Extending Mobile BitTorrent Environment with Network Coding

Péter Ekler
Department of Automation and Applied Informatics
Budapest University of Technology and Economics
peter.ekler@aut.bme.hu

Tamás Lukovszki
Faculty of Informatics
Eötvös Loránd University
lukovszki@inf.elte.hu

Abstract- Peer-to-peer based content sharing is considered as one of the most efficient and popular content distribution solution nowadays. However in such network it is common that parts of the content is not distributed to several peers yet, thus during the download we have to wait a lot for that rare pieces. In addition to that it is also possible that we will not be able to download the rare pieces because the original seeder has already left the network. In this paper we investigate how to increase download efficiency in BitTorrent, in case of rare pieces by using network coding methods. In mobile networks the peer connections are not so reliable, because the phones as peers can leave the network any time e.g. if the network coverage is weak. Bringing BitTorrent technology to mobile phones has already been demonstrated with the implementation of SymTorrent and MobTorrent. In this research we investigate how to extend these implementations with network coding, to increase efficiency in case of rare pieces.

Keywords: BitTorrent, mobile phones, network coding

I. INTRODUCTION

BitTorrent [1] is considered as one of the most popular and efficient peer-to-peer (P2P) content sharing solution. In BitTorrent the data is distributed in small pieces on a non-deterministic order. The main advantage of the technology is that peers are able to share the downloaded pieces before they have downloaded the whole content.

Nowadays the capability and storage capacity of mobile phones and PDAs allow them to reach the Internet quite easily. Bringing content distribution technologies to mobile devices is an interesting and open research area. The first steps towards this goal have already been taken with the implementation of Symella [2], SymTorrent [3] for smartphones. In addition to that MobTorrent [4] have shown that even mainstream mobile phones can participate in large P2P networks.

The BitTorrent protocol ensures high availability using the rarest-first policy for the distribution of pieces of a file. Rare pieces occur in the system when some pieces are available only on a few peers. In this case downloading these rare pieces is a key step in order to download the whole content, because if for some reason the owner peers of these pieces leave the network, we will not be able to download the content. Measurements have shown that peers rarely connect for more than a few hours, and frequently for only a few minutes [5].

From another perspective, if we connect to the BitTorrent network from mobile clients (e.g. by using MobTorrent) it happens quite often that we cannot connect to peers because of some firewall or NAT issues. In this situation if we cannot connect to some peers who have important pieces, the same rare piece problem occurs.

In this paper we propose a network coding [6] based extension for BitTorrent which increase the efficiency in case of the rare piece problem. The proposed extension can be applied in the SymTorrent and MobTorrent mobile BitTorrent clients.

II. BITTORRENT PROTOCOL

In BitTorrent [1] protocol first we need a very small torrent file. This file contains one or more tracker address(es) (HTTP URLs). After the torrent file is available, the BitTorrent client connects to the tracker(s) which sends back the IP addresses of the relevant peers. If the torrent is not so popular the number of retrieved peers can be very small. When the list of peers is available the client starts to connect to them and the real content download goes via the peer wire protocol between the client and the peers. Of course the other peers are also having a BitTorrent client up and running.

In BitTorrent the content is distributed in small pieces (usually 64 KB). Each piece has an ID (index). During the peer wire protocol there is a piece request and piece send message for piece exchanging. The piece request and send message looks as follows:

Piece request:

len=0013	*id=6*	*index*	*begin*	*length*

Piece send:

len=0009+X	*id=7*	*index*	*begin*	*block*

The first part contains the length of the message, after there is the unique message ID. Each peer wire message has an own ID. 6 is the ID of the request and 7 is the ID of the send piece message. The message ID is followed by the index of the piece, which determines a specific part in the file. The pieces are transferred in small blocks between peers, e.g., if the piece size is 64 kB, then the block size is usually 16 kB. In the request message, the piece index is followed by the beginning position of the block (inside the piece) and the length of the block. Similarly in the piece message the piece index is followed by the beginning position of the block and the block data (e.g. 16 kB large).

III. EXTENDED BITTORRENT PROTOCOL WITH NETWORK CODING SUPPORT

In this section we propose a change in peer wire protocol to enable network coding but still keep compatibility. We refer to the modified protocol as *blockcoding*.

5.1. Blockcoding extension

The modifications affect only the piece request and the send messages. Fortunately we do not need to change the structure of the request message, only the meaning of the *begin* parameter. In the blockcoding protocol the *begin* parameter of the request message means that we request a linear combination of the piece blocks but the combination should contain the block with the specified begin parameter. For this we consider the blocks of the pieces as vectors over the finite field $\mathbf{F}_2$. Then the addition of two blocks corresponds to the bitwise *xor* operation of these blocks. Since, for all block B, B *xor* $B = 0$, we just need 0-1 vectors for describing these linear combinations. Let B_1 and B_2 be two blocks corresponding to linear combinations of the original blocks described by vectors v_1 and v_2. Then B_1 *xor* B_2 corrsponds to the linear combination of the original blocks described by v_1 xor v_2. The structure of the piece send message in blockcoding looks as follows:

len=0009+X	*id=7*	*index*	*begin*	*blockcombination*	*block*

The message has an additional *blockcombination* part, which is basically a 0-1 vector, which indicates which blocks are combined in the *block* part. The number of elements in the blockcombination vector is equal to the number of the blocks in the piece.

