ATCS: An Adaptive TCP Coding Scheme for Satellite IP Networks

Wei Dong[1], Junfeng Wang[1], Minhuan Huang[2], Jian Tang[2] and Hongxia Zhou[3]
[1]College of Computer Science, Sichuan University
Chengdu, Sichuan 610064 – China
[e-mail: brant.dong123@gmail.com, wangjf@scu.edu.cn]
[2]National Key Laboratory of Science and Technology on Information System Security,
Beijing Institute of System Engineering
Beijing, 100101 – China
[e-mail: hminwell@gmail.com, jian_tang@126.com]
[3]Sichuan Military District
Chengdu, Sichuan 610015 – China
[e-mail : hx_zhou999@yahoo.com.cn]
*Corresponding author: Junfeng Wang

Received February 3, 2011; revised March 20, 2011; accepted April 18, 2011; published May 31, 2011

Abstract

In this paper we propose ATCS, a practical TCP protocol coding scheme based on network coding for satellite IP networks. The proposal is specially designed to enhance TCP performance over satellite networks. In our scheme, the source introduces a degree of redundancy and transmits a random linear combination of TCP packets. Since the redundant packets are utilized to mask packet loss over satellite links, the degree of redundancy is determined by the link error rates. Through a simple and effective method, ATCS estimates link error rates in real time and then dynamically adjusts the redundant factor. Consequently, ATCS is adaptable to a wide range of link error rates by coding TCP segments with a flexible redundancy factor. Furthermore, the scheme is compatible with traditional TCP variants. Simulation results indicate that the proposal improves TCP performance considerably.

Keywords: Adaptive, network coding, TCP, satellite IP networks

This work is supported by the National Basic Research Program of China (2007CB310702); the National High-Tech Research and Development Plan of China (Grant No. 2008AA01Z208); the Doctoral Fund of Ministry of Education of China (20090181110053); the Applied Basic Research Program of Sichuan Province of China (Grant No. 2010JY0013); the Youth Foundation of Sichuan Province (09ZQ026-028).

DOI: 10.3837/tiis.2011.05.008

1. Introduction

Since the introduction of network coding in the original paper [1], the concept has attracted a lot of interests in recent years. Basically, the key concept is to allow intermediate nodes to encode the received packets before transmitting them, instead of just forwarding them to the next hop with the information nearly untouched. In the original paper [1] the authors showed the capacity gain of network coding for multicast in wired networks. Thereafter, network coding has been applied to wireless networks and receives significant attention as a means of improving network capacity and coping with unreliable wireless links [2][3]. In fact, the unreliability and broadcast nature of wireless links make wireless networks a natural setting for network coding. Many research papers [4][5][6][7][8][9] adopt this nature and apply network coding in wireless network environment.

In spite of many research papers on the application of network coding in wireless networks, surprisingly, there are not many real practical implementations. Therefore, it is not well understood to what extent network coding improves the throughput capacity of a wireless network in a real implementation. This paper aims to characterize and quantify such throughput improvements in satellite networks employing network coding. Since TCP is expected to be the dominant transport protocol in Stateline IP networks, we focus on TCP traffic and hence TCP throughput improvement due to network coding. It is well-known that conventional TCP protocols suffer from throughput performance degradation in satellite networks seriously due to the high link error rate, long propagation delay, and large bandwidth-delay product (BDP) of satellite channels [10]. Traditionally, those protocols designed for more reliable wired networks interpret any phenomenon of packet loss as network congestion and do not take the link error rate into account. Since the reliability benefit of network coding [11] is so obvious, our idea is to integrate network coding into TCP protocol. By doing so we expect that network coding improves the network reliability by reducing the number of packet retransmissions in lossy satellite IP networks and consequently improves the throughput of TCP protocol. However, there are some challenges to be faced to integrate network coding into TCP protocol. First, we must get an opportunity to combine packets. To deploy network coding, a number of packets (the number of packets is dependent on the coding scheme) must be collected, which means that some packets must experience some delay before transmission. Because TCP is a connection-oriented protocol, the delay parameter is non-trivial and either too long or too short is not appropriate. If it is too long, it means some packets must wait a long time before transmission and the other end of TCP probably thinks the packet is lost and then unnecessary retransmission is incurred. However, if it is too short, we probably can't get enough TCP packets to encode. Second, the degree of packet redundancy must be evaluated carefully. To cope with lossy satellite links, redundancy must be introduced and the redundant degree should be consistent with the packet loss rates. Third, protocol compatibility should be considered. To deploy network coding into network stack in an incremental manner, protocol compatibility is an important issue.

