P5F.3

Distribution of real-time digitized weather radar signal over high bandwidth network

Yoong-Goog Cho and V. Chandrasekar
Colorado State University, 1373 Campus Delivery
Fort Collins, CO 80523
Email: ygcho@engr.colostate.edu

1. Introduction

Colorado State University initiated Virtual CHILL (VCHILL) project which is an initiative to enable the real-time operation of radar over the Internet (Chandrasekar et al. (2001)). The VCHILL project aims at providing the same quality of end products at remote sites as at the radar site in real-time. The transfer of real-time digitized radar signal (DRS) over high bandwidth network involves challenges that include real time requirements for end product computation, deficiency of available bandwidth, as well as variable latencies introduced by physical distance, router delays and end system performances (Chandrasekar and Jayasumana (2001)).

In this paper, we propose an end system architecture for transmitting the real-time dual-polarized DRS over User Datagram Protocol (UDP), estimating the radar parameters at the remote sites and delivering them to the display nodes. The architecture is developed based on the client-server model with the operation of multiprocesses and multithreads. The design includes the digitized radar signal acquisition, transmission, radar parameter computation, parameter transfer, as well as generic packet and data structures for the data transmission and sharing. In addition, we develop a congestion control algorithm to adapt the transmission rate to the available bandwidth such that the data stream conforms with the 'TCP friendly' behavior (Mahdavi and Floyd (1997)). The algorithm is accompanied with a data filtering policy in order to provide the highest quality of the estimated radar parameters possible under any network conditions. The proposed architecture for the CSU-CHILL radar has been successfuly implemented on the Sun/Solaris and Linux platforms and its functionality was confirmed via emulation performed over gigabit link.

2. End system architecture

The design is aimed at transmitting the real-time DRS over UDP through a high bandwidth data network, such as Next Generation Internet. The end system architecture which consists of multiprocesses and multithreads is illustrated in Fig. 1. Primary functions of each process are listed in Table 1. Details on the basic design concepts and the way in which processes collaborate together are described in (Cho et al. (2002)).

A collection of the multiple range sample data sets at a particular pointing angle is defined as a ray data block (Cho et al. (2002)). An user payload can be either a ray header or a range sample data set. The ray header is transmitted in the beginning of the ray data block transmission, which characterizes the DRS. The fields of ray header are categorized into data identification, radar operating condition, data for calibration and data for transmission as shown in Table 2. The data for transmission are used for the congestion control. Structure of a header that is placed in front part of the range sample data set is shown in Table 3. These header structures are so designed in a general fashion that they are applicable to various radar systems and applications.

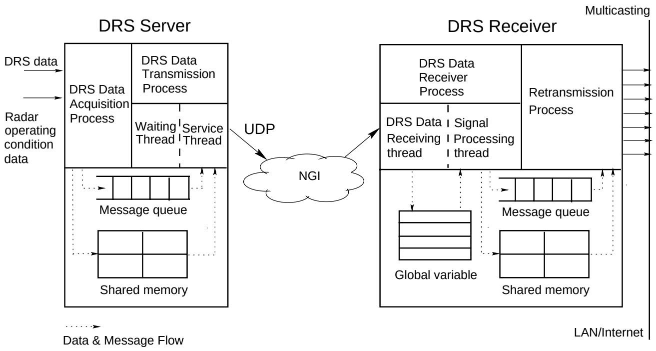

Figure 1: Overall end system architecture for transmitting real-time DRS and computing radar parameters.

In order to transmit the real-time DRS data over UDP, a congestion control algorithm is required such that the transmission flow conforms with the 'TCP friendly' behavior (Mahdavi and Floyd (1997)). Figure 2 shows the end system architecutures for the congestion control performed by a rate shaping algorithm. The function of the congestion control is included in the threads that are responsible for the end-to-end data transmission. At the server, the bandwidth estimator calculates the total range sample data loss rate and estimates the available bandwidth. The information is passed to the transmission rate shaper which selects the range sample data sets to transmit among a ray data block. At the receiver, the data loss counter counts the total data loss and the feedback module sends the relevant information to the server.

