Expansion of Region for Redundant Traffic Reduction with Packet Cache of Network Node

Shunsuke Furuta, Michihiro Okamoto, Kenji Ichijo, and Akiko Narita
Graduate school of Science and Technology, Hirosaki University, Hirosaki, Aomori, Japan

Abstract - In recent computer networks, a large amount of data of the same contents are transferred repeatedly. They are often delivered concurrently, as represented by live broadcast. We have developed network nodes with packet caches in order to reduce such redundant traffic in TCP/IP network. We call the node TR node. We have obtained successful reduction rates experimentally in limited network topology in our studies. In this paper, we propose improved TR node with which region of redundant traffic reduction is extended. Improvements consist of two modifications. One is to quit renting a space in the IP header for a cache control argument and take the argument out to the header allowing fragmentation of a packet. The other is cache synchronization generating a synchronization packet. We show results of implementation of the proposed modifications of the TR node and advantages using them.

Keywords: traffic reduction; network node; packet cache; cache synchronization; fragment

1 Introduction

In recent computer networks, the same contents are often transferred repeatedly. If the contents are transmitted through the same route, transmission is wasteful spending of resources for communications. It is desirable to reduce redundancy from the network traffic to utilize limited resources for computer networks efficiently.

There are two dominant methods to reduce the redundancy from network traffic. One is multicast, and the other is caching. Multicast is a method to eliminate redundancy in traffic to let a packet own multiple destinations substantially. It is appropriate for concurrent transmission of redundant data. IP multicast, with which routers at branch points duplicate datagrams, is considered the most powerful technique. However, some requirements hinder utilizing IP multicast. All equipment on transmission route must be acceptable for IP multicast. Furthermore, only datagram type is acceptable for a transport layer protocol. Meanwhile, application level multicast is relived from such limitations since end hosts manage transmission iterations. At the same time, efficiency of this method is lower than that of IP multicast. The same data are passed through the same route unlike IP multicast in principle. Efficiency of it depends on quality of multicast tree with the hosts and overhead for constructing the trees is not negligible. Caching is a method to shift traffic redundancy to access to storage devices. Caching has been employed for elimination of redundancy produced by repeated request for the same content. Proxy server that sends data instead of the original sender is popular. A proxy server that stands nearer than the original server does shortens length of transmission route and number of network segments containing redundant traffic decreases. Caching device of the proxy server is usually secondary storage so that a file is unit of caching. Hence, it is difficult to reduce redundant traffic with concurrency such as live broadcast or video conference. Packet caching using primary storage is another type of caching that enable to reduce concurrent redundant traffic.

We have developed the network nodes for reducing redundant traffic with packet cache [1]-[6]. We call it TR (traffic reduction) node. Two or more TR nodes cooperatively reduce traffic in a service region. Advantage of the TR node over multicast is transparency for endpoint hosts. There have been several problems to resolve in basic methodology of the TR node in [1]. Speed up of processing in [2] [3] is important to gain enough throughput with inexpensive resources. Methods for efficient memory utilization in [4] [5] also contribute to establish system with the TR nodes in reasonable cost. Another important problem is expansion of the service region. In the original TR node network, an encoding TR node and a decoding TR node must be placed alternately. We introduced forwarding TR node in [6] to decrease number of TR nodes and lower cost in service region consisted with multiple network segments. To enlarge the service region drastically, two modifications have been necessary. One is to replace a pure forwarding TR node with ordinary router, and the other is to adapt the TR node to branch topology of computer networks. It is complicated to adapt branch topology completely so that we focused on the situation with a sender and multiple receivers. In this paper, we propose methods to realize them and show benefit.

