CCF Transactions on Networking (2019) 1:28–36
https://doi.org/10.1007/s42045-019-00016-8

REGULAR PAPER

Peer-to-peer file sharing in next generation eXpressive Internet Architecture

Ziqian Meng[1] · **Zhong Chen**[1] · **Zhi Guan**[2]

Received: 5 August 2017 / Accepted: 7 February 2019 / Published online: 29 April 2019

Abstract
As a popular peer-to-peer protocol, BitTorrent is one of the most important methods for file sharing and distributing in today's Internet. However, the BitTorrent protocol is built entirely at the application level. Without the support of network layer, peers in BitTorrent protocol have to exchange data content through TCP connections and BitTorrent applications has to handle every procedure in the whole file sharing process. As peer-to-peer has become an essential way for people to share and distribute files across the Internet, we believe it should be natively supported mechanism in the future Internet Architecture. The eXpressive Internet Architecture (XIA) is one of three Future Internet Architecture projects funded by US NSF. As a clean-slate network architecture, XIA has some novel features such as evolvability, flexible routing, and in-network cache for content delivery. In this paper, we propose a practical design of BitTorrent protocol implementation on XIA to explore and rethink the peer-to-peer file sharing mechanism in the future Internet architecture.

Keywords Future internet architecture · Peer-to-peer · File-sharing · In-network cache

1 Introduction

Peer-to-peer (P2P) file sharing is one of the most popular ways users acquire and distribute data in today's Internet. Since introduced by Napster in 1999, P2P file sharing mechanism have gained huge popularity among Internet users around the world. With more than 25% of today's total Internet traffic are P2P-related (Price 2011), it has fundamentally changed the way people share and distribute data. BitTorrent (Cohen 2003) is one of the most popular P2P file sharing applications with more than 150 million active users. Since first introduced in Cohen (2003), BitTorrent protocol and applications have been widely used in sharing and distribute content like movies, music and games. The success of BitTorrent is not a coincidence, as it is well designed to provide an effective mechanism to maximize the transfer bandwidth and tackle the free-riding problems. Despite the legal issues, BitTorrent protocol and other P2P file sharing applications have proved to be an efficient and popular way to share and acquire data. Microsoft uses P2P technology in Windows 10 and other software distribution in their content distribution network for downloading large amounts of data without incurring the dramatic costs on the official download server.

However, the BitTorrent protocol is built entirely on the application level. Without the native support of network layer, peers in BitTorrent protocol have to discover other peers' IP address and exchange data through TCP connections. Typically, the BitTorrent application can only handle no more than ten TCP connections with other peers at the same time, which limits the potential of achieving maximum download bandwidth. Furthermore, we argue that the TCP connection might not be the best choice for BitTorrent protocol since the data integrity is checked at application layer and unordered or lost data packets are also not the primary concern in the BitTorrent protocol. Since BitTorrent and other alike peer-to-peer file sharing protocols contributes a large portion of today's Internet traffic. We believe that the design of future Internet architecture should take into

✉ Ziqian Meng
markmzq@pku.edu.cn

Zhong Chen
zhongchen@pku.edu.cn

Zhi Guan
guan@pku.edu.cn

[1] School of Electronics Engineering and Computer Science, Peking University, Beijing, China

[2] National Engineering Research Center of Software Engineering of Peking University, Beijing, China

account a certain level of native support of the P2P file sharing mechanism.

eXpressive Internet Architecture (XIA) (Anand 2011; Han 2012; Naylor et al. 2014) is one of three Future Internet Architecture (FIA) projects funded by US NSF. As a clean-slate network architecture, XIA has some novel features such as evolvability, flexible routing, and the support of in-network caching for content delivery. XIA supports multiple types of communication and protocol as the primary entity of the network. Content chunks can be identified by their hash value and treated as primary entity of the network. It can be routed directly at the network level and cached by the on-path routers. The concept perfectly matches the demand and specification of BitTorrent protocol in many ways. Based on the above observations, we design a native support implementation for BitTorrent protocol in XIA, which is called BitTorrent over XIA, to explore the possible solution for future Internet architecture to support P2P file sharing.

Some research has already focused on the utilization of in-network cache to eliminate redundant P2P traffic and simplify implementation of BitTorrent protocol. However, the efficiency of in-network cache always suffers from low cache hit ratio, which is caused by low-popularity content and small cache size of content routers. In certain cases, low cache hit ratio can cause extreme transmission latency and network overhead. The simulation study shows that our design of BitTorrent over XIA can largely mitigate the low cache hit ratio problem without degradation of network performance.

The rest of this paper is organized as follows. Section 2 introduces technical background related to our research. Section 3 presents the design of BitTorrent over XIA and describes the challenges and potential advantages of the implementation. Section 4 describes our simulation setup as well as the results. Section 5 concludes our work and discusses the problems to be solved in the future.

