

Development of an Optimization Algorithm for Internet Data Traffic

Syed Misbahuddin and Tariq Ibn Aziz

Department of Computer Science and Software Engineering

Hail Community College

King Fahd University of Petroleum and Mineral

PO Box 2440, Hail, Saudi Arabia

{smisbah, taziz} @kfupm.edu.sa

Abstract

In recent era of Information Technology the data traffic over the Internet is increasing uncontrollably. This proliferation of data traffic is due to general shift towards e-business and other application of Information Technology. The businesses relying on Internet lose billions of dollars each year due to slow or failed web services. Therefore, in Internet research, the most conspicuous issue is to develop methodologies to reduce net traffic over the Internet. In this paper, an optimization algorithm is proposed to reduce net data traffic, which works at Internet layer in the TCP/IP reference model. The algorithm monitors data repetitions in IP datagram and prepares a compression code in response of this repetition. If no IP datagrams are repeated, no compression code is sent. Therefore, the algorithm does not put any overhead on the system. Furthermore, as the proposed algorithm works at IP datagrams only, therefore, it remains transparent from all client-server applications.

1. Introduction

The WWW traffic is growing exponentially day-by-day with significant shift towards e-commerce and increasing Internet usages in the society. However, the exorbitant net data traffic over the Internet leads to slow responses to the net users and consequently the e-commerce web sites lose about \$4.3 billion per annum [1]. The quality of web service and proper web response time is influenced by two factors: the web server's slow response and net congestion over the Internet. The web server's slow response is obviously connected with number of client hits. The server's response may be improved improving web servers' performance [2]. Net congestion over the Internet may be controlled by two methods: web-caching and data compression [1]. In web caching, the history of web objects is maintained at some caching servers. When a client requests a web object, a specific cache server generates the requested object instead of obtaining the object from original web server. The web caching mechanism helps reduce the net data traffic. However, web caching is still a challenging area

because all web objects are not cacheable [3]. Data compression algorithms are applied on the information content of the web object.

Syed Misbahuddin et al have presented a data reduction algorithm, which works for message communication based data networks [6]. In this algorithm, the processors connected to a data network, maintain the history of transmitted and received messages. The processor at transmitting side prepares a compression code if some data bytes in the message are repeated and send the non-repeated data bytes to the network. The receiving end prepares the complete message with the help of the message history and the received non-repeated data bytes. The objective of this paper to extend the data reduction algorithm proposed in [6] to IP datagrams. Section 2 reviews IP datagram and section 3 discusses the motivation and the proposed data reduction algorithm. Section 4 presents performance analysis of the proposed data reduction algorithm. Finally, the conclusion is presented in section 5.

2. Review of IP Datagram

In connectionless Internet services, the web objects are broken into individual data packets, which are sent over the Internet independently. These data packets carry information about the intended recipients. At the receiving end of Internet model, the receiver software combines the received packets and reconstructs the originally transmitted web object. Figure 1 shows the general model of IP datagram.

Header	Data Area
--------	-----------

Figure 1: Format of an IP datagram

An IP datagram travels independently over the net. The IP datagram contain variable length of data field. The size of data field depends upon the application sending the data over the net. In current version of IP version 4.0, a datagram size varies from single octet to 64k octets including the header. The header portion contains routing and other informational details about the datagram [4].

3. Data Reduction Algorithm Motivation

It may be commonly observed that in a client-server interaction, a client hits a web server multiple times. This kind of situation is especially observed in web-based emails services, on-line shopping and in some e-commerce. During multiple client-server interactions, most of the web object content may remain intact for long time. The web server uses several session variables to maintain the currency of an already shipped web object to a particular client. These session variables are originated by the web clients and sent to the web server. Furthermore, some commercial web servers maintain the history of the interactions from web clients. For example, a server maintains history of advertisement pushed to a particular client [5]. Based upon these observations of repeated content of web objects and the availability of the client information at the server side, a Data Reduction (DR) algorithm has been investigated in this paper. The DR algorithm is based upon following assumptions.