In this paper, we propose an Adaptive TCP Coding Scheme (ATCS) to improve transmission performance of TCP over lossy satellite links. The main idea of ATCS is that it employs network coding and introduces packet redundancy to mask packet loss from link error. Firstly, the new scheme measures the packet loss rate in real time and then dynamically adjusts its redundant degree accordingly. Consequently, with this flexible redundant factor, the scheme is adaptable to a wide range of link error rates while keeping the redundancy degree at an optimum level. Furthermore, ATCS doesn't assume any traffic pattern and is compatible with traditional TCP protocols.

The remaining of this paper is organized as follows. In Section 2, some related work is briefly described. In Section 3, ATCS is firstly presented and then its implementation is elaborated in details. In Section 4, the performance of ATCS is evaluated through simulation. Finally, we summarize the protocol in Section 5.

2. Related work

In recent studies, network coding has been demonstrated to be helpful with regard to network throughput in wireless networks [7][12]. With randomized network coding approach employed, Ho et al. [13][14] presented their first theoretical studies on achievable flow rates and code assignment algorithms. Based on the result, many researches concentrate on the practical application of network coding in a practical setting, with an emphasis on the wireless network environment. The systematic research on network coding conducted by Chou et al. [8] drew a conclusion that randomized network coding can be designed to be robust to random packet loss, delay, as well as any changes in network topology and capacity.

At the same time, the reliability benefit in both wired networks and wireless networks is extensively studied. In general, network coding improves network reliability by reducing the number of packets transmission in lossy networks. In [15], the authors characterized the reliability benefit of network coding for reliable multicasting, and the extent of the reliability benefit of network coding was derived. In [16], a distributed random linear network coding approach for transmission and compression of information in general multisource multicast networks was proposed, and the authors concluded that the approach can take advantage of redundant network capacity for improved success probability and robustness.

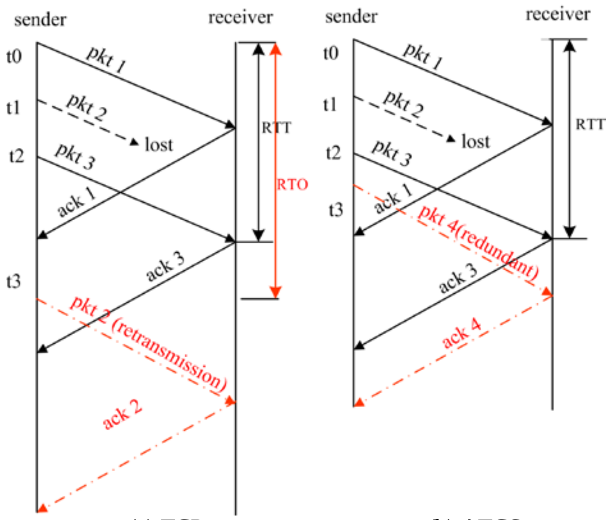
While most researchers concentrate their researches on multicast and broastcast traffic, it has been considered the possibility of deploying network coding solutions for unicast traffic scenarios [2][7][17]. The first practical approaches to network coding for multiple unicast data flows are proposed in [2] and [7]. In [2], the authors proposed a network coding scheme to perform information exchange between two different nodes in a line topology. In [7] Katabi et al. introduced COPE, a practical network coding scheme that operates above the MAC layer and performs packet coding operations according to an "opportunistic" mechanism. In [17] the authors provided a theoretical analysis of the unicast information distribution.

Although the capacity gain and reliability gain are well-known, there are few studies on the practicality of using network coding in TCP protocols. In [18], the authors proposed a mechanism that the sink acknowledges every degree of freedom (i.e., a linear combination that reveals one unit of new information) even if it does not reveal an original packet immediately. However, the new interpretation of acknowledges (ACKs) is not compatible with current TCP protocol.

3. Adaptive TCP Coding Scheme (ATCS)

3.1 The principle of ATCS

The principle concept behind the proposed scheme is the introduction of information redundancy mechanism through a linear network coding approach [19][20] over TCP segments. By employing information redundancy, we expect that packet losses can be masked with network coding. Consequently, retransmissions incurred by packet losses are decreased and network throughput is improved. This idea is illustrated in **Fig. 1**. Suppose the sender transmits three packets to the receiver and one packet will be lost druing transmission. As can be seen from **Fig. 1-(a)**, traditional TCP protocol adopts a reactive approach to cope with packet losses. At time *t0*, three successive TCP packets are sent from the source and packet 2 is lost during transmission. However, it will take some time for TCP protocol to notice the packet loss and the amount of time depends on the RTT measurement. Compared with traditional TCP protocols, ATCS employ a proactive method to cope with link errors, which is demonstrated in **Fig. 1-(b)**. When the source send three successive TCP packets, ATCS applies network coding to the sending TCP packets and linearly combines the sending packets, yielding four network coding packets with one redundant packet included. Then the four encoded packets are sent from the source. If one packet is lost during transmission, we can regain the original three TCP packets because all other packets are linearly correlated and the retransmission is totally unnecessary. Note that in **Fig. 1-(b)** we assume that packet 2 is lost. Actually, ATCS can regain the three original TCP packets from any single packet loss. With network coding, ATCS is expected to reduce packet retransmission and thus decreases packet delays and consequently improves network throughput. As can be seen from **Fig. 1**, both ATCS and TCP will need to transmit four packets for information completeness. So from this point of view, ATCS doesn't incur real redundancy, which will be demonstrated in Section 4 by simulation.