2 Technical background

2.1 eXpressive Internet Architecture

eXpressive Internet Architecture (XIA) is a novel future Internet architecture with support of multiple communication styles and the ability of introducing new communication entities to achieve network architecture evolution. As the common view in the designs of future Internet architecture, XIA separates identifier and locator to improve scalability and simplify mobility. In terms of network identifier, XIA supports multiple types of network identifier. Different types of communication entities, such as network domain, host, service and data chunk, are coded as different network identifiers in the XIA. Each type of entity is coded as equal-length 160-bit XIA identifier (XID). network domain (NID), host (HID) and service (SID) are coded according to their public keys for intrinsic security purpose. Data chunks are coded by the hash value of the content for integrity verification purpose. Different XIDs are processed by corresponding protocols on the on-path routers. To mitigate the flat network routing problem brought by hash-style identifier, XIA uses NID for global addressing. After sent to the specific network domain, the packets are forwarded to the host by the HID, which specifies the recipient of the packets. SID is used to achieve anycast-style communications when the originators send packets to a set of servers, instead of one single server, to request a particular service. CID can be used to retrieve content directly from network, regardless of the actual location of content storage. Since content can be directed expressed and recognized by the routers, XIA can natively support the in-network cache by the on-path routers of the network packets.

In terms of network locator, XIA uses directed acyclic graph (DAG) representation of multiple XIDs as the network address. The DAG structure of XIA address is high flexible, which allows the address constructor to express its intent of how the network packets with certain address should be processed along the forwarding path. The simplest form of a DAG address can be constructed by only a dummy source • and a HID node as shown in Fig. 1a. The dummy source represents the conceptual source with no specific meaning. In order to be forwarded through multiple network domains in real-world Internet environment, currently the DAG address is constructed as shown in Fig. 1b. Routers check the NID node in DAG and route the packets accordingly before they reach the destination network domain.

Another feature of XIA address is "fallback" path, which allows alternative routing paths expressed in one DAG address. Figure 1c shows a DAG with fallback path, which is

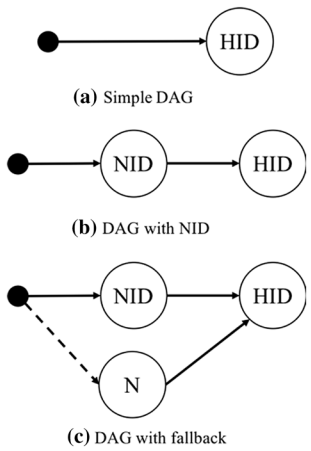


Fig. 1 DAG address structure in XIA

represented as a dotted line pointing to node N. If the direct routing path expressed in DAG is unreachable, routers would try to forward the packet to node N by the fallback path. Supposedly node N knows the routing path to the next node in DAG. The solid lines always have higher priority than dotted ones (fallback path) and the fallback path is optional and by definition the fallback node should always know the routing path of the packets.

2.2 BitTorrent protocol

In the BitTorrent system, a host runs BitTorrent protocol and participates in the file sharing process is called peer. A peer who publishes the file or has the complete set of shared data pieces is called seeder. A peer who has incomplete dataset and downloads data from other peers is called leecher. The collection of seeders and leechers who are participating in the same BitTorrent file sharing process is called a swarm. Each shared file is divided into equal-sized data chunks called pieces. Pieces are the smallest data units in the BitTorrent protocol. It will be further divided into packets in the transport layer and network layer (Tarkoma 2010).

To start a P2P file sharing process. A seeder has to first create a .torrent file, which contains essential metadata of the shared files and a list of network address of trackers. Trackers keep track of seeders in the same swarm. The metadata in the .torrent file contains the file structure of shared files and the cryptographic hash values (usually SHA-1) of each piece for the integrity verification purpose upon its arrival to the leechers. After the creation of .torrent file, the seeder publishes the file to the Internet for other peers to start downloading process.

To start a downloading process, a peer should first acquire the .torrent file from the Internet and connects to trackers listed in the torrent file to register itself as a leecher. Tracker replies the leecher with a list of peers who are currently sharing the files specified in the .torrent file. The leecher establishes the TCP connection with other peers to transmit data pieces. After receiving the data piece, the leecher verifies the integrity of the piece by its hash value stored in the .torrent file. To achieve faster dissemination and wider distribution of the data pieces, BitTorrent protocol implements an algorithm called "Rarest First". A leecher constantly looks for the rarest data piece that is available among the swarm and request the piece from other peers. The Rarest First strategy has been proven to be more effective than the pure random piece selection strategy (Bharambe et al. 2005; Felber and Ernst 2004).