● Data acquisition process · Read radar operation parameters from radar controller. · Read digitized radar signal from digital receiver. · Assemble header information. · Write header information and DRS data into shared memory.
● Data transmission process · Create listening TCP socket. · Create threads for serving the accepted client with UDP socket. · Book keeping for all client socket information. · Read the DRS data from the shared memory. · Assemble packets and transmit them. · Estimate available bandwidth. · Control transmission rate.
● Data receive process · Create TCP/UDP socket and establish TCP connection to the DRS server. · Request and get the DRS data over UDP socket. · Count range sample data loss rate and send the information to the DRS server. · Compute the radar parameters. · Calibrate reflected signal power. · Scale the calculated data for display. · Make header information for transmitting parameters. · Write the radar signal parameters into shared memory.
● Parameter transmission process · Make UDP sockets for multicasting data transmission. · Read header information and the radar signal parameters from the shared memory. · Make data packet which contains header and data. · Transmit the data to multicating groups.

Table 1: Functions of each process designed for the real-time DRS data transmission and computation.

Data Identification	
Header ID	0 - ray header, 1 - range sample data set header
Radar ID	Unique number for a specific radar
Start time	Date and time in Unix
Radar operating condition	
OP mode	0 - V only, 1 - H only, 2 - VH, 3 - VHS
Scan mode	0 - RHI mode, 1 - PPI mode
Sweep number	Integer
Ray number	Integer
Azimuth	Azimuth angle degrees × 1000000
Elevation	Elevation angle degrees × 1000000
Pfr	Pulse repetition frequency × 1000
Ngates	Number of gates
Gate spacing	Millimeters
Start range	Millimeters
Npulse	Number of pulses transmitted at a particular angle
Calibration	
Txmit power H	Horizontal peak transmit power dBm × 100
Txmit power V	Vertical peak transmit power dBm × 100
Receiver gain H	dB × 100
Receiver gain V	dB × 100
Zdr offset	dB × 1000
Test type	TBD
Transmission	
Npulse packet	number of range sample data sets in a data packet
RTT	round trip time(msec)
Transfer rate level	1 - 10
Transport protocol	0 - TCP, 1 - UDP

Table 2: Structure of ray header.

3. Congestion control algorithm for the real-time transmission

Data rate generated by a radar is determined by a combination of various factors, such as dynamic range of the signal, receiver bandwidth and number of receive channels. For a dual channel coherent receiver system sampling at 1 MHz with the dynamic range of 90 dB (quantized with 16 bits), the data rate can be estimated to be 64 Mbps. The transmission rate level is designed such that the full data rate is divided into 10 levels with each level corresponding to a data loss rate. Level 10 indicates full data transmission at the rate generated by the radar whereas level 1 would correspond to 90% data loss.

The congestion control algorithm can be classified as a source-based rate control with Additive Increase/Proportionate Decrease (AIPD) algorithm (Widmer et al. (2001); Lee et al. (2001)). Figure 3 shows a timing chart of data exchange between the server and the receiver. In the beginning the sender transmits a ray header after getting a request of the DRS data transmission. The transfer level in the ray header enables the receiver to count the loss of the range sample data sets during transmission of a ray data block. Following the ray header transmission, the sender starts to transmit the range sample data packets. The receiver sends the feedback, which contains the transfer level and the total number of range sample data loss to the server, upon the arrival of the last packet of a ray. The feedback packet arrives at the server during the following ray data transmission because of the propagation delay. The server checks the arrival of the feedback packet after transmitting the last range sample data set of a ray. Once the server finds a feedback packet to read, the server reads it and estimates the available bandwidth and updates the next transmission rate. However, if there is no feedback packet to be read in the server UDP socket buffer, then the server continues to transmit the next ray header and range sample data sets with the previous transmission rate. A transmission rate is kept constant during transmission of a ray data block. The structure of the feedback packet is shown in Table 4.

Header ID	0 - ray header, 1 - range sample data set header
Sweep number	Integer
Ray number	Integer
Data number	Sequence number of range sample data sets in a ray
Data code	0 - normal data, 1 - last data in a ray, 2 - retransmitted data

Table 3: Structure of a range sample data header.

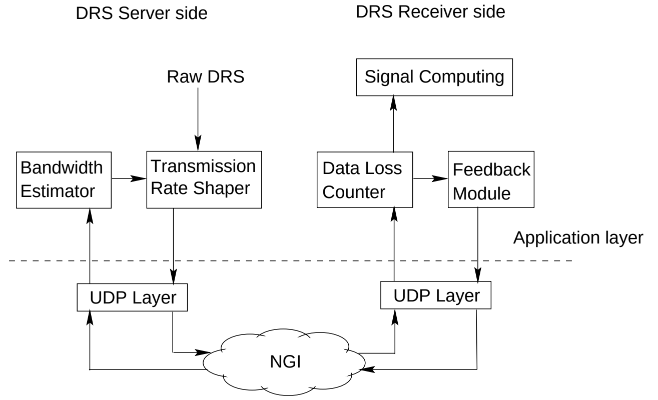

Figure 2: An end system architecture for performing congestion control over UDP.

