Rate Allocation Algorithm with Successive Refinement in Peer-to-Peer Networks

Yang Su\textsuperscript{1,a} and Mi Lu\textsuperscript{2,b}

\textsuperscript{1}Electrical & Computer Department, Texas A&M University, TX, USA,
\textsuperscript{2}Electrical & Computer Department, Texas A&M University, TX, USA
\textsuperscript{a}suy@tamu.edu, \textsuperscript{b}mlu@ece.tamu.edu

**Keywords:** Multiple Description Codes, Network codes, rate allocation algorithm, and video transmission

**Abstract.** We introduce a new across-peer rate allocation algorithm with successive refinement to improve the video transmission performance in P2P networks, based on the combination of multiple description coding and network coding. Successive refinement is implemented through layered multiple description codes. The algorithm is developed to maximize the expected video quality at the receivers by partitioning video bitstream into different descriptions depending on different bandwidth conditions of each peer. Adaptive rate partition adjustment is applied to ensure the real reflection of the packet drop rate in the network. Also the granularity is changed to the scale of atomic blocks instead of stream rates in prior works. Through simulation results we show that the algorithm outperforms prior algorithms in terms of video playback quality at the peer ends, and helps the system more adjustable to the peer dynamics.

**Introduction**

With an increasing demand for more reliable and faster video transmission through the networks, the decentralized systems such as large content distribution, are becoming more popular among the service providers. Although there is a trend that the traditional peer-to-peer networks are being replaced by other distributed systems such as cloud systems, peer-to-peer networks still serve as a very efficient way to stream live videos. However, some other issues such as instability and isolation come to emerge as well. In order to cope with the instability issue in the distributed networks, the multiple description codes [1] can replace the conventional retransmission scheme, which can result in heavy network congestion, to enhance the robustness of transmission. Another concern in transmitting packets through data networks is efficiency, which can be measured in the throughput of the network. Network coding can be adopted to improve throughput albeit increasing the computation resource and complexity [2].

**P2P Video Stream Framework**

**Assumptions.** P2P networks can be implemented with two strategies: push methods, and pull methods [7]. Push methods construct multicast trees to connect all the peers, while pull methods build a directed graph to interconnect all the users to exchange the availability bitmaps periodically. Each of them has both advantages and disadvantages. Push methods can guarantee faster transmission speeds, but they are very vulnerable to sudden departure of active users. Pull methods are more impervious to peer dynamics, and easier for implementation, but can incur more latency during transmission because of updating the segment availability bitmaps. Our system will adopt a scheme based on the combination of two methods mentioned above: push methods based on directed graph structures. Originally, push methods were used in the tree structures, while pull methods were used in the directed graph structures. The system pushes the packets from the upstream nodes to downstream nodes, after obtaining the accurate information on the missing segments of all the downstream peers.
Algorithm without network coding. The client sends a request that includes both rate constraint information and the requested video title to the server, and the server checks which sources possess the relevant video information, and decide which nodes to send packets to the client. Considering the rate constraint, the server also decides the total streaming rate in which the video is transmitted through the network.

We let \( p_{ij} \) denote the possibility of receiving \( j \) packets out of \( M \) packets for the client numbered \( i \). We here distinguish all \( T \) clients for the reason that different clients might have different network conditions. Therefore, we define the average distortion \( E(D) \) as

\[
E(D) = \frac{1}{T} \sum_{i=1}^{T} \sum_{j=1}^{M} p_{ij} D(R_{ij})
\]

(1)

From the observation of Eq(1), we can see that the formula resembles the expected distortion function in [8]. The rate allocation problem can be solved with typical convex approximation approach. Obviously, this problem is also subject to several constraints, which are:

\[
R_1 \leq R_2 \cdots \leq R_{M-1} \leq R_M
\]

(2)

\[
\sum_{i=1}^{M} \frac{M}{i(i+1)} R_i \leq \sum_{j=1}^{S} R_j^*
\]

(3)

where \( R_j^* \) represents the total outgoing bandwidth of the source \( j \). Because of the P2P system property that video data can be retrieved from different nodes, Eq(3) can be considered as an extension to the total rate budget constraint in [8]. All the outgoing rates from different sources still sum up to:

\[
R_{\text{total}} = MR_1 + \frac{M}{2} (R_2 - R_1) + \cdots + \frac{M}{M} (R_M - R_{M-1})
\]

(4)

As a result, we can follow the same procedure stated in [8] to get optimal rate allocation solution using Lagrange Multipliers method. The problem that remains to be solved is how to determine the rate allocation among different upstream peers. Because each upstream peer might have different bandwidth conditions, we have to monitor the network condition for each peer, so that we can optimize the rate allocation across all of them. We apply the round-robin polling method to inquire about the updated bandwidth condition for each upstream peer, collect all the information, and decide the overall rate allocation. Next, we assign different rate tasks to each upstream peer based on the collected bandwidth conditions. The complete algorithm is shown the following algorithm:

```
while not at end of the video do
    Server X sends polling requests to each node in the source set \{1, 2, ..., S\};
    Each node \( S_i \) in the source set responds to the server with the feedback of the current link capacity \( C_i \);
    Server X sums up all the link capacity \( C = C_1 + C_2 + \cdots + C_S \);
    Server X maintains the max-heap \( H_{\text{max}} \);
    Server X performs the overall rate allocation base on the rate-distortion function;
    Set the rate pivot \( \tau_{\text{curr}} = 0 \);
    for each node \( i \) in \( H_{\text{max}} \) do
        assign the rate task \( \tau_{\text{aug}} = \min(C_i, R_{\text{max}} - \tau_{\text{curr}}) \) to node \( i \);
        update \( \tau_{\text{curr}} \) to \( \tau_{\text{curr}} = \tau_{\text{curr}} + \tau_{\text{aug}} \);
        if \( \tau_{\text{curr}} \geq R_{\text{max}} \) then
            \( \tau_{\text{curr}} = 0 \);
        end
    end
end
```