2 Basic Method for traffic reduction

We outline operation of the TR node for eliminating redundancy in traffic with Fig.1 and Fig. 2. The TR node is a functional router. We have implemented its functions with programs that work on Linux operating system. We designed the TR node for TCP/IP environment and implemented it assuming Ethernet for link layer protocol to perform experiments easily. There are three roles in traffic reduction
with TR nodes. They are encoding, decoding, and forwarding. In Fig. 1, there are two TR nodes. TR node E is situated on the upstream side and receives the same data repeatedly in a short time. It works as an encoder. When it receives a packet, it searches the same data in its packet cache as transported by the received packet. If it is the first time for TR node E to receive data A, the node cannot find it in the cache. TR node E records the data in the cache and transmits data A to TR node D with an instruction code of requesting for recording the data. The instruction code is put in IP header. When TR node E receives data A for the second time, it can find the data in its cache unless the data has been already removed. TR node E encodes data A and sends it with an instruction code to request decoding. Fig. 2 shows format of encoded data. Encoded data is smaller than received one. In general, TCP does not always divide data streams at the same byte offset into segments. Therefore, the data received by TR node E may partially match with one of the records in the cache. The node often represents the data using several cache records or portion of raw data in this case. TR node E sends original data if result of encoding has longer size than received one. TR node D lay on the downstream side works as a decoder. TR node D reconstructs data A referring its cache. If any TR nodes are placed between TR node E and TR node D, they merely forward the received data.

We define the reduction rate $R_{s_j}$ in a network segment $j$ as follows.

$$R_{s_j} = \frac{D_j - C_j}{D_j} \quad (1),$$

where $D_j$ is transmission rate in the case that the TR nodes do not reduce redundant traffic at all, and $C_j$ is transmission rate obtained with the TR node operations for redundant traffic reduction. We define the reduction rate of target network $R_n$ as weighted average with $D_j$ over traffic reduction region as follows.

$$R_n = \frac{\sum_j R_{s_j} \cdot D_j}{\sum_j D_j} \quad (2).$$

We can obtain ideal reduction rate when all redundant streams are divided at the same offset and transported with the largest frames having the shortest headers. In this case, the encoding TR node builds the shortest packets from the largest data. The encoded packet contains only one block of type 0. Packets with block type 1 or block type 0 containing multiple blocks decline traffic reduction rate. Under TCP/IP/Ethernet environment, a packet with 14-byte header, 8-byte preamble and 4-byte FCS for Ethernet, 20-byte IP header, and 20-byte TCP header has the largest data. The ideal value of $R_{s_j}$ is $1440(M-1)/(1526M) = 0.94(M-1)/M$ with number of redundant streams $M$ if we set 4 bytes for each of the sizes of the fields of number of blocks, block type, record number, offset, and length in Fig. 2.
3 Expansion of service region

3.1 Restriction of service region with the previous TR node

There were confinements of service region with the previous TR node as shown in Fig. 3. Arrows in the figure mean routes of TCP originated packets. We had to arrange TR nodes continuously without branch, as in case (a) provided that number of the forwarding TR node may be 0 or more than 1. There were roughly two reasons for restriction of service region of the TR nodes. One was illegal change of IP header and the other was flaw in method for cache synchronization.

The previous TR node used total length field in the IP header for record number in the packet cache. While network nodes could know packet length seeing frame size and header length, usage of the total length field by the TR node was illegal for IP. If an ordinary router was situated in the TR node network like case (b), the router discarded an encoded packet containing illegal values in the total length field. If we would reduce redundant traffic at both upstream side and downstream side of the ordinary router, we had to set pairs of an encoding TR node and a decoding TR node on both sides.

Flaw in the method for cache synchronization caused trouble with such a TR node network that involved branches in a transmission route as shown in case (c) and (d). In case (c), branches of transmission route shoot from a TR node toward downstream side. The previous TR node did not guarantee cache coherency if branch point existed. The packet with instruction for recording for cache synchronization was passed only one route of the branches. If the fastest data of redundant streams was sent to the direction to X, the data was recorded in the caches of TR node A and B. When successive data of the same content was sent to TR node C, TR node A encoded it, and TR node B forwarded it to TR node C despite TR node C did not have the original data. TR node C constructed data using information in the received packet and different source in its cache. Then an incorrect packet arrived at a receiver. This trouble was not brought if only one route of transmission was used for each group of redundant streams. Avoiding the trouble, TR node B sent encoded packets only one direction and decoded packets to others.

The decoding TR node also failed to make up a proper packet in case (d), in which several branches came together from upstream to one. In this case, packets with instruction for recording from multiple encoders might overwrite records in the cache of the gathering spot and caused conflict. When TR node E sent an encoded packet assuming that TR node G could decode properly, the corresponding record in the cache of TR node G might have been already replaced with a data sent from TR node F. This disorder was caused by any concurrent flow of TCP streams from two or more encoding TR nodes to one forwarding or decoding TR node.