The server updates the transmission rate by selecting the range sample data sets to be transmitted as proposed in (Cho and Chandrasekar (2003)). If no data loss is reported, the server increments the transmission rate level by one. However, if a certain amount of data loss is reported, the server adjusts the transmission rate to adapt to the available bandwidth. A combination of this congestion control algorithm with the optimized selection of the range sample data provides the best quality of service under the dynamically changing network conditions.

4. Implementation and performance evaluation

The proposed end system architecture for transmitting the real-time DRS of the CHILL is evaluated on both Sun/Solaris and Linux platforms. A simple test bed was used consisting of a server and a receiver connected through a network emulator. NIST Net emulation package changed the available bandwidth of the established gigabit link between the server and the receiver. The DRS data, which was simulated with the dual-polarized Doppler parameters, was used for evaluating the implemented program functionality and performance. The following radar operating conditions were assumed; a) the pulse repetition time is 4/3 msec, b) the number of pulses radiated at a specific angle is 128, c) the angular resolution of a ray is 1 degree, and the sampling frequency is 1 MHz.

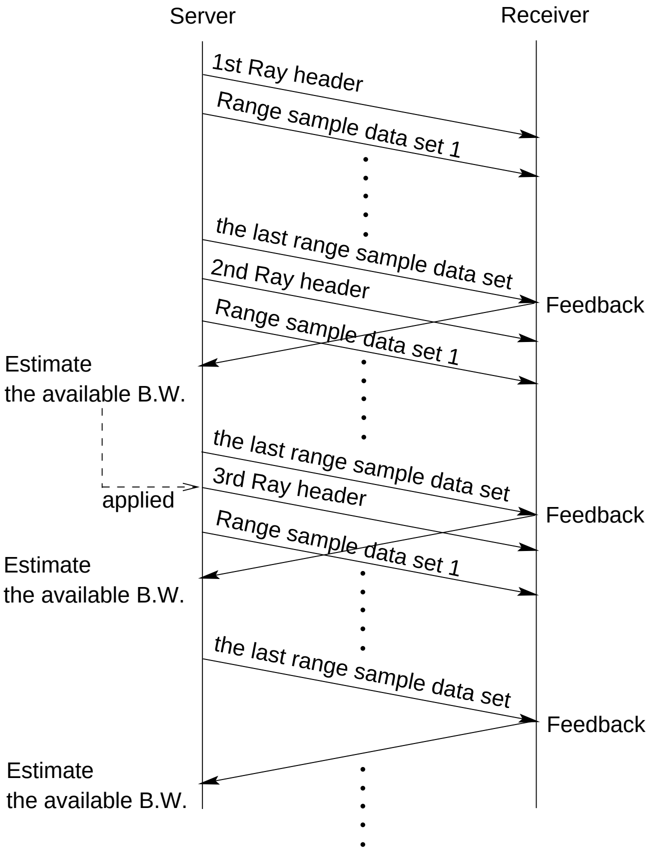

Figure 3: Timing diagram between the server and the receiver.

The DRS data was stored in the server RAM after being loaded. The data acquistion process emulated the radar operation by writing the ray header to shared memory followed by the 128 range sample data sets sequentially. The data loading speed was set to be constant around 65 Mbps, which approximately corresponds to a sampling frequency of 1 MHz. However, the transmission rate was varied with the period of a ray data block transmission time based on the available bandwidth. The NIST Net set the available link bandwidth with 10 levels. Then, the highest bandwidth and step were 64 Mbps and 6.4 Mbps, respectively. One way delay was set to 50 msec.

Figure 4 shows the change in the transmission level with different available bandwidth. It is clearly observed that the server adapts the transmission rate dynamically to the available bandwidth, which resembles the changes in TCP congestion window size. Comparison of displays between the case when our proposed congestion control algorithm is employed and the case with no congestion control also shows the improvement of the end user display.

Header ID	0 - request for transmission, 1 - feedback, 2 - request for retransmission
Message	Reserved
Sweep number	Integer
Ray number	Integer
Transfer level	1 - 10
Data loss	total number of range sample data loss at a transfer level
Pulse number for retransmission	Reserved

Table 4: Structure of feedback packet. Retransmission associated fields are included for future upgrade.