(a) TCP (b) ATCS

Fig. 1. Transmissions paradigm TCP vs ATCS

Although TCP with network coding can provide above-mentioned nice property, there are some issues to cope with. First, the redundancy degree must be carefully designed. Redundancy degree is an important factor and either too little or too much redundancy is not appropriate. If there is too little redundancy, the receiver probably can not get enough information to decode the encoded network-coding (NC) packets, i.e., the redundancy is not large enough to cope with link errors. For instance, there are four original TCP packets to be sending and ATCS introduces a redundant packet. Suppose two packets are lost during transmission. Then in the receiving side, we receive only three NC packets and the information is not enough to decode the original packets. This leads to a situation where packet losses are not masked from the TCP layer. On the other hand, too much redundancy is not good because the transmission rate becomes limited by the rate of the coding itself. The general guideline is that if there are *N* TCP packets to be sent in the sending side and there are maybe *R* packets to be lost during transmission, for the receiver in order to correctly decode the original information *R* redundant packets must be introduced to cope with link errors. It means that we must introduce enough packets to cope with link error rates. ATCS adopts an intelligent approach and dynamically adjusts its redundancy degree factor according to real-time link error rate measurements.

To employ redundancy to compensate for the packet losses, we assume that the sending rate is lower than the capacity of links and the buffers within end systems are large enough so that no packet losses caused by buffer overflow. So link error is the major factor in packet losses. ATCS estimates packet loss rate by following Equation (1),

$$P_{esti-loss} = \alpha \cdot n_{retrans} / n_{total} \tag{1}$$

where $n_{retrans}$ and n_{total} denote the retransmitted packets and total transmitted packets respectively, and α is a tunable factor which indicates the percentage of retransmission caused by packet loss. Then the redundant parameters can be determined by the packet loss estimation accordingly.

A new layer (i.e., network coding layer, NC Layer) is incorporated into existing protocol stack to facilitate TCP packet coding scheme. As shown in **Fig. 2**, the NC layer is situated between TCP and IP layers. The utility of NC layer is to encode TCP packets into NC packets in the sending side and decode TCP packets from NC packets in the receiving side.

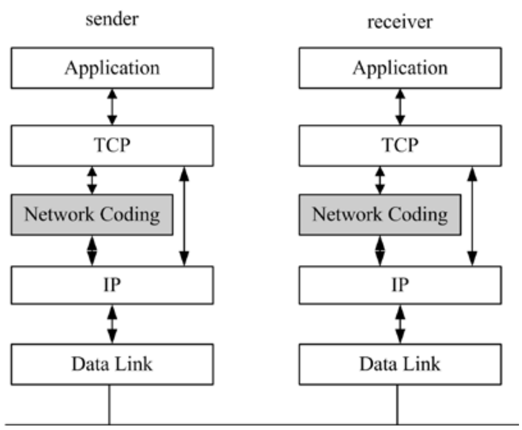


Fig. 2. New network protocol stack with network coding

To facilitate following discussions, **Table 1** which consists of some terminologies is first introduced. The encoding process of ATCS can be described by the following two steps.

STEP 1: Opportunity listening

In this step, the sender listens for traffic from upper layer (i.e., the TCP layer). When a packet arrives, instead of sending it directly to IP layer, NC layer preserves the packet into a buffer. An encoding group which consists of *N* TCP packets must be formed before staring coding process.

STEP 2: Network coding

Upon obtaining a group of TCP packets, random linear code is utilized to combine them [20]. ATCS linearly combines the packets by following Equation (2),

$$\left(y_1 \cdots y_M\right) = \left(x_1 \cdots x_N\right)\begin{pmatrix} a_{11} & \cdots & a_{1M} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NM} \end{pmatrix} \tag{2}$$

where $x_i (1 \le i \le N)$ and $y_i (1 \le i \le M)$ are the original TCP packet and the encoded NC packet respectively, and $a_{ij} (1 \le i \le N, 1 \le j \le M)$ is the encoding coefficient from a finite field [21]. By utilizing encoding scheme denoted by Equation (2), the NC layer generates M ($M \ge N$) correlative packets. The difference between M and N is defined as redundant factor R (i.e., R=M-N). Then the NC layer encapsulates the encoded packets with a NC packet header and forwards them to IP layer. Note that the redundant factor R can be zero when the estimated link error rate is zero. However, even in this case, ATCS has a unique merit of security that traditional TCP protocols don't possess. There exist several security vulnerabilities in TCP protocols [22]. However, as far as security is concerned, ATCS is more robust than traditional TCP protocols since ATCS employ network coding to linearly combine the TCP packets before sending them and a single packet alone does not reveal real information.