Nevertheless, a destination host can detect error with checksum in TCP header and discard incorrect packets. A sender host executes retransmission after expiration of retransmission timer. The encoding TR node can know retransmission comparing IP addresses, port numbers, and sequence number of a received packet to list of traces for packets that have come by. The encoding TR node does not encode data on the retransmitted packet. The data arrives at the destination without recasting in turn. The most consequential problem is not delay of transmission but decline of throughput. TCP retransmission is accompanied with suppression of transmission rate because of mechanism for congestion avoidance.

![Diagram](traffic_reduction_region.png)

(a) Case of successful traffic reduction.

![Diagram](failure_transmission.png)

(b) Case of failure in transmission at the ordinary router.

![Diagram](cache_synchronization.png)

(c) Cache synchronization failure of un-sent record.

![Diagram](competition.png)

(d) Cache synchronization failure due to competition.

Figure 3. Restriction of service regions by the previous TR nodes.

Fig. 4 shows a target network topology. This is typical for server-client type service. Service regions with the TR nodes are expanded drastically. Resolving illegal change of IP header, we can add unknown network constructed with ordinary routers to the service region. Improving cache
synchronization method for branch of routes toward downstream side, we can apply service of the TR node to branch topology of a local area network. We can obtain significant benefit with these resolutions so that we tentatively postpone the problem of cache synchronization of branches coming from upstream.

3.2 Remediation of illegal change of the IP header

The encoder TR node puts an instruction code in the protocol field of the IP header of the received packets. The instructions are given in TABLE I. They are replaced with the protocol number of TCP by the decoder TR node. Codes F2 and FB are introduced in this study as explained later. The instructions require arguments in TABLE I and II. The previous encoding TR node wrote argument N on total length field in the IP header. Disagreement between true length and the value in the field means error for an ordinary router obeying IP. The router discards the packets encoded by the TR node with instruction FC and FE. The reason why we adopted the illegal change was to avoid increase of traffic in any case. If we stop using the total length field in the IP header for argument N, we must extend the length of a packet to send putting the argument in the other place. In this study, we placed emphasis on advantage to reduce traffic in many network segments with a few TR nodes.

The present TR node at the edge of upstream side in the service region makes a temporary packet adding the argument N to the end of data field for instruction FC and FE. If length of it exceeds the upper limit of length of the packet in the network segment, the temporary packet is fragmented as shown in Fig. 5. We introduced a new instruction F2 to notify the fragmentation for decoding TR node. The fragmentation is carried out with manner of IP except packet reassembly node. The decoding TR node reconstructs the original data. Assuming the same condition as we show in the section 2 for ideal reduction rate, we obtain $R_{ij}$ as $(1440(M-1)-48)/(1526M)$, which is only 0.02 smaller than that obtained by the previous TR node even if $M = 2$. The proposed method increases traffic if redundancy is not contained. Ideal reduction rate is $1438*(M-1)/(1524M)$ that is accomplished if fragment does not occur, that is, the length of data in the original packet is always 2 bytes shorter than the maximum length.

### TABLE I. INSTRUCTIONS EMBEDDED IN THE PROTOCOL FIELD BY THE ENCODER NODE

<table>
<thead>
<tr>
<th>code</th>
<th>Request</th>
<th>argument</th>
<th>(See TABLE II)</th>
</tr>
</thead>
<tbody>
<tr>
<td>F2</td>
<td>reconstruct and record</td>
<td>D,N,F</td>
<td></td>
</tr>
<tr>
<td>FA</td>
<td>no operation</td>
<td>D,N</td>
<td></td>
</tr>
<tr>
<td>FB</td>
<td>synchronize</td>
<td>D,N</td>
<td></td>
</tr>
<tr>
<td>FC</td>
<td>record</td>
<td>D,N</td>
<td></td>
</tr>
<tr>
<td>FD</td>
<td>decode</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>FE</td>
<td>decode and partially record</td>
<td>D,N,B</td>
<td></td>
</tr>
</tbody>
</table>

### TABLE II. ARGUMENTS FOR THE INSTRUCTIONS.

<table>
<thead>
<tr>
<th>argument</th>
<th>attribution(position)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>raw data (data field)</td>
</tr>
<tr>
<td>N</td>
<td>record number (IP header in the previous TR node, see text)</td>
</tr>
<tr>
<td>F</td>
<td>fragment control information (IP header)</td>
</tr>
<tr>
<td>B</td>
<td>encoded block (data field)</td>
</tr>
</tbody>
</table>