3.1 Assumptions

1. The Internetworking model is connectionless, which uses IP datagram for information exchange. All IP datagrams follow IP version 4.0.
2. One bit in "Type of service" field of IP version 4.0's header is used to denote the data reduction process. This bit will be defined as Data Reduction Bit (DRB).
3. The algorithm is implemented at Internet layer in the TCP/IP model at both client and server side.
4. Both client and server maintain limited of histories of IP datagrams of web objects.
5. Each IP datagram is assigned the identification number according to the content of web page.
6. The data field length in IP datagram is at least 8 bytes long.

3.2 The Algorithm

The algorithm divides data field of IP datagram into fixed eight groups. The width of each group varies from 1 byte to N/8 bytes for 8 bytes to N bytes long data field in IP datagram respectively. According to the algorithm, the web server keeps a copy of the recently transmitted IP datagram in a buffer called T_BUFF. Each entry in T_BUFF consists of two fields: ID field and data field. The ID field holds the information to identify a particular IP datagram. The data field keeps a copy of the data field of the transmitted datagram. When a client requests the same web object, the data reduction algorithm, intercepts the generated IP datagram to

check its presence in T_BUFF. If T_BUFF contains the copy of the IP datagram, the data reduction algorithm will verify if the data field of the IP datagram is changed. If some data bytes groups have not been changed, then the DR algorithm will set DRB in the IP header to "1" to reflect the repetition of data bytes. The algorithm will then prepare an eight bit compression code (CC) to indicate the repeated data bytes group in the IP datagrams' data field. In the compression code, a bit with a value of "1" indicates that a corresponding data byte group has been repeated. A bit with a value "0" indicates a data byte group is not repeated. The non-repeated data byte group(s) will follow the compression code in the data field of IP datagram. The indices of repeated and non-repeated data byte groups is determined by bit numbers of compression code with values "1" or "0" respectively. The IP datagram with compression code is shown in Figure 2 and data reduction algorithm is summarized by flow chart shown in Figure 3.

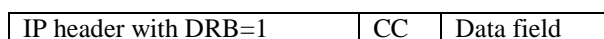


Figure 2: IP Datagram with compression

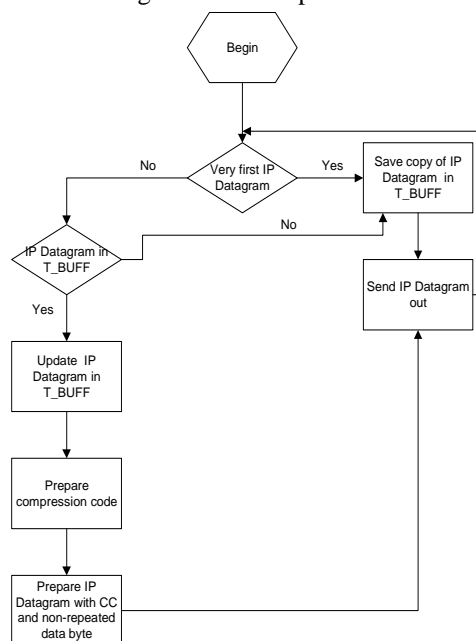


Figure 3: Data reduction algorithm executed at the Web server side

The "decompression algorithm" running at the client side recovers the actual transmitted IP datagram from the received IP datagram. The algorithm checks the data reduction bit in the received IP datagram. If this bit is "1", then the process will assume that some data byte groups in the received datagram have been repeated and the copies of the repeated data bytes groups are already present in R_BUFF at the client side. In this case, the client

will interpret the very first byte of the received datagram's data field as an eight bit compression code. The compression code will describe the indices of the repeated and non-repeated data byte groups. The data decompression algorithm will collect non-repeated data bytes from the received IP datagram and repeated data bytes groups from R_BUFF buffer at the client side. The reconstructed IP datagram will be sent to the appropriate layer to reconstruct the received web page. The data decompression process can be summarized in flow chart shown in Figure 4.

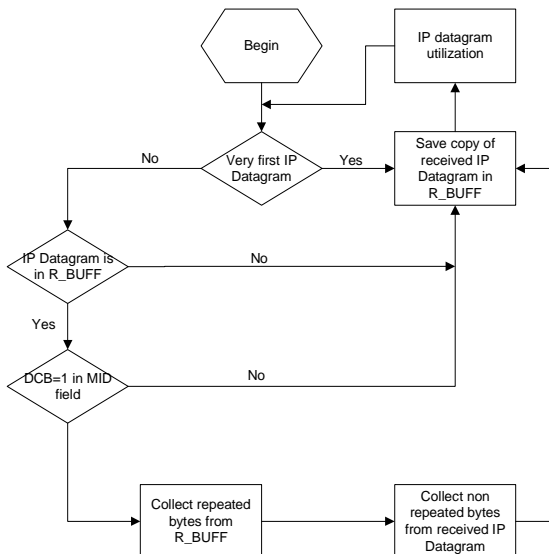


Figure 4: IP datagram reconstruction process at the client side.