Note that ATCS does not assume any specific traffic pattern. However, the traffic flow may be too light to collect a group of packets in a short time. To make the algorithm adaptable to all kinds of traffic flow, each time a packet joining the encoding buffer list, a timer with a period of time T is started (the tuning of the T parameter is discussed later). If the timer expires, the related packet is removed from the list and sent directly to the IP layer.

Table 1. Terms and their meanings for Network coding

Terms	Descriptions
NC group	A group TCP packets that be linearly combined.
N	The number of packets of a NC group before coding.
M	The number of packets of a NC group after coding. ($M \ge N$)
R	The redundant packet number of a NC group, $R = M - N$.

The packet decoding is the reversal process of packet encoding. For the receiver the received packet can be reconstructed by Equation (3),

$$y_i = \sum_{j=0}^{N} a_{ij} x_j \tag{3}$$

which means that every receiving packet is a linear combination of the sending packets of the group. Because of packet redundancy, a packet loss does not affect the whole decoding process (more precisely, R packet losses don't affect the decoding process). The decoding function is given by Equation (4),

$$\left(x_1 \cdots x_N\right) = \left(y_1 \cdots y_N\right)\begin{pmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{pmatrix}^{-1} \tag{4}$$

where $x_i (1 \le i \le N)$ and $y_i (1 \le i \le N)$ are the decoded TCP packet and received NC packet. When decoding action is done, the decoded packets, i.e., TCP packets, are dispatched to TCP layer for further processing as that in conventional TCP protocols.

3.2 ATCS implementation

The implementation of all the above-mentioned ideas in the existing protocol stack needs to be done in as compatible a manner as possible. We present a solution which embeds the network coding operations in a separate layer between TCP and IP on the source and receiver side. The exact operation of these modules and some related discussions are described as follows.

3.2.1 Packet format

In our implementation, a brand-new packet format (i.e., NC packet format) which is easily distinguishable from TCP packet format is designed, and the NC header is shown in **Fig. 3**. The packet fields and their meanings are shown in **Table 2**. Without losing its generalization, following algorithms assume a one-way TCP flow.

As can be seen from **Fig. 3**, the NC packet header has an overhead of 24 bits: 8-bit NC group ID, 4-bit original packet number, 4-bit redundant packet number and 8-bit encoding coefficient vector. However, the packet header overhead can be ignored compared with the traffic transmitted, which will be demonstrated in the simulation section.

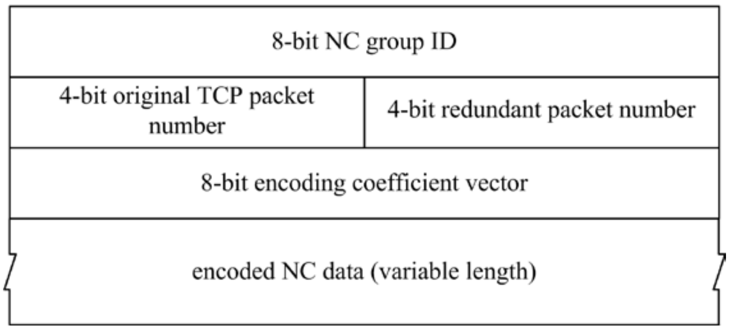


Fig. 3. Network coding packet header

Table 2. NC pakcet header fields and their meanings

Terms	Descriptions
NC group ID	The ID of a NC group.
Original Packet Number	The packet number of the encoded NC group.
Redundant packet number	The redundant packet number utilized to cope with link error.
Encoding coefficient vector	The encoding coefficient vector used for network coding.
Encoded NC data	The NC packets data after encoding.

3.2.2 The choice of tuning parameters in ATCS

In ATCS, some tuning parameters are introduced. Since these parameters are very important to protocol performance, we discuss the choice of these parmeters in this section. The discussion is divided into two parts. In the first part, we discuss how we choose the timer for buffered packets and the reason for such choice is justified. In part two, other tuning parameters (including the number of buffered packet, redundancy degree) are discussed, and how these values are determined in this study is also elaborated in details.