2.3 BitTorrent design challenges

2.3.1 Heavy responsibility at application layer

Since the BitTorrent protocol is an application layer protocol and the currently TCP/IP architecture is host-centric. A leecher has to discover the IP address of other peers before it starts downloading file pieces. In the early days of BitTorrent, trackers were used to track every peer in the swarm. A leecher has to connect to the trackers to acquire the list of other peers. However, the concept of trackers breaks the decentralization philosophy of BitTorrent protocol and can cause single point failures from time to time. In the later extension of the protocol, distributed hash table (DHT) (Loewenstern 2008) and peer exchange protocol (PEX) (Wu et al. 2010) are introduced as decentralized ways for peers to discover others. However, the peer still need to acquire the IP address of bootstrap DHT node in the .torrent files and maintain a list of IP address of other peers.

After the peer discovery process, the leecher needs to establish TCP connections with other peers to acquire data pieces. Since it costs system resource to maintain TCP connections, operating systems usually limit the maximum number of TCP connections one BitTorrent application can have at the same time to no more than ten. This restriction could limit the BitTorrent application's potential capability to achieve higher download speed. Moreover, given that BitTorrent application has no knowledge of the topology and link status at the network layer, the application has to randomly choose a certain number of peers to connect at the beginning. During the downloading process, the leecher need to constantly pick new peers from the swarm and try to establish connection with them in the hope of achieving potentially higher download rate. When the data piece arrives at the leecher, the BitTorrent application has to verify the integrity of the piece by calculating the hash value of received piece and compare it with the value stored in .torrent file.

Due to the lack of native support from network and TCP protocol, BitTorrent protocol has to handle nearly every step of the P2P file sharing process. This heavy responsibility can be inefficient since the application layer has to pay extra effort to obtain and maintain the knowledge of network status.

2.3.2 Incentive mechanism

As the most popular P2P file sharing protocol, the success of BitTorrent is not a coincidence. Since P2P is a decentralized system and data pieces are transferred directly between peers, it inevitably suffers from the free-riding problem (Adar 2000) which caused by peers who just aggressively obtain data from other peers and never share with others. BitTorrent protocol successfully tackles the free-riding problem by introducing an incentive mechanism called "tit-for-tat" (Cohen 2003, 2008). During the file sharing process, a peer can "choke" a free-rider by stop uploading data to it. Tit-for-tat strategy encourages peers to share as much content as possible to achieve maximum download speed,

which guarantees that every peer has to contribute while obtain from others. Since the tit-for-tat mechanism was first introduced in BitTorrent, the performance of incentive mechanism in BitTorrent protocol has been studied theoretically (Choe et al. 2007; Yue et al. 2006; Liao et al. 2013; Qiu and Rayadurgam 2004; Zhang et al. 2015a, c) and practically (Pouwelse et al. 2005; Xia and Jogesh 2010; Guo et al. 2007). Moreover, Locher et al. (2006) shows that BitTorrent does not provide sufficient incentives to rule out free-riders. Lots of research has focused on the improvement of the data sharing incentives in BitTorrent and other P2P file sharing protocol. Propshare (Fan et al. 2009) attempts to improve the original tit-for-tat strategy by unevenly splitting the upload rate in terms of the contribution received from the previous round. BitTyrant (Piatek et al. 2007) tries to determine the exact amount of contribution necessary to maximize its download rate by dynamically adapting and shaping the upload rate allocated to its neighbors. Some incentive mechanisms in other P2P file sharing systems take more indirect and complicated approaches to tackle the free-riding problem. In Sandvine and (2017), encryption of data pieces is implemented to ensure the data peers who upload are reciprocated. In some reputation-based incentive mechanism, peers upload preferentially to peers who contributed most in the history.

2.4 Related works

BitTorrent and other P2P file-sharing protocols contribute to more than half of the upstream traffic and a quarter of downstream traffic on the Internet (Shin et al. 2017). In addition, study from (Yamamoto and Akihiro 2012) shows that more than 70% of BitTorrent packets are forwarded between ASes. Given that a popular file can be shared hundreds of thousands of times across the Internet in a short period of time, a considerably large portion of the inter-AS P2P traffic is redundant. Due to the huge traffic usage and potential advantage of eliminating redundant traffic, the Internet Service Providers (ISPs) and researchers have every motivation to reduce the traffic volume caused by P2P protocols. Many research focuses on utilizing in-network cache or domain-edge cache to eliminate the inter-AS P2P traffic (Yamamoto and Akihiro 2012; Hefeeda et al. 2011; Nakao et al. 2008; Anand et al. 2008). Caching and traffic localization have been proven to be effective ways to eliminate redundant traffic caused by P2P sharing protocols. The main challenge of exploiting in-network cache is to retain high cache hit ratio. Traditionally, the least recently used (LRU) algorithm is widely used as cache replacement algorithm. However, the LRU algorithm just utilizes temporal content locality in the past time that does not reflect content requests in the future (Kondo et al. 2015). Recently, with the fast-pace development of Information-Centric Network (ICN) and Content-Centric Network (CCN), in-network cache utilization has again become a hot research topic (Dehghan et al. 2016). Some researchers focus on developing a more sophisticated cache replacement strategy that aims to improve cache hit ratio (Kondo et al. 2015; Dehghan et al. 2016). Others have been exploring the optimization of in-network cache scheme by introducing coordination between content routers (Ming et al. 2014; Hu et al. 2015).