To illustrate the IP datagram reconstruction process at the client side, we assume that the IP datagram's data field is 8 bytes long. According to DR algorithm, the data field is divided into eight groups and the size of each data byte group is one byte long. Assuming groups 0, 1, 2 and 3 are repeated in the data field of IP datagram, the CC's bits 0, 1, and 3 will be set to "1." To indicate the positions of non-repeated data bytes, the bits 4 to 7 of the CC will be set to "0." The CC to reflect this situation is shown in Figure 5. The IP datagram received at the client side along with compression code and non-repeated data bytes is shown in Figure 6.

Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7
1	1	1	1	0	0	0	0

Figure 5: Compression code showing repeated and non-repeated data byte group's indices

IP datagram Header with DRB=1	CC	Non-repeated data bytes

Figure 6: The IP datagram received at the client side

In this scenario the data decompression algorithm running at the web client side, retrieves bytes 0, 1, 2 and 3 from R_BUFF and bytes 4, 5, 6 and 7 from the received IP datagram. In this example, total 5 data byte groups are transmitted instead of 8 data byte group (CC + 4 non-repeated data byte group). In other words, due to DR algorithm, the IP datagram arrived at the client side in relatively short period of time. Therefore, by applying the proposed data reduction algorithm, more IP datagrams can be transferred from web server to a web client in given amount of time. The number of transmitted data bytes decreases as the number of repeated data bytes increases. Consequently, if all data bytes are repeated then only one byte of compression code will be sufficient to represent all data bytes. If no data bytes are repeated, the compression code is not needed and IP datagram transmission remains normal.

4. Discrete Event Simulation Model for the Proposed Data Reduction Algorithm

In order to obtain a quantitative performance of the proposed data reduction algorithm in [6], a discrete event simulation model was developed. This model schedules several events at different times. Some of the events are: message arrival event, bus ready event, report event etc. Message arrival event is scheduled when a processor generates a message. As the data reduction algorithm presented in our paper is an extension and modification of the algorithm presented by Syed Misbahuddin et al, the same simulation model can be utilized for the quantitative performance analysis of DR algorithm discussed here. The simulation program developed in [6] has been tailored according to the DR algorithm for Internet data traffic congestion control. For the purpose of comparison, the simulation program was executed with and without proposed data reduction algorithms and various performance parameters were estimated. The simulation model used following assumptions:

1. A scenario is assumed in which a client contacts a web server several times.
2. An IP datagram is composed of 8 bytes of data field and 64 bytes of IP header field. Eight bytes long data field is divided into eight groups of single byte long each.
3. A web object is broken into 20 IP datagrams.

4. During a client-server interaction session, the individual IP datagrams are sent to the web client at specific period.
5. Data bytes in each IP datagram are repeated with probabilities described in Table 1:

Data Bytes	Probability of Repetition
0	0.0005
1	0.0005
2	0.9999
3	0.9999
4	0.9555
5	0.9555
6	0.9999
7	0.9999

Table 1: Data repetition probabilities

The probabilities shown in second column of Table 1 are based upon the assumption that in some web pages, some web content may have high chances of data repetition. For instance, when a web client hits a web based email server multiple times, the inbox web page may have over 90% chances of information repetition.

4.1 Compression Ratio

Compression ratio is defined as the percentage ratio between data bytes saved from transmission to the actual number of data bytes in an IP datagram. In order to evaluate the compression ratio for the proposed algorithm, all repetition possibilities are considered. If all data byte groups in an IP datagram are repeated, then only one byte of compression code will be sent instead of sending whole data field. If seven out of eight data byte groups are repeated then the compression code is sent followed by one non-repeated data byte group. Similarly, if 6 data byte groups out of eight are repeated then the compression code is sent followed by two non-repeated data byte groups and so on. Table 2 lists all the repetition possibilities in a typical IP datagram along with the achieved compression ratio.

