the time period t_i, the count calculated by the Eq. 7 is used for the number of active peers allocated to every machine in consideration of sum_1 being divided by f with remainder probably. F is the count of the usable machines:

$$sum_1(m_h) \approx \frac{sum_1}{f} \tag{7}$$

In the simulative system, the peers can be divided into multiple categories according to the variation of their action in P-CDN which results in the differentiation of the probability distribution. Accordingly, the set PR is divided into a sequence of subsets: $sub(PR)_1$, $sub(PR)_2$.., $sub(PR)_w$. The calculation to count of $sum_1(m_h)$ is modified and expressed by the Eq. 8. $S^j{}_1$ is put forward to denote the value of s_1 for the peers in the subset $sub(PR)_j$:

$$sum_1(m_h) \approx \frac{1}{f}\sum_{j=1}^{w} |sub(PR)_j| \times s_1^j \tag{8}$$

In order to balance the counts of the active peers in the available machines accompanying with the transfer of the time period, $at(m_h, pr_g)$ is put forward to denote the relationship at a certain time period t_i. The relationship is addressed by the Eq. 9:

$$at_i(m_h, pr_g) = \begin{cases} 0, & \text{if } pr_g \text{ is not active or active but not in } m_h \\ 1, & \text{if } pr_g \text{ is active in } m_h \end{cases} \tag{9}$$

As a result, the number of the active peers in every machine at the time period t_i can be denoted by the Eq. 10:

$$sum_i(m_h) = \sum_{g=1}^{s} at_i(m_h, pr_g) \tag{10}$$

Combined with the probability of the state transfer, the number of the peers which is active in the machine m_h in the time period t_i and is not active in the period t_{i+1} can be achieved by the Eq. 11:

$$\text{Disappear}_i(m_h) = \sum_{g=1}^{s}\{at_i(m_h, pr_g) \times \sum_{y=0}^{1}\sum_{x=0}^{1}[x(1-y) \times P_{xy}(pr_g)]\} \quad (11)$$

Correspondingly, the number of peers which is active in the machine m_h in the time period t_i and is also active in the period t_{i+1} can be acquired by the Eq. 12:

$$\text{Still}_i(m_h) = \sum_{g=1}^{s}\{at_i(m_h, pr_g) \times \sum_{y=0}^{1}\sum_{x=0}^{1}[xy \times P_{xy}(pr_g)]\} \quad (12)$$

Additionally, the number of peers which is not active in the machine m_h in the time period t_i but is active in the period t_{i+1} can be obtained by the Eq. 13:

$$\text{Appear}_i(m_h) = \sum_{g=1}^{s}\{at_i(m_h, pr_g) \times \sum_{y=0}^{1}\sum_{x=0}^{1}[(1-x)y \times P_{xy}(pr_g)]\} \quad (13)$$

In consideration of all active peers in the simulative system, the number of active peers after balanced at the time period t_{i+1} can be calculated by the Eq. 14:

$$\text{Sum}_{i+1}(m_h) = \frac{1}{f}\sum_{h=1}^{1}[\text{appear}_i(m_h) + \text{still}_i(m_h) - \text{disappear}_i(m_h)] \quad (14)$$

In order to reduce the expenditure of the balancing, the newly active peers is preferentially allocated to the machine whose value of $sum_{i+1}(m_h)$ is larger than that of $still_i(m_h)$. And then the peers may be transferred among the machines according to the requirement of the balancing.

The performance analysis of the proposed simulative software architecture: In the research field of P2P or P-CDN, the variation of the constitutions and the network architectures result in only the simulative software with extendable interfaces can meet requirements of the diversity of the research directions. Comparative to simulative architectures for special purposes, the proposed architecture in this study can be used for versatile purposes after extending. In contrast to restricted extendibility because of the coupling of the constitutions (Fu *et al*., 2006; Oztoprak and Akar, 2008), the full extendibility is accomplished by the hierarchical architecture with implementation from the bottom to the top and the reverse controlling in the proposed software.

