A Cost Sensitive Best Network Selection Scheme in Heterogeneous Wireless Networks.................................1753
Feng He and Furong Wang

Fast Handover Scheme for Supporting Network Mobility in IEEE 802.16e BWA System.................................1757
Lei Zhong, Fuqiang Liu, Xin hong Wang and Yusheng Ji

GCQCP Based Vertical Handoff Scheme for Heterogeneous Hierarchical Wireless Networks..........................1761
Guoqin Ning, Jing Zhang, Gan Liu and Guangxi Zhu

Handoff Strategy Analysis by Stackelberg Model Over HSDPA .................................................................1766
Qian Jun, Yanping Yu, Kuang Wang and Guangxin Zhu

Enhanced Application-Driven Vertical Handoff Decision Scheme for 4G Wireless Networks..........................1771
Xin Guan, Rongxin Tang, Songnan Bai and Dongweon Yoon

The Algorithm Design of Agent for Detecting and Analyzing Data in Intrusion Detection Based
on Immune Principle........................................................................................................................................1779
Zhongmin Chen, Yu Wang and Baowen Xu

Joint Channel Assignment and Transmission Scheduling for Throughput Optimization
in Wireless Mesh Networks -Further Study on A Partitioning Approach .........................................................1784
Cheng Wang and Changjun Jiang

Delay Performance Analyses for GBN-ARQ and SR-ARQ Protocols ............................................................1788
Suo-Ping Li and Cun-Ming Liu

Route Optimization Solution Based On Extended Prefix Information Option For Nested Mobility Network ......1792
Lu LiHua and Liu YuanAn

Protocol Independent Distributed Message Copying Multicast Scheme for the Wireless Network ..................1797
Hong Yao, Yonghua Zhu, Lin Li, Zhihong Dong, Weimin Xu and Jianyong Yang

A Data Link Layer Protocol Based on Wireless Industrial Control Network Model-
Visual Token Control Protocol WICN-Z ............................................................................................................1801
Biao Shi, Zhen-Jun Tang, Xian-Hua Li, Ling-Feng Zhang and Xin-Hua Yu

Performance Comparison of Mobile IPv6 and its Extensions ......................................................................1805
Zailong Zhang, Jun Fang, Wuxia Wang and Shunyi Zhang

SIP for Mobile Networks and Security Model ..................................................................................................1809
Huaxu Wan, Guiping Su and Hongyan Ma

Enhanced Wireless TCP for Satellite Networks ...............................................................................................1813
Lin Cui, Seok Koh and Xin Cui

Stability-Guaranteed Clustering in Satellite Networks ......................................................................................1817
Dong-Ni Li, Xin Wang and Da-Kun Zhang

A New Way Based on Relative Forward Delay to Improve TCP Performance
Over Heterogeneous Networks ..........................................................................................................................1821
Dapeng Qu, Jiasheng Xue and Tiesheng Fan

The New Label Design for Hierarchical MPLS Network Based on Uni-interface ...........................................1825
Ming Cao, Xinmeng Chen and Xingchuan Yuan

New Analysis of BAN-Yahalom Protocol by Strand Spaces and its Improvement ...........................................1829
Yu Han and Hengshan Lv
Enhanced Wireless TCP for Satellite Networks

Lin Cui, Seok J. Koh
Department of Computer Science
Kyungpook National University
Daegu, Korea
{cuilin, sjkoh}@cs.knu.ac.kr

Xin Cui
Business School
Shandong University at Weihai
Weihai, China
chgms@sdu.edu.cn

Abstract—Satellite link is generally featured by long propagation delay, large bandwidth-delay product (BDP) and high bit error rate (BER). This tends to make the conventional TCP suffer from performance degradation in satellite networks. Thus Forward Error Correction (FEC) scheme is suggested for error recovery over satellite channel and RFC 2488 suggests that TCP does not reduce its congestion window (cwnd) for corruption. However, FEC decoder directly discards any unrecovered corrupted packets at the output. This cannot let TCP to use some useful feedbacks from the lower layers for improving its performance. This paper proposes an enhanced congestion control scheme, together with a handshaking mechanism between FEC and TCP, to improve goodput performance of TCP over satellite links. Simulation results show that the proposed scheme could achieve a better goodput performance, compared to the combination of conventional FEC and the existing TCP.