RB	NRB	Number of transmitted bytes	Number of bytes Saved	CR (%)
8	0	CC+ (0 NRB)=1	7	87
7	1	CC+ (1 NRB)=2	6	75
6	2	CC+ (2 NRB)=3	5	62
5	3	CC+ (3 NRB)=4	4	50
4	4	CC+ (4 NRB)=5	3	37
3	5	CC+ (5 NRB)=6	2	25
2	6	CC+ (6 NRB)=7	1	12
1/0	7	CC+ (7 NRB)=8	0	0

RB = Repeated bytes group NRB: Non-repeated bytes groups

Table 2: Compression ratio for a typical IP datagram

Last column in Table 2 shows the compression ratio corresponding to the repetition level. For the best case the compression ratio is 87% when all data byte groups are repeated. In worst case, when one or

no byte groups are repeated, no compression code is sent and therefore, the compression ratio is 0%. In this case, the client will consider the received IP datagram as a normal datagram and, therefore, will not interpret the very first byte in the data field as the compression code.

4.2 Average Router Queue Length

The routers are used to forward the Internet data packets to their intended destination. A packet reaches to its destination after traveling through various routers. Due to net-congestions, routers maintain packets queues. When a queue at router exceeds certain threshold, then newly arrived packets are dropped. By applying the proposed data reduction algorithm, an IP datagram takes less time in the router if the data byte groups are repeated. In other words with the proposed data reduction, the router will remain less congested. Table 3 shows that without the data reduction algorithm, on average there are eleven IP datagrams waiting in the router queue. On the other hand, approximately eight IP datagram packets are queued up with the proposed data reduction algorithm. This means that are 27% less IP datagrams are waiting in the queue on average when the proposed data reduction algorithm is applied. Figure 7 shows the graphical relationship between router queue length and number of IP datagrams.

Number of IP datagrams	Average Router queue length	
	Without compression	With compression
5	1.466336	0.401681
10	3.987524	1.85949
15	7.59334	4.382214
20	11.601007	7.963825

Table 4: Router queue length Vs number of IP datagrams

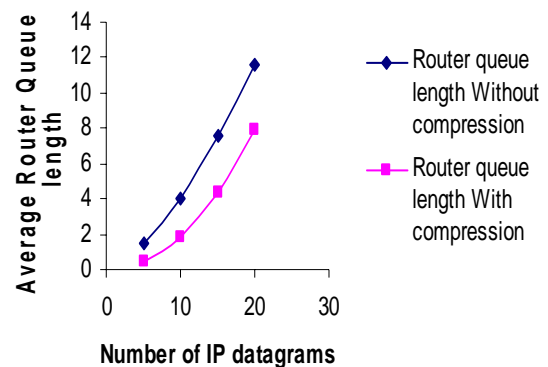


Figure 7: Average router queue length Vs number of IP datagrams.

Table 5 shows the router queue length in terms of the number of data bits. Figure 8 shows the graphical representation of Table 5. This result shows that there is less data accumulation in the queue when the proposed data reduction is applied.

Number of IP datagrams	Average Router queue length in bits	
	Without compression	With compression
5	846.51566	208.630217
10	2291.735525	963.188289
15	4377.948363	2262.166587
20	6677.449564	4103.463786

Table 5: Average router queue length in bits Vs number of IP datagrams

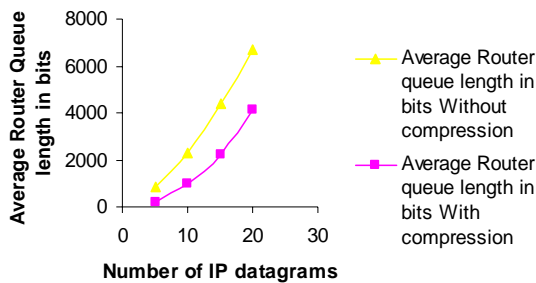


Figure 8: Average router queue length in bits Vs number of IP datagrams

4.3 Average IP datagram Delay

Average IP datagram delay is defined as the average amount of time spent by an IP datagram in the queue at a router. The numerical results generated from this simulation show that average IP datagram delay is significantly low when the proposed data reduction algorithm is applied. Table 6 compares average IP datagram delay with and without the data reduction algorithm. Figure 9 shows the graphical representation of Table 6.