3.3 Cache Synchronization

Cache synchronization over arbitrary network is unrealistic. We tried to realize cache synchronization for the TR node sequence with branch toward downstream and without ordinary router. Data transmission between nodes is inevitable for cache coherency. There are roughly two methods. One is always to keep coherency. With this method, when the TR node sends an encoded packet with instruction FC or FE, it transmits the same data to all succeeding TR nodes. This is a simple method to implement. However, the transmission may be wasteful because there may be TR nodes that are not on the route of the redundant streams. Nevertheless, the transmission causes momentary increase of traffic. The other method is to delay synchronization until the
data becomes required to decode. The TR nodes guarantees cache coherency when it sends an encoded packet with instruction FD or FE for which its descender TR node must reconstruct an original data.

We introduced a new instruction FB for cache synchronization as shown in TABLE I. Destination of a packet with this instruction is the neighboring TR nodes so that TCP header is unnecessary. In this study, we assume branch topology is only in local network. We can allow the TR node to borrow total packet length field in IP header for record number. Note that the packet with FB instruction escape fragmentation if we put the record number in the data field since the packet length is shorter by size of eliminated TCP header.

The packet cache in the TR node is a large data table with several management fields. We appended fields of synchronization flag to the packet cache. The node on upstream side is responsible for cache synchronization between two neighboring nodes. The fields are arranged for all TR nodes on downstream side. When the TR node makes new record, it clears the fields of the record. As the TR node sends the data in the record to one of next hop nodes, it sets flag corresponding to the current next hop node. When the TR node sends a packet with instruction FD or FE, the node confirms whether the involved record in the encoded data is already sent to the next hop. If the corresponding field is unset, the node transmits the data. The most favorable case is that the received packet contains only one encoded block. The TR node reconstructs a packet with instruction FC and sends it only. If the received packet has more than two encoded blocks or instruction FE, the TR node generates synchronization packets for unsent data. It is also necessary to send the received packet in this case. Fig. 6 shows an example for generation of synchronization packets. The received packet has two encoded blocks of un-synchronized cache records in this example.

4 Evaluations and discussion

We built a computer network for evaluation of the proposed methods. The functions of the TR node was implemented on computers with AMD Opteron 1210 (1.8 GHz) CPU, 1GB main memory, Debian 4.0 (Linux 2.6.18-6-486) operating system. The redundant contents consisted of random numbers. Line speed was 100 Mbps. In the present measurements, sender sent packets using bandwidth fully.

4.1 Passing an ordinary router

To confirm validity of the proposed method described in the section 3.2, we constructed a computer network shown in Fig. 1, and set an ordinary router between two TR nodes. We emulated 5 receivers with 5 processes working on a machine. We counted number of packets with each instruction at upstream side and downstream side of the router. Measurements were carried out a few seconds and 300 seconds. Fig. 7 shows the result this measurement.

With the present modification, packets went through the ordinary router successfully. If all redundant streams are divided at the same offset and data length is always maximum, the encoding TR node sends 4 packets with instruction FD for 5 received packets, and 2 for the 5 with instruction F2. Number of sent packets is larger than that of received packets since fragmentation occurs. In the condition of the present experiment, packet sizes were mostly maximum frame size so we observed packets with instruction F2 frequently. As mentioned before, this is not a condition that gives the best reduction rate. Furthermore, packets with instruction code FE were generated for stream division at different position. We obtained \( R_n = 0.74 \). The maximum reduction rate is 0.75 for \( N = 5 \) as described in the former section.

![Figure 7](image)

Figure 7. Number of packets for each type and behavior of an ordinary router for encoded packets.

We can obtain benefit of redundant traffic reduction with less TR nodes with the present modification. For example, we consider such network that contains 2 consecutive network segments separated by 3 network nodes. We can achieve \( R_n = 0.74 \) with 2 TR nodes at the ends of
considered region under the same condition of redundant traffic as the present experiment. If we use the previous TR nodes, 3 TR nodes are required to obtain nearly equivalent efficiency to $R_n = 0.74$. When we can prepare only 2 TR nodes of the previous type, we must place an ordinary router before or after sequence of the 2 TR nodes and elimination of redundant traffic is performed only in one network segment. Then $R_n = 0.38$ over the region. Advantage of the present specification becomes larger in the network with more ordinary routers.