In order to keep the compatibility the peers somehow have to decide whether to exchange the pieces on the original way or by following the blockcoding protocol. BitTorrent [1] contains a handshake message between the peers before starting the real piece exchange. This handshake message has a reserved 8 byte which can be used to sign protocol versions. For the blockcoding support we use the 00000011 signal bytes. If both of the clients send the same pattern in the reserved 8 bytes during the handshake, than blockcoding protocol can be used between them.

5.2. Retrieving original block from the linear combination

After the block combinations have arrived only one task remain, restoring the original blocks. In order to do so, we can use the blockcombination vectors received in each piece message to create a matrix. After that we can use the Gauss elimination to calculate the diagonal matrix. Finally we have to execute the same combination steps on the combined data blocks parallel, during the Gauss elimination.

Finally one question remains. How should we combine the blocks when we got a message that the other peer requests a block combination which contains the block started from a specific position? Our first experiments show that random combination strategy [7] seems to be promising. The following proposition shows that in our block coding scheme it is not enough to use only block pair combinations, i.e. paircoding [8] does not work in our case.

Proposition 1. In the blockcoding protocol if the request targets a specific block, than by selecting always only one other block for the combination will not be enough to restore the original blocks even after all of the combinations are sent.

Proof: In a diagonal matrix each vector contains exactly one none zero element (in our case exactly one '1' element). During the Gauss elimination we add one vector to another by using the *xor* operation. Consider *v1* and *v2* vectors which contain even number of '1' elements (in our case 2). After the *v1 xor v2* operation the number of '1' elements in the result vector will change with even amount of '1' elements. This way it is not possible to create a diagonal matrix. □

5.3 Increasing the efficiency in BitTorrent with blockcoding in case of rare pieces

By applying blockcoding we can increase the reliability of content distribution in case of rare pieces. It is possible that peers have only part of the rare piece and some blocks are missing. In case of blockcoding the request message targets not only to a specific block but to any combination which has the requested block. In this way the probability of getting the block is higher.

Determining the index of rare pieces is possible with the BitField and have messages [1].

IV. CONCLUSIONS AND FUTURE WORK

With SymTorrent [3] and MobTorrent [4] applications nowadays BitTorrent is available on smartphones and mainstream mobile phones as well. We have proposed a network coding based extension to the BitTorrent called *blockcoding* which attenuates the problem of rare pieces. Future work includes a full blockcoding implementation in MobTorrent and detailed measurements related to traffic overhead and efficiency.

Acknowledgement: This work is supported by the New Hungary Development Plan (Project ID: TÁMOP-4.2.1/B-09/1/KMR-2010-0002 and TAMOP 4.2.1/B-09/1/KMR-2010-0003).

REFERENCES

[1] BitTorrent specification, Jun. 03, 2010. [Online]. Available: http://wiki.theory.org/BitTorrentSpecification

[2] I. Kelényi, B. Forstner, B. Molnár, "Symella 1.41", *Budapest University of Technology and Economics*. Jun. 03, 2010. [Online]. Available: http://amorg.aut.bme.hu/projects/symella

[3] I. Kelényi, P. Ekler, Zs. Pszota, "SymTorrent 1.41", *Budapest University of Technology and Economics*. Jun. 03, 2010. [Online]. Available: http://amorg.aut.bme.hu/projects/symtorrent

[4] P. Ekler, Á. Ludányi, "MobTorrent 1.10", *Budapest University of Technology and Economics*. Jun. 03, 2010. [Online]. Available: http://amorg.aut.bme.hu/projects/mobtorrent

[5] P. Maymounkov, D. Mazieres: „Kademlia: A peer-to-peer information system based on the xor metric". In *Proceedings of IPTPS02*, Cambridge, USA, Mar. 2002.

[6] R. Ahlswede,; N. Cai, S.-Y. R. Li, and R. W. Yeung: *Network Information Flow. IEEE Transactions on Information Theory*, IT-46 46: 1204–1216, 2000

[7] T. Ho, R. Koetter, M. Medard, D. R. Karger and M. Effros: The Benefits of Coding over Routing in a Randomized Setting. In: *IEEE International Symposium on Information Theory*, 2003.

[8] Ch. Ortolf, Ch. Schindelhauer, A. Vater: Classifying Peer-to-Peer Networking Coding Schemes. In: *Proc. 21st ACM Symposium on Parallelism in Algorithms and Architectures, SPAA*, 2009.