Number of IP datagrams	Average IP datagram delay in seconds	
	Without compression	With compression
5	0.001478	0.000425
10	0.002014	0.000962
15	0.002555	0.001461
20	0.002915	0.001989

Table 6: Average IP datagram delay Vs number of IP datagram

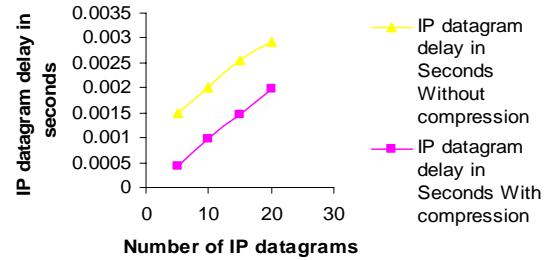


Figure 9: Average IP datagram delay Vs number of IP datagrams.

5. Conclusion

The Internet traffic is growing due to increasing trends of its application in all facets of life. Opening new web sites, online shopping, online news, online buying and selling of stocks, online banking, Internet emails and Internet telephony are some examples of major contributing factors for increased net traffic and net congestion. It may be easily observed that in most client-server interactions, most of web page contents remain unchanged. This observation of repeated web object can be exploited to devise a data reduction algorithm. In this paper, an optimization algorithm for Internet data reduction has been proposed, which utilizes the content repetition of web objects. The proposed algorithm generates a compression code if some data bytes in the IP datagram are repeated. The web client can reconstruct the original IP datagrams with the help of compression code and received non-repeated data bytes. The performance of the proposed data reduction algorithm has been evaluated in terms of queue length at router and IP datagram delay. The numerical results generated from the simulation model indicate that the proposed data reduction algorithm helps reduce the data congestion at Internet and improves web transactions

6. References

- [1] Mazen Zari, Hossein Saiedian and Muhammad Naeem, "Understanding and reducing web delays," IEEE computer, December 2001, pp. 30-37.
- [2] J. Bangs and J. Mogoul, "Scaleable Kernel Performance for Internet Servers under Realistic Loads," Proc. Usenix 1998 Technical Conf., Usenix, Berkeley, California, pp. 1-12.
- [3] J. Almeida, V. Almeida, and D. Yates, "Measuring the Behavior of a World Wide Web Servers," Proc. 7t IFIP conf. High Performance Networking (IFIP), Kluwer Academic Publishers, Norwell, Mass., 1997, pp. 57-72.
- [4] Ed Taylor, "The Network Architecture Design Handbook," McGraw-Hill, 1998.
- [5] Douglas E. Comer, "Computer Networks and Internet", Prentice Hall, 1997.

- [6] Syed Misbahuddin, Syed M. Mahmud and Nizar Al-Holou “ *Development and performance analysis of a data reduction algorithm for automotive multiplexing*” IEEE transactions of Vehicular technology, Vol. 50, No. 1, January 2001.

Biography

Dr. Syed Misbahuddin received the B.E. degree in electronics engineering from Dawood College of Engineering and Technology, Karachi, Pakistan, in 1983, M.S. degree in electrical engineering from King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia in 1988 and Doctor of Engineering degree in Electrical and Computer Engineering from the University of Detroit Mercy, Detroit, MI, USA in 1998.

From 1988 to 1992, he was working as an Assistant Professor in the department of Computer engineering, NED University of Engineering and Technology, Karachi, Pakistan, where he was engaged in variety of computer engineering research. Currently, he is an Assistant Professor in the department of computer science and software engineering, Hail Community College, KFUPM, Hail, Saudi Arabia. His research interests are in the area of distributed computing, data reduction algorithms and microprocessor engineering. He is member of IEEE.

Tariq Ibn-i-Aziz received his M.Sc in Computer Science from Quaid-i-Azam University, Islamabad, Pakistan and MS in Computer Science from University of Bridgeport, CT, USA. Mr. Tariq Ibn-i-Aziz is a lecturer in the department of Computer Science and Software Engineering, Hail Community College, KFUPM, Hail, Saudi Arabia. He has over 15 year of teaching and industrial experience in computer science. Before joining KFUPM, Mr. Aziz was working as an IT consultant in various North American companies. His research interest area are in Java based business solutions and Internet Data Traffic Engineering.