Other future Internet architecture projects have also conducted research on content delivery (Zhang et al. 2012, 2015b) and native support of in-network cache to eliminate redundant P2P traffic. In Mastorakis et al. (2017), NDN project has already proposed a BitTorrent-like P2P file sharing application, nTorrent. The application bases on the natively data-centric NDN network architecture, which maximizes efficiency of data retrieval by leveraging network layer information directly. Our research in this paper is different from the nTorrent application. The implementation of BitTorrent over XIA focus on leveraging the flexible routing of XIA to guarantee an acceptable download speed in low cache hit scenarios. In the meantime, our implementation can still benefit from the in-network cache whenever and wherever it is available.

3 BitTorrent over XIA

3.1 Potential advantages of BitTorrent over XIA

Some potential performance improvements can be achieved by leveraging the novel features of XIA. Based on the observation, we notice that the hash values of data chunks in torrent file meets perfectly with the concept of content identifier (CID) in XIA, which means the CIDs of data chunks can be directly stored in the xtorrent file and used as destination in DAG address. In XIA, all the content packets with CID can be cached by the on-path routers. This feature can also be leveraged by BitTorrent over XIA. With the native support of in-network cache, BitTorrent and other P2P redundant traffic can be automatically eliminated by the network architecture. Moreover, since the CID is the hash value of the content, the integrity verification of received data chunks can be done at the network level instead of application level, which leads to a more light-weighted BitTorrent application.

Instead of leveraging in-network cache, our implementation mainly focuses on benefiting from the XIA's flexible routing design, which allows multiple routing paths and destinations to be constructed into one DAG address. Content request packets can have multiple destination as backup routing choices when the primary destination is unavailable. We believe this feature is extremely effective in the low cache hit scenarios, which is caused by low-popularity content or small cache size. With the introduction of in-network cache,

most experimental BitTorrent-like applications have abandoned the TCP connections between peers (Zhang et al. 2012, 2015b; Mastorakis et al. 2017). Instead, a UDP-style content request is used to retrieve content from the network. However, in the low cache hit scenarios, a large portion of content requests cannot be satisfied by in-network cache and will be eventually routed to peers. Since peers do not maintain TCP connections with others, they have no information of the network status. Peers may be temporally offline or have already left the swarm. The network links between peers can have very low quality, which leads to unacceptable download speed. With the support of flexible routing, a peer can send request of a specific data chunk to multiple seeders to have a better chance in successfully fetching the data piece to ensure a stable download speed and increase the robustness of the BitTorrent over XIA.

3.2 Challenges of designing BitTorrent over XIA

Several challenges have to be faced in implementing BitTorrent protocol on XIA. Since the currently used BitTorrent protocol and applications are entirely built on the application level. The protocol and applications have to be largely modified to fully utilize the novel features of XIA.

Firstly, the structure of network address in XIA is totally different from the network structure currently used in the Internet, which means the communication functions in BitTorrent applications have to be totally modified to adopt the DAG-style addresses in XIA. In current BitTorrent protocol, a peer has to maintain TCP connections to send requests to specific seeders to acquire certain data chunks. While in XIA, a host can just send a UDP-style CID request packet to the network to fetch the data chunks without the knowledge the location of the content, which is largely different from the current data transfer mechanism in BitTorrent. The most challenging modification is the data sharing incentive of BitTorrent over XIA. Tit-for-tat strategy has been proved to be a successful incentive. However, the tit-for-tat strategy is based on the upload rate of each peers while in XIA a large proportion of content requests can be satisfied by routers with the cache of requested data chunk. Since the contribution of each peer cannot be measured by its upload rate, a new kind of incentive mechanism is needed for BitTorrent over XIA.

3.3 Design of BitTorrent over XIA

3.3.1 .xtorrent file structure

We redesign the .torrent file to meet the network architecture and novel features of XIA. Figure 2 shows the structure of .xtorrent file, which plays the same role as .torrent file in BitTorrent protocol.

As illustrated in Fig. 2, the general structure of .xtorrent file is similar to .torrent file. Trackers' URL in "Announce" field are replaced with trackers' DAG addresses. The "CID List" field contains the content identifiers (CIDs) of all the data pieces to be shared by .xtorrent file.

Before sharing with other peers, a publisher creates the .xtorrent file for to-be-shared files. Generally, the publisher needs to divide shared files into equal-length data pieces, calculate each data piece's hash value and fill it into CID list field. After the creation of .xtorrent file, it will be published on the Internet.

3.3.1.1 Protocol procedure For peers who want to participate in the specific file sharing process, the protocol procedure of BitTorrent over XIA is illustrated in Fig. 3.