5. Summary and Conclusion

End system architecture that is tailored for transmitting the high bandwidth real-time weather radar signal over UDP, including data and packet structures, has been proposed. The architecture is designed based on the client-server model with multiprocesses and multithreads in order to meet the real-time requirements. In addition, a congestion control scheme, which can be classified as a traditional source based rate control with AIPD algorithm and feedback, has been developed to conform with the 'TCP friendly' behavior. The scheme maximizes the quality of the end products of the radar data by intelligently selecting the transmitted data. The performance evaluation applied to the CSU-CHILL radar clearly shows that the server dynamically adapts the transfer rate to the available bandwidth. The user end radar displays also show much higher quality compared to the case with no congestion control. The design is generalized and modularized in such a way that it is applicable to any radar signal transmission applicatons.

6. Acknowledgment

This research was supported by the National Science Foundation (NSF) (ATM-0121546) and DARPA NGI Program. The CSU-CHILL radar is supported by NSF through a cooperative agreement (ATM-0118021).

References

Chandrasekar, V., D. Brunkow, and A. P. Jayasumana: 2001, CSU-CHILL operation over the Internet: Virtual CSU-CHILL. *The 30th International Conference on Radar Meteorology*, 58–60.

Chandrasekar, V. and A. P. Jayasumana: 2001, Radar design and management in a networked environment. *Proceedings of SPIE*, volume 4527, 142–147.

Cho, Y.-G. and V. Chandrasekar: 2003, Real-time weather radar signal transmission. *Proceedings of IGARSS '03*.

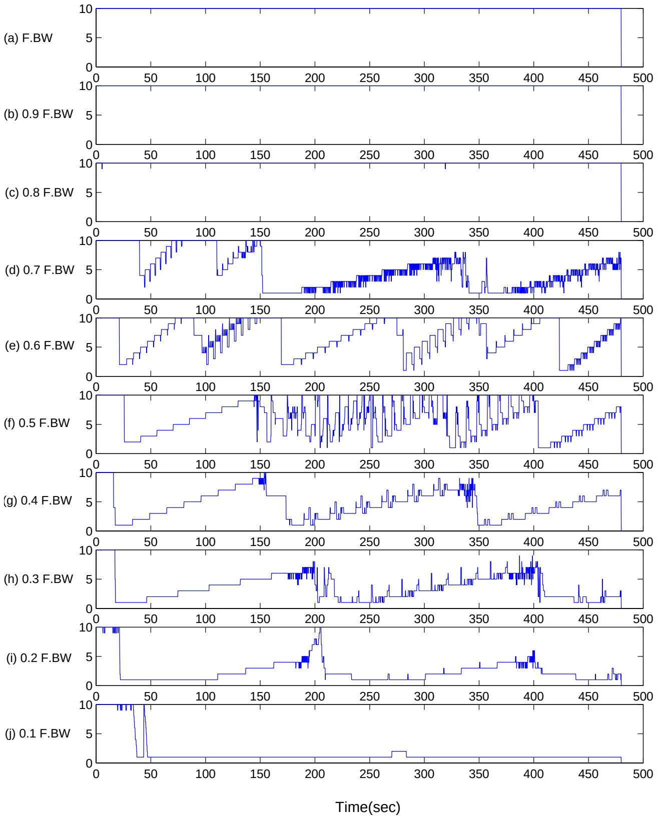

Figure 4: The transfer level changes according to the available link bandwidth. The X axis is time and Y axis is transfer level. The F. BW stands for required full bandwidth of 64 Mbps for transmitting DRS without packet loss. The measurements were conducted for 480 sec, in which eight sweep data were transmitted.

Cho, Y.-G., V. Chandrasekar, A. P. Jayasumana, and D. Brunkow: 2002, Architecture and implementation for high bandwidth real-time radar signal transmission and computing application. *Proceedings of SPIE*, volume 4863, 87–98.

Lee, K.-W., T.-E. Kim, and V. Bharghavan: 2001, A comparison of end-to-end congestion control algorithms: The case of AIMD and AIPD. *INFOCOM 2001*.

Mahdavi, J. and S. Floyd: 1997, Tcp-friendly unicast rate-based flow control.
URL `http://www.psc.edu/networking/papers/tcp\_friendly.html`

Widmer, J., R. Denda, and M. Mauve: 2001, A survey on TCP-Friendly congestion control. *IEEE Network*, 28–37.