Keywords—satellite, FEC, TCP, BER, algorithm

I. INTRODUCTION (HEADING 1)

Satellite link is generally featured by long propagation delay, large bandwidth-delay product and high bit error rate. At present, the two types of satellites, Geostationary Earth Orbit (GEO) and Low Earth Orbit (LEO), are mostly used in the satellite networks. The one-way transmission delay is about 250 ms for the GEO satellite and 10–100 ms for the LEO satellite [1].

It is noted that the typical bit error rates of satellite links are ranged from 10^{-6} to 10^{-4}, which are much higher than those of terrestrial links [2]. Accordingly, the conventional Transmission Control Protocol (TCP) tends to suffer from performance degradation in satellite networks, since any corruption event is regarded as an indication of network congestion and the congestion window (cwnd) will be reduced in the same way. However, we note that for the packet corruption, other than packet loss (by network congestion), TCP does not need to reduce its cwnd in satellite networks [3].

On the other hand, to improve the throughput of TCP over satellite links, Forward Error Correction (FEC) scheme has been suggested for error recovery over satellite channels [3], while TCP makes some enhancements at transport layer (e.g., TCP-Peach [4], TCP-SACK [5], I-TCP [6], etc).

The drawback of FEC is that it consumes some extra bandwidth to transmit the redundant information. Therefore, increasing the level of FEC redundancy has dual effects on TCP throughput performance. On one hand, it increases the achievable TCP throughput since the throughput is a monotonic increasing function of the level of FEC redundancy. On the other hand, it decreases the effective channel bandwidth because the effective channel bandwidth is opposite to the level. Thus TCP throughput should be maximized when the effective channel bandwidth becomes equal to the achievable TCP throughput. The studies in [7] [8] also show that there is an optimal amount of redundancy to add, above which the end-to-end performance degrades instead of improving.

This paper proposes a new TCP congestion control scheme in cooperation with FEC mechanism to further improve the TCP throughput over satellite networks, which is based on a corruption-aware Adaptive Increase and Adaptive Decrease algorithms for congestion control and needs a slight revision for the interface of the existing FEC. Compared to the Additive Increase Multiplicative Decrease (AIMD) algorithms, the proposed scheme will further inflate the available channel bandwidth even in the face of packet corruptions if only there is no unrecovered lost segments until then. Also, in the proposed scheme the TCP sender will estimate the available bandwidth based on the amount of integral segments as well as corrupted segments since the corrupted segments also arrive at the receiver side.

The rest of this paper is organized as follows. Section II describes the proposed scheme in detail. Section III shows some simulation results using the ns-2 networks simulator and Section IV concludes this paper.

II. PROPOSED SCHEME

A. Handshaking between FEC and TCP

Generally, link layer FEC encoder fragments a frame packet into several codewords and sends redundant information along with the original data so that the lost/corrupted original data can be recovered, at least in part, from the redundant information without retransmission.

The significant FEC example for satellite networks is BCH code. BCH encoder encodes k data bits into n-bit codeword by adding n-k parity checking bits for the purposes of checking and recovering the errors. Given the length of a BCH codeword is n=2^m-1 for any integer m≥3, we will have t (where t<2^m-1 and n-k≤mt) that is the bound of the error correction. That is, BCH decoder can correct any combination of errors (burst or separate) fewer than t in the
n-bit BCH codeword; otherwise the whole frame packet won’t be reassembled at the output of FEC.

In this paper, however, it is supposed to classify the FEC codewords into two classes, one is header class and the other is data class. All codewords that compose the packet header (e.g. IP header and TCP header) are classified into the header class, and the others are classified into the data class. For the codewords of the header class, the ones with more than 7 errors will be directly discarded in the same way, together with the other codewords belonging to the same frame (as does in existing FEC), since IP header or TCP header may contain some wrong information. Nevertheless, for those codewords of data class, even if they cannot be correctly recovered, the FEC decoder should still assemble a fix-sized packet that has an integrated header and the corrupted payload and still deliver them to the transport layer of the receiver.