1. Firstly, the peer needs to get possession of the .xtorrent file, which can be acquire from a Webserver or other peers.
2. The peer sends handshake message and registers itself to trackers according to the DAG addresses listed in the "Announce" field of .xtorrent file.
3. Tracker replies the peer with a list of active seeders.
4. Peer sends handshake message to seeders in the list. The seeder replies with a list of data pieces it currently has.
5. The peer sends CID request packets to retrieve data pieces from seeders. Before forwarding the packets, on-path routers check if the requested data piece is in their cache, if so, the router would directly reply the CID request with the packets of data piece. The detail of DAG address will be discussed later.
6. After receiving the CID request, seeder replies the request with packets of data pieces. The on-path routers

Field	Description
Announce	Tracker1:AD_1→HID_1 Tracker2:AD_2→AD_3→HID_2 Tracker3:AD_4→HID_3 Tracker4:AD_5→HID_4
Size	xxx bytes
File Names	File1 File2 File3 ……
Piece Length	256KB
CID List	CID_1:E4A2CFA462AD60E53AE5D284A103AA5EA673CD66 CID_2:DB09964C8A70B57110755ED928E8B51962CBFFB2 CID_3:00987C989B0FEC84799FFA62DEBEAECDD9A66DE5 CID_4:7468237B26FE9A86D5F02023916EFAEA44EEC8F8 ……

Fig. 2 File structure of .xtorrent file

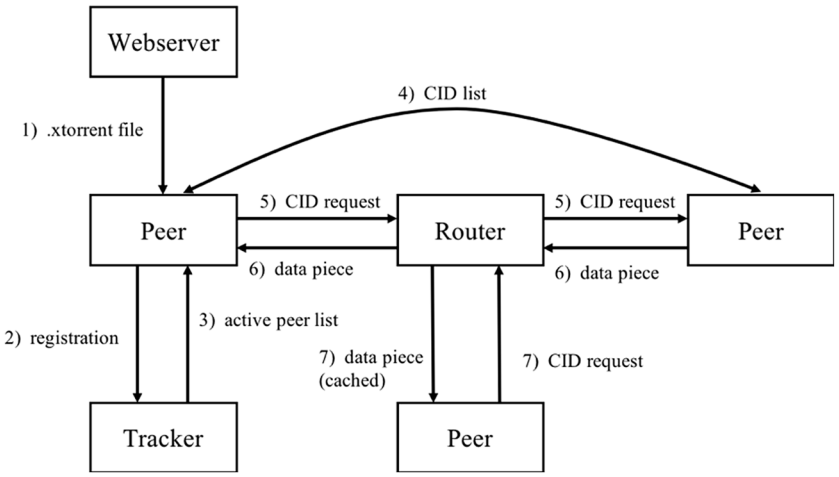


Fig. 3 Protocol procedure of BitTorrent over XIA

of the packets choose to cache the data piece according to their own caching policies.

7. Other peers send the CID request to retrieve the same data piece.
8. On-path routers check the CID in their cache and directly reply the CID request with cached content.

3.3.2 DAG address structure

The possible DAG addresses of CID request packet are shown in Fig. 4. Since the DAG structure provides huge flexibility in terms of network addressing and routing. A peer has several options when constructing the destination address of the CID request packets.

The DAG address in Fig. 4a is the most straight-forward option for leechers. The CID request packets with this DAG will be routed to the seeder at NID → HID for the content of CID. The address shown in Fig. 4b is embedded with multiple fallback paths in case the primary seeder is absent. When the seeder at $NID_1 \rightarrow HID_1$ is not reachable, routers would try to forward the packet to seeders at $NID_2 \rightarrow HID_2$ or $NID_3 \rightarrow HID_3$ for the content identified by CID.

In the network domain where CID routing table is fully implemented on the routers. The leecher can choose to fetch data pieces with DAG shown in Fig. 4c. The DAG address expresses the intent that a leecher wants to fetch the content from anywhere of the network regardless of its storage location. In the network environments where CID routing table is partially supported, a leecher can use the DAG shown in Fig. 4d. When the CID in the DAG cannot be found in the routing table or CID routing is not supported by the router, the packets would be forwarded via fallback paths.