Integrating network coding into TCP protocol is challenging, because TCP is a time-critical protocol. If some delay during transmission is introduced and the receiver doesn't acknowledge it timely, the sender may deem that the packet is lost or some error occurs, and retransmission algorithm will be triggered. However, due to its characteristics, the delay introduced by network coding is unavoidable. Actually, for other network coding based protocols the delay problem also exists. There are two reasons. First, in the sending side, a node must obtain enough opportunity to apply network coding. Consequently, some delay must be introduced to collect packets to deploy network coding. Second, in the receiving side, because information of several original packets is mixed up, enough packets must be obtained before the original packets are recovered from the coded packets. Thus, this waiting process will introduce some delay. For TCP, the delay is crucial to the performance. If the delay is too long, the sender may deem that the packet is lost during transmission and so unnecessary retransmission is triggered. On the other hand, if the delay is too short, the network opportunity will be little. So the timer for the buffered packets must be carefully chosen. In general, the basic principle is that the delay parameter should be chosen that the retransmission overhead incurred by the delay should be kept at a minimum level, while the network coding opportunity is fully employed to improve the TCP performance over lossy satellite links. We choose a fraction of current retransmission timeout (RTO) as the timer for buffered packets. The reason for this choice is explained as follows. First, it is well-known that the retransmission timer of TCP is closed related to RTO measurement. So the choice of a fraction of RTO time will not incur unnecessary retransmission. Moreover, the RTO measurement is a dynamic parameter of the network, and it reflects the network state in time. Second, satellite network is featured in its long propagation delay and its RTT has an order of about 100ms [23], and the RTO is a multiplicative of RTT. This period is enough for ATCS to collect enough packets to apply network coding provided the satellite links are fully utilized. Last, as we will seen later, the simulation results are promising to support our choice. In all simulations presented in this research, we choose 1/3 of current RTO time as the delay timer. Although the delay issue is an important one, it is surprising that this problem is never addressed in previous research, and we will investigate this issue more carefully in our future reasearch.

The redundant degree problem is also an important one and deserve to be examined in more details. The utility of redundant packets in ATCS is to cope with link error over satellite links. From this standpoint, the redundant degree must be big enough to cope with link error. So we have following equation (5).

$$\frac{R}{N} \geq P_{esti-loss} \tag{5}$$

On the other hand, the redundant packets should be kept at a minimum level because the redundant packets will devour the network bandwidth. Considering the redundant packet number R is an integer, so we have following equation (6).

$$R = \lfloor P_{esti-loss} \times N \rfloor \tag{6}$$

3.2.3 Sender side: network coding

The coding process is illustrated in **Fig. 4**. First, ATCS starts an encoding timer in order to collect N packets to deploy network coding. If the traffic is so heavy that ATCS can collect N packets in a very short time, the coding process is triggered and the encoded TCP packets (i.e., NC packets) are forwarded to IP layer for transmission. However, if the traffic is so light that ATCS doesn't have enough packets to code, the encoding timer will finally expire and the original packets are forwarded directly to IP layer.

Note that in our algorithm we draw a distinction between data packets and control packets. On receiving a control packet, instead of attempting to wait for more TCP packet to code, ATCS forwards it directly to IP layer for transmission. There are several reasons for doing this. First, control packets (such as the three handshaking packets which consist of SYN packet.) are more time-critical. So introducing a delay before transmitting them is not reasonable. Second, control packets (for instance, pure ACK packets) generally have a small size and the network coding price is too big. Compared with size of a control packet, the size of NC packet header can't be ignored. However, applying network coding to data packets is more economical. Furthermore, for TCP, control packets are rarely sent in a successive way in a short time, which means that ATCS does not have an opportunity to code. That's the last reason why TCP control packets are specially handled.

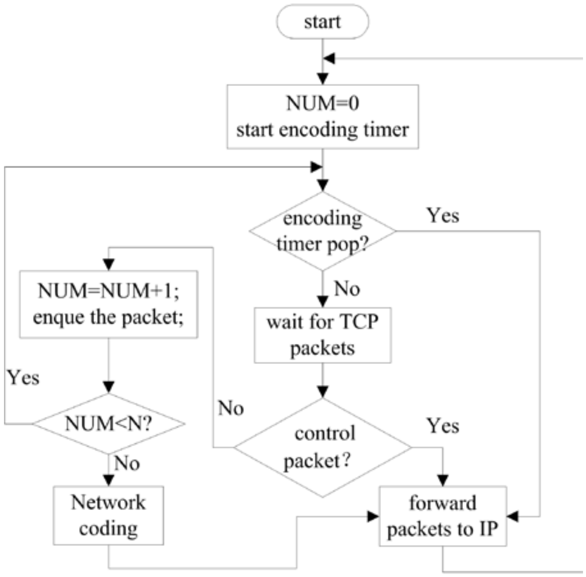


Fig. 4. Flowchart of network coding process

3.2.4 Receiver side: Network decoding

The flowchart of network decoding is shown in **Fig. 5**. On receiving a NC packet, ATCS will start decoding process. The information of NC packet header is utilized to decode the packet. When receiving a NC packet, the procedure enques the packet into the receiving buffer according its NC group ID. Then the packet number of buffer with the same group ID is checked to see whether the packet number is equal to *N*. If yes, the real network decoding procedure is triggered and the decoded TCP packets are then forwarded to TCP layer for further processing.