4.2 Validity of cache synchronization

We carried out measurement for evaluation of the present cache synchronization using the computer network shown by Fig. 8. The sender dealt 6 redundant streams, that is, 3 streams to the receiver Y and A, respectively. Transmitted data by the sender split on halves toward TR node C and D. In ideal case, one for 6 packets of received packets by TR node B is of instruction FC and 5 for 6 are of FD and have the smallest data field. Then the node sends 2 packets of FC and 4 of FD for 6 received packets. Packets with instruction FB are never generated. $R_n = 71$ in this case.

![Figure 8. An example of TR node network with branch constructed for evaluation](image)

Experimental results are given in Fig. 9 and Fig. 10. The former presents transmission rates and the latter shows rate of packets with each instruction sent by TR node B. Segment A-B means zone between TR node A and B in the experiment. Fig. 10 demonstrates generation of packets with instruction FB and denotes existence of packets with instruction FD having multiple encoded blocks. Hence, $R_n$ was lower than ideal value. If we construct the network of Fig. 8 with the previous TR nodes, TR node B must forward encoded packets to only TR node C (or D), regarding TR node D (or C) as an ordinary router and decodes packets for it (or C). Otherwise, throughput significantly drops because of congestion avoidance by TCP. TR node B sends one packet with instruction FC and two with FD to TR node C, and three reconstructed packets to TR node D for six received packets. Then $R_n = 0.31$ at most. Thus, the obtained value of $R_n$ with the proposed cache synchronization method was much better than that with the previous TR node. Advantage of the present TR node increases for larger TR node network.

![Figure 9. Reduction of redundant traffic using the present cache synchronization method.](image)

![Figure 10. Rates of instructions](image)

5 Related works

Concept of shared cache in [7] is very similar to the TR node. Rabin fingerprint [8] is used to generate identifier of cached data. Cache coherency is assumed in the discussion of it. [9] and [10] also use Rabin fingerprint and routing is optimized for redundant traffic elimination. In [9], all caches on routers that are involved in traffic reduction hold the same data. On the other hand, a central module to manage status of caches is introduced in [10]. It offers deployment of redundant data over caches along route of packets in order to save memory space. [7], [9], and [10] emphasize independency of protocol or application. Procedure proposed in [11] is advantageous in memory saving. Cache of ingress router holds only identifier. Policy of synchronization that all egress routers must synchronize with corresponding ingress routers completely is the same as [7] or [8]. Study of [11] is focusing on P2P traffic exchanged by a particular application BitTorrent. Data identifiers are obtained by the hash function MD5. Traffic reduction routers in [11] confirm cache
coherency with message sent by egress routers to the ingress ones as data updated.

We lay weight independency of protocol in the same way as [7], [9], [10], while region for traffic reduction is different from those studies. Their targets are ISP network. On the other hand, our target is traffic traversed WAN and LAN as shown in Fig. 3. The TR nodes do not suppose optimal routing for redundancy elimination whereas it must know whether if there is another TR node on downstream. This characteristic can cut cost of introducing functional router and routing management. Furthermore, server-client type service is our target so that assigning responsibility of cache synchronization to the upstream node is appropriate rather than the other method. Some differences between related works and our study are trade off. One of them is method to obtain identifier. We use so simpler method than the other related works do that load of the nodes is lighter. However, the identifier does not reflect characteristic of whole data. Therefore, the TR node is weak for accidental matching. The next problem is memory saving. Memory saving in [10] is substitution between resource and reduction rate, or memory and bandwidth. Similarly, saving copy of data in the encoding TR node is memory consuming but free from collision.

6 Conclusions and Future works

In this paper, we proposed procedures to apply service with the TR node to wider network region. One is to enable to employ its service over network including ordinary routers. It is achieved by shifting an argument for cache management from the field in IP header to data field, with fragmentation if required. The other is to adapt the TR node for branch topology. It is accomplished with cache synchronization control sending synchronization packets. These methods are implemented to the computer network constructed for experiments. Validity of our proposed method was shown by experiments and estimations.

The most important advantage of the TR nodes is redundant traffic reduction of real time transmission. TV meeting is one of the most significant application as well as live broadcast with sever-client type delivery. In the case of TV meeting, two or more sender exists and cause competition of cache utilization. In future work, correspondence to branch topology with multiple encoder TR nodes is desirable.

7 References