Figure 5 shows the standard process flow within a router when forwarding a CID request packet. Once a content request packet arrives, the router will first check if the content identified by CID is in its cache. If so, the router will

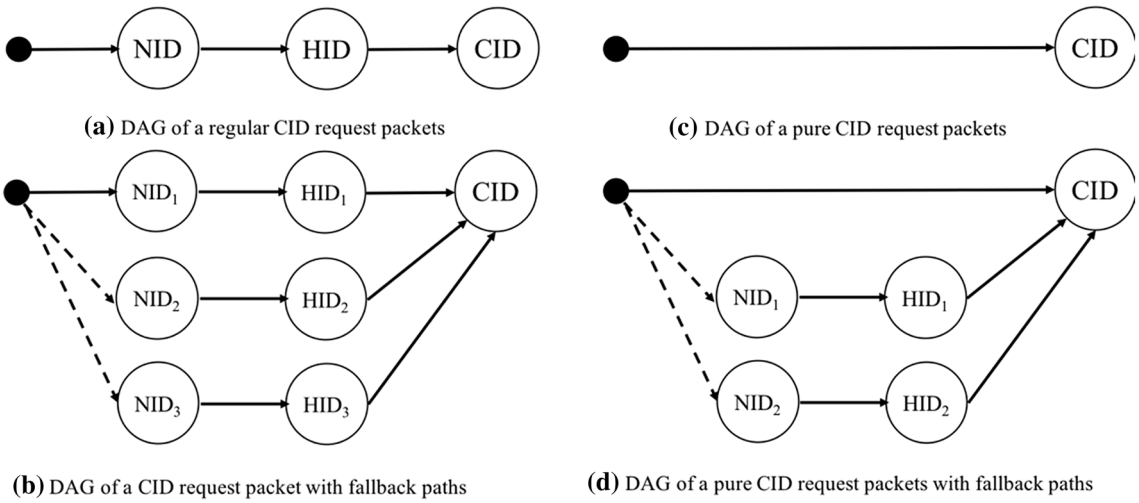


Fig. 4 DAG options for CID request packets

satisfy the content request by directly return data pieces of the content. If the requested content is not in the cache, the router will look up the CID routing table (if available) to find the next hop destination for the CID packet. If the CID routing table is not implemented in the router or there is no routing entry for the CID, the forwarding decision will be made based on the NIDs and fallback paths specified in the DAD address. If a content request packet is still not routable, it will be forwarded to the default gateway router.

4 Simulation study

4.1 Simulation setup

Since we focus on the network performance in low cache hit ratio scenarios, we only study on the successful data retrieval rate of BitTorrent over XIA in the cases where cache hit ratio is fixed. In our simulations, we assume that the .xtorrent file is acquired by all the peers. The shared content consists of 10 files with the size of 100 MB each. The network topology and user behavior are simulated based on the data collected from real BitTorrent users of China Education and Research Network 2 (CERNET2), which is the world's largest native IPv6 backbone network. The topology consists 503 nodes and 1578 links. Peers follows rarest-piece-first strategy and tit-for-tat mechanism is used as data sharing incentive when choose whether or not to share data with particular peers.

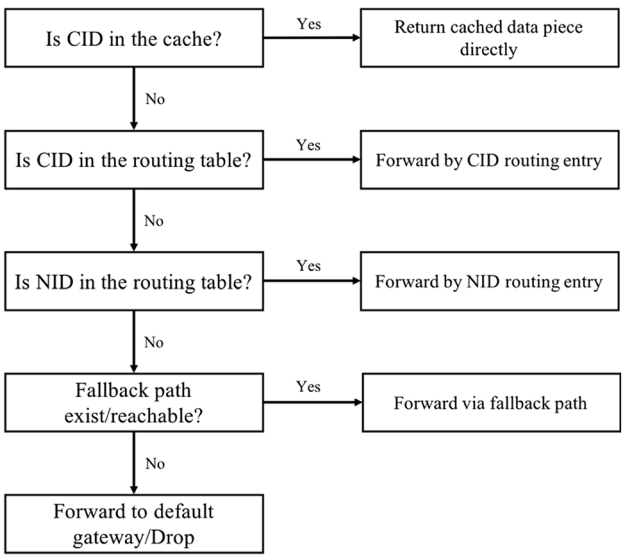
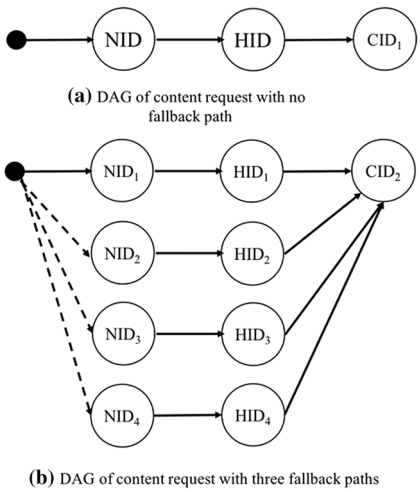


Fig. 5 Standard process flow within routers

4.2 Result and evaluation

We conduct simulations on the successful data retrieval rate with different DAG structures in different cache hit ratio scenarios. There are four types of DAG address in our simulation, the DAG with no fallback path, the DAG with only one fallback path, the DAG with two fallback paths and the DAG with three fallback paths. The examples of DAG address used in the simulation is shown in Fig. 6.