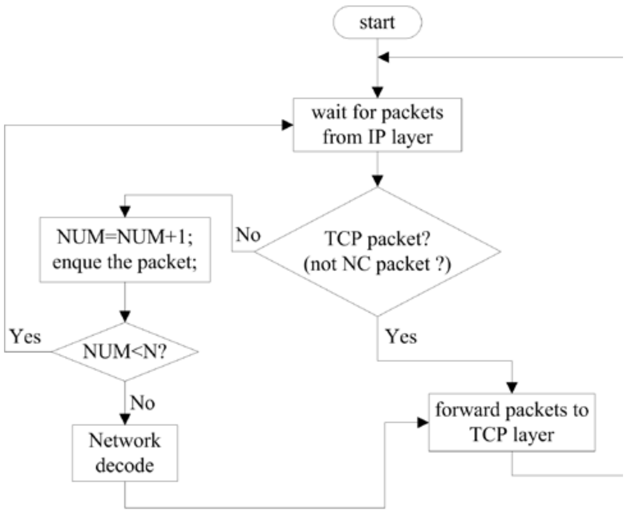


Fig. 5. Flowchart of network decoding process

For TCP, out of order issue is very important. If not carefully processed, it will lead to unnecessary retransmissions and even network congestions. However, for ATCS out of order is trivial. After decoding, the decoded TCP packets are firstly sorted by TCP sequence number and then forwarded to TCP layer in sequence.

Also note that only N packets, instead of M packets, are needed to decode, which is a nice property of ATCS. If one more packet with the same NC group ID arrives after the decode process, the packet will be discarded.

4. Simulation Results and Analysis

We have implemented ATCS in the network modeler OPNET. To validate our research, we compare the performance of ATCS and other several TCP variants by examine the performance of persist connections (i.e., long file transfers) over simulated satellite network topology. FTP is chosen as the simulation application. Three compared terms which included throughput, file download time, effectiveness of the protocol are employed to demonstrate the performance of ATCS and other TCP protocols. In general, we choose FTP application to better facilitate the evaluation of the performance of ATCS and other TCP protocols from three aspects (i.e., throughput performance, macroscopic delay behavior and effectiveness of protocol). The reasons for such choice are as follows. First, FTP is a TCP based application which is suitable for our research. Second, it is well-known that network coding based protocols are opportunistic protocols. To fully utilize the benefit of network coding, coding opportunity must be obtained. Consequently, to justify our research we are interested in comparing the performance of ATCS and other TCP flavors by examining the throughput performance of persistent connections over simulated satellite network topology, and FTP application is then a natural choice. With long file transferring between the client and the server, the throughput gain can be easily deduced. Third, as we mentioned above in this study, in general ATCS will reduce the flow delay by avoiding retransmission which is common in conventional TCP to cope with packet loss. So file download time in FTP is picked as the term to demonstrate the macroscopic delay performance of ATCS. Last, the effectiveness of all protocols is compared to show whether there is real redundancy involved in ATCS.

The simulation model is shown in **Fig. 6**. A FTP session is established between the client and the server through the GEO satellite. The FTP client requests a 2KB file periodically and the inter-request time is 0.5s. The sever responds with a segment length of 512bytes. All links have a capacity of 10Mbps. The propagation delay of satellite link and other link are 300ms and 5ms respectively. The simulation time is 120s. The buffer size is fixed to 64KB. The link is modeled with bit error rate (BER) ranges from 10^{-6} to 10^{-5}. Three classical TCP flavors (i.e., TCP New Reno, TCP Reno and TCP SACK) are compared with the ATCS in all simulation scenarios.