The handshaking interactions between TCP/IP stacks and FEC mechanism can be implemented by using two unused bits in TCP header, of which one is for the TCP/IP stacks of the sender to set an “additional header checking request” flag in each sending packet and the other is for the FEC decoder to set an “integrated header” flag in each corrupted packet which has an integrated header. As for the completely integrated packets, the FEC decoder will erase both flags for them. Thus the additional flags cannot influence the verification of the existing TCP overall checksum that resides at the TCP header.

Therefore, the proposed scheme can be performed as an option of the existing TCP and can be compatible very well with it.

B. TCP Option for loss and corruption

On reception of a TCP segment, the receiver will first verify the two additional flags, namely “additional header checking request” flag and “integrated header” flag. Once neither of them is set, the receiver will perform as normal. Whereas if both are set, that means the packet header has been verified by FEC decoder before reassembling this packet and the payload contains some bit errors. Except these two cases, the whole segment will be discarded without further processing.

As a response to an unrecovered lost or corrupted packet, in the proposed scheme every duplicate ACK (DUPACK) will carry an additional option, as shown in Fig.1.

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length=?</th>
<th>32-bit corrupted segment’s sequence number</th>
<th>Congestion flag</th>
</tr>
</thead>
</table>

Figure 1. Proposed TCP option for loss and corruption

where the 32-bit sequence number field will be set to a certain number once a corrupted packet is indicated by both additional flags (the available sequence number will be extracted from the integrated header). Also, if there is no unrecovered 'lost segment' (but not corrupted) until then, the congestion flag field will keep empty. In other words, any DUPACK with non-empty congestion flag field implies that some packet losses may happen and the fast recovery algorithm should be invoked at the sender side.

Notice that in a certain case without any packet loss, the sequence number field and congestion flag field might be all empty. In the other case with the burst of packet loss and corruption in a row, the both fields could be non-empty.

C. Enhanced Congestion Control

Based on the additional options contained in the DUPACKs, the TCP sender can more precisely estimates the current corruption strength (α) and the available end-to-end network bandwidth (BWE) by counting the amount of corrupted data, Dk, as well as the amount of integral data, Dk

During every RTT interval, using the following formulas, respectively:

\[ \alpha = \frac{D_k}{D_k + C_k} \]  
\[ b_{\text{we}} = \frac{D_k + C_k}{RTT_k} \]  
\[ BWE = \frac{2r-t_s}{2r-t_s + t_k} \times BWE_{k-1} + t_k \times b_{\text{we}} + b_{\text{we}_{k-1}} \]

where 1/τ is the filter cut-off frequency (a typical value of τ is 0.5s), and τ is the recent ACK’s interval. It is noted that the formula (3) was already described in the existing Westwood scheme [9], whereas the other two formulas are newly proposed in this paper.

Based on the estimations, the proposed congestion control algorithm can be described as below:

When the current RTT is larger than the minimum RTT: reestimate α by formular (1); reestimate BWE by formular (2) and (3); reset C and D to zero;

When an ACK/DUPACK arrives at the sender:

For a normal ACK without the additional option (Case 1): reset the counter of DUPACK to zero; update the amount of the integral data (D); apply the normal additive increase algorithm (α=1);

For a DUPACK which carries an additional option with non-empty congestion flag field (Case 2): if (the sequence number field is not empty) { retransmit the indicated data segment immediately; update the amount of corrupted data (Ck) by Ck + MSS; } else { update the amount of integral data (D) by Dk + MSS; } increase the counter of DUPACK by 1; if (the counter of DUPACK = 3) and (the first missing segment is not the corrupted one)) { retransmit the missing segment; apply the adaptive decrease algorithm of TCP Westwood+ [10] as follows: ssthresh = (BWE*RTT) / MSS; if (cwnd > ssthresh) cwnd = ssthresh;
if ((the counter of DUPACK = 3) and (the first missing segment is the corrupted one)) { 
    reset the counter of DUPACK to zero;
}

For a DUPACK which carries an additional option with empty congestion flag field (Case 3):
If (the sequence number field is not empty) {
    retransmit the indicated data segment immediately;
    update the amount of corrupted data ($C_k$) by $C_k + MSS$;
} else {
    update the amount of integral data ($D_k$) by $D_k + MSS$;
}
reset the counter of DUPACK to zero;
apply the proposed adaptive increase algorithm ($\alpha < 1$):
if ($cwnd < ssthresh$) 
    $cwnd = cwnd + \alpha$;
else 
    $cwnd = cwnd + \alpha / cwnd$;