Figure 7 shows the successful data retrieval rate with different DAG structures in fixed cache hit ratio scenarios. As shown in the chart, the content request packets with multiple fallbacks paths gain a huge improvement in terms of data retrieval rate when the cache hit ratio is low. The more fallback paths a packet have, the higher successful data retrieval rate it can achieve. When the cache hit ratio is 10%, the data retrieval performance can be improved by more than 480%. As the hit ratio increases, the performance gap is narrowed. When the hit ratio is more than 80%, the performance of data request with multiple fallback path can only be improved by 7%.



(a) DAG of content request with no fallback path

(b) DAG of content request with three fallback paths

Fig. 6 DAG address used in simulation

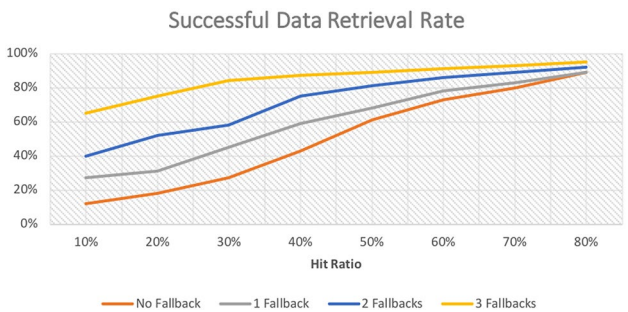


Fig. 7 Successful data retrieval rate

5 Discussion and future work

As P2P file sharing is one of the most essential ways of transferring content in current Internet, it is beyond doubt that it should be natively support by future Internet architecture to achieve better performance and less network overhead. In this paper, we propose a practical implementation of BitTorrent protocol on the XIA, a novel network architecture, as an exploration of possible ways for deploying P2P file sharing protocol on future Internet architecture. We focus on the performance improvement gained by the flexible routing feature of XIA instead of in-network caching scheme and policies. The simulation study shows that our implementation achieves a significant performance improvement in scenarios where in-network cache hit ratio is relatively low.

We encounter several challenges during the process of designing incentive strategies in file sharing. The behavior of content transmission is largely different from current Internet due to the widely existence of in-network cache. Therefore, the P2P file sharing system and incentive strategy need to be carefully designed to fully utilize the novel features. There are still several design problems to be solved in the implementation of BitTorrent over XIA. For instance, XIA uses LRU as its in-network cache replacement algorithm which has a limited efficiency in some scenarios. We believe that a cache replacement algorithm specifically optimized for peer-to-peer scenario is needed to unleash the power of P2P file sharing in XIA and other future Internet architectures. Although the tit-for-tat strategy has been proven to be an effective incentive of file sharing, its performance and necessity in the cache-based network architecture needs to be evaluated by more experiments and simulation study. Moreover, we believe a carefully-designed new coordinative caching scheme is needed to fully utilize the flexible address structure of XIA. We will try to solve these problems in the future work.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

Adar, E., Bernardo, A.H.: Free riding on Gnutella. First monday 5.10 (2000)

Anand, A., et al.: Packet caches on routers: the implications of universal redundant traffic elimination. ACM SIGCOMM Computer Communication Review, vol. 38, no. 4. ACM (2008)

Anand, A., et al.: XIA: an architecture for an evolvable and trustworthy Internet. In: Proceedings of the 10th ACM Workshop on Hot Topics in Networks. ACM (2011)

Bharambe, A.R., Herley, C., Padmanabhan, V.N.: Analyzing and improving bittorrent performance. Microsoft Res. Microsoft Corp. One Microsoft Way Redmond WA **98052**, 2005–2103 (2005)

Choe, Y.R.: Analyzing and improving a bittorrent network's performance mechanisms. ACM MM'07 (2007)

Cohen, B.: Incentives build robustness in BitTorrent. Workshop on Economics of Peer-to-Peer Systems, vol. 6 (2003)

Cohen, B.: The BitTorrent protocol specification, version 11031 (2008)

Dehghan, M., et al.: A utility optimization approach to network cache design. Computer Communications, IEEE INFOCOM 2016—The 35th Annual IEEE International Conference on. IEEE (2016)

Fan, B., John, L., Dah-Ming, C.: The design trade-offs of BitTorrent-like file sharing protocols. IEEE ACM Trans. Netw. **17**(2), 365–376 (2009)

Felber, P., Ernst, W.B.: Self-scaling networks for content distribution. In: Proceedings of International Workshop on Self-* Properties in Complex Information Systems (2004)

Guo, L., et al.: A performance study of BitTorrent-like peer-to-peer systems. IEEE J. Select. Areas Commun. **25**, 1 (2007)

Han, D. et al.: XIA: efficient support for evolvable internetworking. NSDI, vol. 12 (2012)

Hefeeda, M., Hsu, C.-H., Mokhtarian, K.: Design and evaluation of a proxy cache for peer-to-peer traffic. IEEE Trans. Comput. **60**(7), 964–977 (2011)

Hu, X., et al.: Enhancing in-network caching by coupling cache placement, replacement and location. Communications (ICC), 2015 IEEE International Conference on. IEEE (2015)