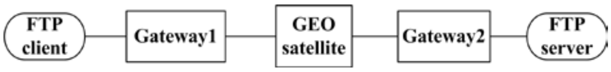
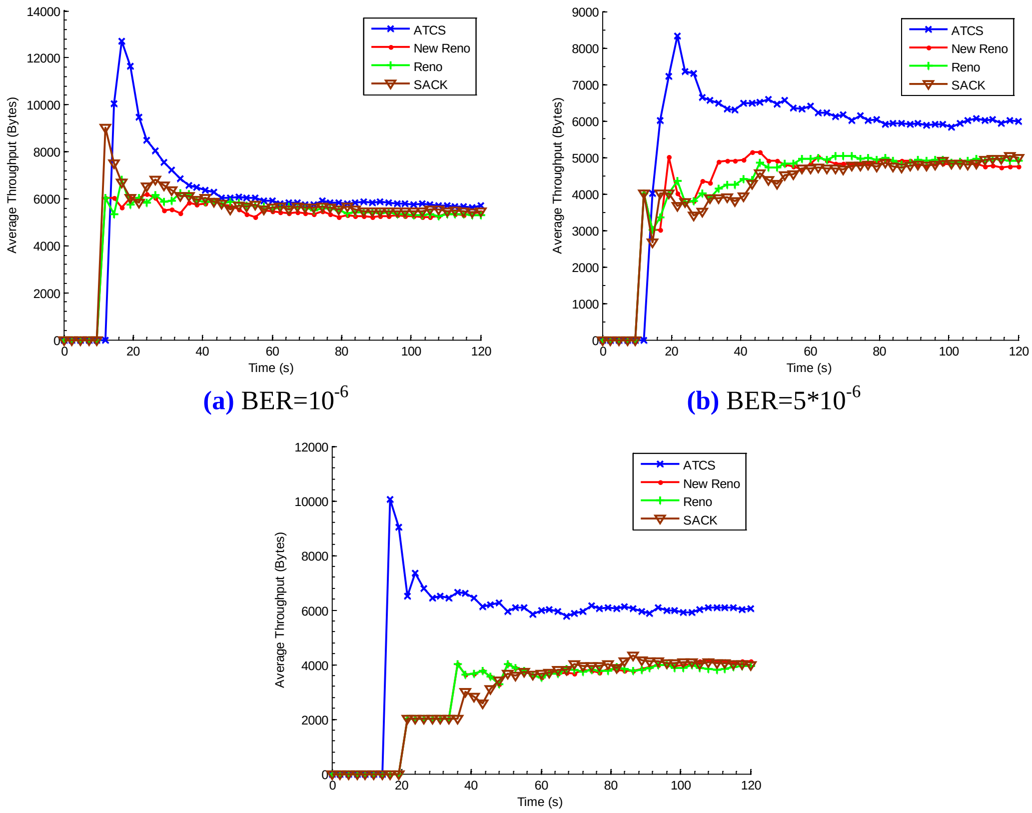


Fig. 6. Simulation model

Firstly, we examine the variation of throughput with loss rate for the three TCP variants and ATCS. **Fig. 7-(a)** displays the average throughput of all schemes when BER is 10^{-6} and ATCS has a light throughput gain of around 2%. With the degradation of link errors, the throughput performance of ATCS is more impressive. As demonstrated by **Fig. 7-(b)** and **Fig. 7-(c)**, ATCS achieves around 20% and 50% more throughputs respectively at $5*10^{-6}$ and 10^{-5} BER. Such results are anticipated since the proposal can mask packet loss through aforementioned mechanism. As the bit error rate increases, conventional TCP degrades dramatically and its throughput drops from 6000 to 5000, then to 4000 packets. However, the performance of ATCS is not affected greatly by BER and its throughput is rather stable.