III. SIMULATION RESULTS

Using the ns-2 simulator [11], we evaluate the performance of the proposed scheme by performing the file transfer application for 100 seconds and calculating the average available goodput that is based on the acknowledged maximum in-order sequence number. The simulation topology is as follows:

![Simulation topology](image)

In the figure, three connections are established through the same edge routers, terminals and GEO satellite, where each satellite link has the capacity of 2Mbps and default propagation delay (default RTT is about 530ms or so), while other links have the bandwidth of 10Mbps and a propagation delay of 2ms. Queue size of each intermediate node is set to 512 packets and each packet has a fixed size of 1040 bytes. Furthermore, we assume that the BCH code of FEC mechanism is employed over the link layer and satellite channel will be error-free while BCH (255, 87, 26) code is applied. Notice that a BCH (255, 87, 26) codeword consists of 87 data bits and 168 parity bits (approximately a goodput rate of 1/3), and can correct up to 26 errors in a 255-bit codeword.

Since the BCH (255, 87, 26) code approximates a goodput rate of 1/3, thus the available data goodput should also approximate 1/3 of the total throughput. Simulation results show that as satellite channel is error-free, the average throughput can reach 582.1 Kbps per connection without considering overhead and the delay used for fragmenting and reassembling link-layer frames as well as FEC codewords, and thus the available data goodput should approximate 198.6 Kbps for each connection.

Instead of the BCH (255, 87, 26) code, if we apply a BCH code with a lower error correction rate, e.g., BCH (255, 131, 18) code, over satellite channel and leave some residual errors for the proposed congestion control scheme to recover in an end-to-end manner, we might achieve a higher goodput under the same channel conditions. Simulation results also show that the combination of a lower error correction rate’s BCH code and the proposed scheme could be a better choice.

Figure 3 depicts the traces of the available average goodputs and residual packet corruption rates for various BCH codes in cooperation with the proposed congestion control scheme.

![Figure 3. For the proposed scheme with various BCH codes](image)

From the figure, we can see that in cooperation with the lower error correction rates’ BCH codes, the proposed scheme can achieve the higher goodputs than the combination of the traditional TCP and the higher error correction rate’s BCH code. In particular, when applying the BCH (255, 87, 26) code under the traditional TCP, the maximum average goodput only approximates 198.6 Kbps even if the satellite channels are error-free. On the contrary, when the BCH (255, 91, 25) code is performed under the proposed scheme, the average goodput could exceed the maximum goodput of the conventional case if only the residual packet corruption rate is smaller than 4% or so. Likewise, if applying the BCH (255, 131, 18) code instead of the one of BCH (255, 91, 25), the performance gain could be expected with the residual packet corruption rate lower than 12%.

However, the same approach cannot be simply repeated by other TCP variants. As an opposite example, we also depict the same traces of the available average goodputs and residual packet corruption rates for various BCH codes in cooperation with Westwood+ TCP, as shown in Fig. 4.
In this paper, we present a corruption-aware adaptive increase and adaptive decrease algorithm based TCP congestion control scheme for satellite networks, together with a handshaking mechanism between FEC and TCP. Our first contribution is estimating the available network bandwidth based on the amount of integral data as well as the amount of corrupted data as per RTT interval since the corrupted packets also arrive at TCP receiver. Further, sssthresh is only reset when lost but not corrupted segment occurs. Our second contribution is developing an adaptive increase and adaptive decrease algorithm by which the sender does not stop increasing cwnd if only there is no unrecovered lost but not corrupted packet until then. Our third contribution is suggesting an effective handshaking mechanism between FEC and TCP so that TCP can use the useful feedbacks from the lower layers to improve its performance over satellite networks.

Simulation results show that, in cooperation with an appropriate BCH code of FEC mechanism, the proposed scheme could achieve a better goodput performance than the combination of FEC and the existing TCP. Such performance gain is anticipated since the proposed scheme can estimate the network bandwidth more precisely and still increases its cwnd if only there is no packet loss to be detected, thus it can more effectively utilize the satellite channel.

ACKNOWLEDGMENT

This research was supported by the MIC, Korea, under the ITRC support program supervised by the IITA (IITA-2006-C1090-0603-0026).

REFERENCES