For the simulative system based multiple machines, the loading balance is usually determinative to the efficiency about the simulation of P-CDN with large scale of peers (Shen and Xu, 2009; Tsurumi *et al*., 2009). As the loading balance should be effective from the beginning to the end, the performance of loading balance can be reflected by the continuous variation in respect to CDAPM (the Count Differentiation of the Active Peers in the Machines). The CDAPM is addressed by the Eq. 15:

$$\text{CDAPM} = \sqrt{\frac{1}{f}\sum_{h=1}^{f}[\text{sum}_{rt}(m_h) - \overline{\text{sum}_{rt}}]^2} \quad (15)$$

The balancing mechanism is aimed at the adjustment to the count of peers for machines at a time point in designated time period. Consequently, the value of $sum_i(m_h)$ can not mean the count

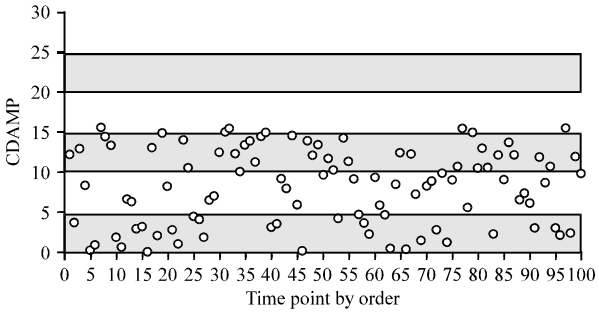

Fig. 3: The value distribution of CDAPM in the simulative procedure with 10000 peers

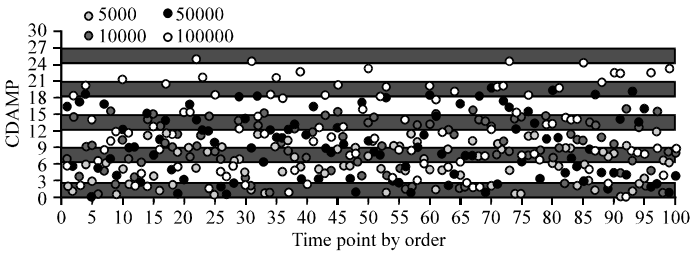

Fig. 4: The comparison of the CDAPM with the variation of the scale of peers

of the active peers for m_h is not varied in the time period t_i. In order to analysis the balance performance of the proposed architecture, $sum_n(m_h)$ is the count of the active peers for m_h at a time point rt randomly selected regardless of the partition of the time periods. Based on the proposed software architecture, a simulative P-CDN with about 10000 peers and 6 available machines is executed as an experiment. In the simulative procedure, 100 time points are randomly selected by order. In Fig. 3, the variation of CDAPM is shown. From the results shown, the values of the CDAPM are varied continuously but the maximum value is restrained to a limited value zone. To this simulative procedure, the CDAPM is smaller comparative to 10000 peers as the maximum value is less than 16.

The balancing performance of the proposed software architecture with the scale of peers increasing is exhibited by the Fig. 4. In order to indicate the influence of the scale of peers in P-CDN to CDAPM, the simulative P-CDN is executed under the same circumstance with the above experiment beside that the scale of the peers is change to 5000, 50000 and 100000, respectively. The results shown in the figure indicate that the variation of the value zone for the CDAPM is growing with the scale of the peers increasing but the maximum of CDAPM is still is restrained to a smaller value comparative to the value of the scale.

Summarily, the proposed software architecture for P-CDN simulation can not only implement extendibility by the hierarchical structure but has better performance of loading balance by the embodied mechanism.

CONCLUSION

To conform to the requirement of the simulation to P-CDN for multiple purposes, the simulative software architecture is proposed based on hierarchical structure. By the four hierarchies, the P-CDN can be implemented and extended from the bottom to the top. And meanwhile, the information about organization and control can be transmitted from the top to the bottom. As a

results, the extendable requirement of the simulation to P-CDN is fully guaranteed with the integrate functions. In order to balance the loading of multiple machines available to the simulation, a loading balance mechanism utilizing the Markov Chain model is proposed. The experimental results indicate that the proposed mechanism can achieve better performance of the loading balance and the better performance will be kept even when the scale of the peers grows larger.

ACKNOWLEDGMENT

This study was funded by the Science and Technology Department of Tangshan City (No. 12140201A-7) and the Key Projects in Science and Technology of Hebei Province (No. 13227503D-2).

REFERENCES

Baccaglini, E., M. Grangetto, E. Quacchio and S. Zezza, 2012. A study of an hybrid CDN-P2P system over the planet lab network. Signal Process. Image Commun., 27: 430-437.

Caviglione, L. and C. Cervellera, 2011. Design, optimization and performance evaluation of a content distribution overlay for streaming. Comput. Commun., 34: 1497-1509.