Kondo, D., HyunYong, L., Akihiro, N.: Content piece rarity aware in-network caching for BitTorrent. Global Communications Conference (GLOBECOM), 2015 IEEE. IEEE (2015)

Liao, W.-C., et al.: Modeling BitTorrent-like systems with many classes of users. ACM Trans. Model. Comput. Simul **23**(2), 13 (2013)

Locher, T., et al.: Free riding in BitTorrent is cheap. In: Proceedings of Workshop on Hot Topics in Networks (HotNets) (2006)

Loewenstern, A., Arvid N.: BEP 5: DHT protocol. Last modified on Feb 28 (2008)

Mastorakis, S., et al.: nTorrent: peer-to-peer file sharing in named data networking. Computer Communication and Networks (ICCCN), 2017 26th International Conference on. IEEE (2017)

Ming, Z., Mingwei, X., Dan, W.: Age-based cooperative caching in information-centric networking. Computer Communication and Networks (ICCCN), 2014 23rd International Conference on. IEEE (2014)

Nakao, A., Sasaki, K., Yamamoto, S.: A remedy for network operators against increasing P2P traffic: enabling packet cache for P2P applications. IEICE Trans. Commun. **91**(12), 3810–3820 (2008)

Naylor, D., et al.: XIA: architecting a more trustworthy and evolvable internet. ACM SIGCOMM Comput. Commun. Rev. **44**(3), 50–57 (2014)

Piatek, M., et al.: Do incentives build robustness in BitTorrent. In: Proceedings of NSDI, vol. 7 (2007)

Pouwelse, J., et al.: The bittorrent p2p file-sharing system: measurements and analysis. IPTPS, vol. 5 (2005)

Price, D.: An estimate of infringing use of the internet. Disponıvel em. http://documents.envisional.com/docs/Envisional-Internet-Usage-Jan2011.pdf (2011)

Qiu, D., Rayadurgam, S.: Modeling and performance analysis of BitTorrent-like peer-to-peer networks. ACM SIGCOMM computer communication review, vol. 34. no. 4. ACM (2004)

Sandvine, Inc ULC. Global Internet Phenomena Report (2017)

Shin, K., et al.: T-chain: a general incentive scheme for cooperative computing. IEEE ACM Trans. Netw. (2017)

Tarkoma, S.: Overlay Networks: Toward Information Networking. CRC Press, Boca Raton (2010)
Wu, D., et al.: Understanding peer exchange in bittorrent systems. In: Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on. IEEE (2010)
Xia, R.L., Jogesh, K.M.: A survey of bittorrent performance. IEEE Commun. Surv. Tutor. **12**(2), 140–158 (2010)
Yamamoto, S., Akihiro, N.: P2P packet cache router for network-wide traffic redundancy elimination. Computing, Networking and Communications (ICNC), 2012 International Conference on. IEEE (2012)
Yue, Y., Lin, C., Tan, Z.: Analyzing the performance and fairness of BitTorrent-like networks using a general fluid model. Comput. Commun. **29**(18), 3946–3956 (2006)
Zhang, F., et al.: Content delivery in the mobilityfirst future internet architecture. Sarnoff Symposium (SARNOFF), 2012 35th IEEE. IEEE (2012)
Zhang, F., et al.: EdgeBuffer: caching and prefetching content at the edge in the MobilityFirst future Internet architecture. World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2015 IEEE 16th International Symposium on. IEEE (2015a)
Zhang, X., et al.: A distributed in-network caching scheme for P2P-like content chunk delivery. Comput. Netw. **91**, 577–592 (2015b)
Zhang, M., Luo, H., Zhang, H.: A survey of caching mechanisms in information-centric networking. IEEE Commun. Surv. Tutor. **17**(3), 1473–1499 (2015c)

Ziqian Meng Ph.D. candidate of EECS at Peking University. He obtained bachelor's degree at Computer Science at Peking University in 2013. He was a visiting scholar in CMU from 2015 to 2016. His research interests include future internet architecture, network security and blockchain.

Zhong Chen Ph.D, Professor of School of EECS at Peking University, and Director of MoE Key Lab of Network and Software Assurance, Director of Financial Information Research Center of Peking University. He has awarded Beijing Excellent Teacher Award in1996, 1st Prize National Higher Education Teaching Achievements Award in 2005, 2nd Prize National Science and Technology Award in 2010. His research interests include domain-specific software engineering, network and information security, blockchain.

Dr. Zhi Guan is an associate professor of MoE (Ministry of Education) Key Laboratory of Network and Software Security Assurance, Peking University and National Engineering Research Center of Software Engineering of Peking University since 2009. He gives lectures on Information Security, Cryptography and Network Security. He is the creator of the GmSSL open source cryptography library project. His current research interests include cryptography engineering and security of blockchain and crypto-currency.