Before we introduce the network codes into the system, there are still several other issues to be resolved. In the algorithm mentioned above, we idealize the situation that each peer bandwidth
perfectly matches the assigned task rate. Meanwhile, in order to be compatible with network codes, we have to decide the fundamental coded block, which is the indivisible unit used to perform network codes. In our context, we name them the atomic block. In order to tackle with those issues, we have to refine the rate-partitioning scheme. Previously, we perform successive refinement with additive increase on each description. Instead, we perform successive refinement with exponential increase. In this case, the second layer remains divided to two equal parts, while the third layer is divided to four equal parts, instead of three parts. If we have M partitions in total, the last layer will be divided into $2^{M-1}$ parts. Each part in the last layer can be considered to be the atomic block. Another significant improvement with exponential-increase refinement is that the complexity of Reed-Solomon codes [6] is reduced, because only repetition codes are needed to generate replicas of the original parts for the fact that the number of divided parts in each layer is the power of 2. By incorporating this new refinement method, we revise the previous across-peer rate allocation algorithm to the refined algorithm based on the atomic blocks.

**Algorithm with network coding.** If we introduce the network coding to the P2P system, the rate allocation strategy should be adjusted in order to apply network coding to the system. In the prior work [3], random network coding is used to generate M linear combinations of k linearly independent packets as $y_j = \sum_{i=1}^{M} f_{ij} x_i$ where $f_{ij}$ coefficients are chosen from the Galois Field $GF(2^q)$. Fixing up the appropriate size of the Galois field can minimize the probability of obtaining linearly dependent combinations at the clients. As mentioned in the last section, we use the atomic block to perform network codes. However, when it applies to the lossy networks, loss of packets can aggravate the problem, increasing the failure rate that the client is unable to get enough linearly independent combinations. To alleviate the situation caused by network congestion and link failures, we use hierarchical network codes instead. The typical scenario where HNC is applied is that the source has a scalable video encoder and can produce a base layer and several enhancement layers [4]. To be applicable in our situation, the layers are labeled from 1 to M, with 1 being the most important and M being the least important. As a result, data from the 1st layer can be recovered with the highest probability at the client while the data from the $M^{th}$ layer can be successfully obtained with the lowest probability [5].

Then we can apply the hierarchical network coding to the encoded packets. Let $c_j$ represent the number of atomic blocks in each description, and we can generate $cM$ packets with randomly generated coefficients:

$$
\begin{align*}
N_1 &= f_1^1 x_1 + \cdots + f_1^c x_c \\
N_2 &= f_2^1 x_1 + \cdots + f_2^c x_c + \cdots + f_2^c x_c \\
&\vdots \\
N_M &= f_M^1 x_1 + \cdots + f_M^c x_c + \cdots + f_M^c x_c + \cdots + f_M^c x_c 
\end{align*}
$$

(5)

where $f_{ij}$ coefficients are randomly chosen from the non-zero elements of $GF(2^q)$. As a result of this structure, packets in the first layer have larger possibility to be recovered than those in other layers, because we are able to decode data of the first layer by receiving $c_1$ linearly independent packets of $N_1$ type, while we are only able to decode data of the second layer by receiving at least $c_1 + c_2$ linearly independent packets of either $N_1$ or $N_2$ type.

If we assume that the rate of the failure caused by random network coding is so small that we can ignore this type of failure, we can bring up the following rate allocation strategy adjusted for network coding scenario. If we still define $p_{ij}$ as the possibility that the client numbered $i$ receives $j$ out of $M$ packets and assume that every packet has the same probability to fail to reach the client, the coefficients before the rate-distortion function $D(R_j)$ should be changed from $p_{ij}$ to $\hat{p}_{ij}$ by
adaptive probability profile adjustment. The reason why we have to adjust the possibility is that data in each layer have been encoded into \((c_{j+1} - c_j)\) packets, and only when we receive the correct combination of those packets can we decode into the original data, otherwise it is still considered to be failure of decoding. Because of the property of Reed-Solomon, we can recover the \(i\) equal parts out of \(n\) chunks using parameter \((n,l,n-l+1)\).

The rest of procedure is the same as the previous algorithm without network coding. We still define the average distortion function as:

\[
E(D) = \frac{1}{T} \sum_{i=1}^{T} \sum_{j=1}^{M_i} \hat{p}_{ij} D(r_j)
\]

(6)

which is the same as Eq(1). The constraints are the same as the previous situation as well, and the bandwidth constraints for intermediate nodes are loosened since network coding can help the network take more packet loads, and this benefit is difficult to be characterized as a constraint to this problem, and yet should be not ignored. Then we apply the convex optimization algorithm to find out the optimal rate allocation to achieve that lowest distortion in the same way.

Conclusions and acknowledgements

In this paper, we have proposed a new across-peer rate allocation algorithm with successive refinement to improve video transmission in the peer-to-peer networks. The change of granularity from stream rates to the size of atomic blocks helps better reflect the real network condition. Because of the fact that peer-to-peer structures share a lot of similarities with other distributed systems, we will continue to extend this work to other types of distributed systems, such as cloud computing.

References


Rate Allocation Algorithm with Successive Refinement in Peer-to-Peer Networks
10.4028/www.scientific.net/AMM.496-500.2200
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.