Fortino, G. and C. Mastroianni, 2008. Special section: Enhancing content networks with P2P, Grid and Agent technologies. Future Gener. Comput. Syst., 24: 177-179.

Fu, H., N. Wakamiya and M. Murata, 2006. A cooperative mechanism for hybrid P2P file-sharing networks to enhance application-level QoS. IEICE Trans. Communi., E89-B: 2327-2335.

Fujita, Y., D. Mori, Y. Saruwatari and K. Tsuda, 2008. Reverse-query diffusion over unstructured overlay network for content delivery. Int. J. Comput. Appli. Technol., 33: 131-137.

Ganjam, A., S.G. Rao, K. Sripanidkulchai, J. Zhan and H. Zhang, 2010. On-demand waypoints for live P2P video broadcasting. Peer-to-Peer Network. Appli., 3: 277-293.

Guo, L., S. Chen and X. Zhang, 2006. Design and evaluation of a scalable and reliable P2P assisted proxy for on-demand streaming media delivery. IEEE Trans. Knowl. Data Eng., 18: 669-682.

Hefeeda, M.M., B.K. Bhargav and D.K.Y. Yau, 2004. A hybrid architecture for cost-effective on-demand media streaming. Comput. Networks, 44: 353-382.

Hu, C., M. Chen, C. Xing and B. Xu, 2012. EUE principle of resource scheduling for live streaming systems underlying CDN-P2P hybrid architecture. Peer-to-Peer Network. Appli., 5: 312-322.

Huang, Y., H. Du and G. Zhang, 2012. Clustering model of P2P CDN based on the prediction of user requirements. J. Networks, 7: 532-539.

Laoutaris, N., G. Smaragdakis, A. Bestavros, I. Matta and I. Stavrakakis, 2007. Distributed selfish caching. IEEE Trans. Parallel Distrib. Syst., 18: 1361-1376.

Lin, C.S. and I. Lee, 2010. Applying multiple description coding to enhance the streaming scalability on CDN-P2P network. Int. J. Commun. Syst., 23: 553-568.

Melnik, R.V.N., 2008. Markov chain network training and conservation law approximations: Linking microscopic and macroscopic models for evolution. Applied Math. Comput., 199: 315-333.

Nguyen, K., T. Nguyen and S.C. Cheung, 2010. Video streaming with network coding. J. Signal Process. Syst., 59: 319-333.

Oztoprak, K. and G.B. Akar, 2008. Hybrid fault tolerant peer to peer video streaming architecture. IEICE Trans. Commun., E91-B: 3627-3638.

Pogkas, I., V. Kriakov, Z. Chen and A. Delis, 2009. Adaptive neighborhood selection in peer-to-peer networks based on content similarity and reputation. Peer-to-Peer Network. Appli., 2: 37-59.

Qian, J., J. Yin, J. Dong and D. Shi, 2011. JTangCSPS: A composite and semantic publish/subscribe system over structured P2P networks. Eng. Appli. Artificial Intelli., 24: 1487-1498.

Shen, F. and Y. Zhu, 2013. A hybrid P2P-CDN based on locality-awareness and interest-awareness. Inf. Technol. J., 12: 403-408.

Shen, H. and S. Xu, 2009. Coordinated en-route web caching in multiserver networks. IEEE Trans. Comput., 58: 605-619.

Tsurumi, H., T. Miyata, K. Yamaoka and Y. Sakai, 2009. Local optimal file delivery scheduling in a hop by hop file delivery system on a one link model. IEICE Trans. Commun., E92.B: 34-45.

Wahlisch, M., T.C. Schmidt and G. Wittenburg, 2011. On predictable large-scale data delivery in prefix-based virtualized content networks. Comput. Networks, 55: 4086-4100.

Wu, T. and D. Starobinski, 2008. A comparative analysis of server selection in content replication networks. IEEE/ACM Trans. Networking, 16: 1461-1474.

Xu, D., S.S. Kulkarni, C. Rosenberg and H.K. Chai, 2006. Analysis of a CDN-P2P hybrid architecture for cost-effective streaming media distribution. Multimedia Syst., 11: 383-399.

Yeh, H.W., W. Chan, E. Symanski and B.R. Davis, 2010. Estimating transition probabilities for ignorable intermittent missing data in a discrete-time markov chain. Commun. Stat. Simul. Comput., 39: 433-448.