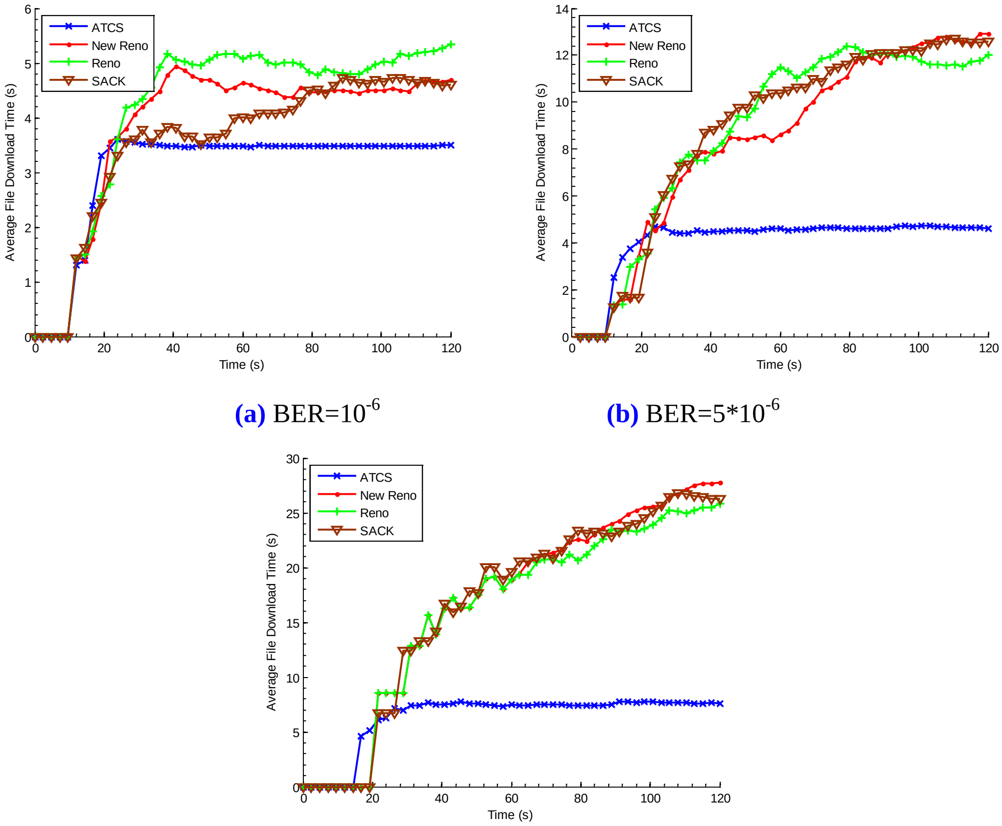


(a) BER=10^{-6} **(b)** BER=$5*10^{-6}$

(c) BER=10^{-5}

Fig. 7. Impact of BER on average throughput

Fig. 8 shows the file download time performance of ATCS and three TCP variants. At all error rate levels, the performance of ATCS is superior to other TCPs. It can be observed that as bit error rate level increases the file download time of TCP degrades dramatically while the performance of ATCS is more robust than that of other flavors.



(a) BER=10^{-6} **(b)** BER=$5*10^{-6}$

(c) BER=10^{-5}

Fig. 8. Impacts of BER on average file download time

As described before in this study, ATCS introduce redundant packets to cope with link errors. Although extra TCP packets are introduced by this mechanism, unlike other TCP flavors, there is almost no retransmission due to the proactive method ATCS employs. Consequently, we argue that there is maybe no real redundancy by comparison with other TCP protocols. To verify this, the effectiveness of all protocols is evaluated. The ratio of effective payload to the total packets in the TCP layer is the definition of effectiveness as in Equation 7,

$$P_{eff} = T_{eff} / T_{total} \tag{7}$$

where T_{eff} and T_{total} are the effective traffic in bytes and total sending bytes in the sending side respectively. Note that only effective traffic received is included (i.e., duplicate packets are not interpreted as part of effective traffic). However, every sending packet will be counted as the total traffic, not considering whether it is a retransmission packet. While for standard TCP protocols T_{total} is the sum of T_{eff} and retransmission bytes, for ATCS T_{total} represents the sum of T_{eff}, bytes of retransmission packets, NC header overhead and redundant packets introduced by network coding. From **Fig. 9**, it can be observed that ATCS is more effective than other TCPs under all BER conditions. At 10^{-6} BER, ATCS chooses a less redundant parameter (*N*=3, *R*=1) and there is a marginal improvement in protocol effectiveness. At relative high BER, even if ATCS chooses more redundant parameters (*N*=2, *R*=1) the performance superiority stands out. This phenomenon can be explained as follows. At relatively low BER, TCP employs its intrinsic mechanism to mask packet loss and its retransmission count is approximately equal to the overhead packets of ATCS. Although ATCS introduces more redundancy at high BER, it shows its effectiveness by yielding even more throughput.

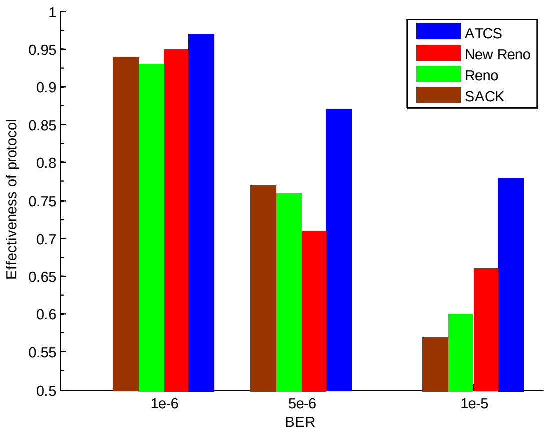


Fig. 9. Effectiveness of all algorithms

5. Conclusion

In this paper we present ATCS, an adaptive TCP coding scheme for satellite IP networks. The proposed algorithm utilizes network coding to introduce information redundancy to cope with link error rate over satellite IP networks. ATCS is compatible with traditional TCP protocols and can be easily deployed to current network protocol stack. Moreover, with an intelligent packet loss method, ATCS is adaptive to a large range of link error rates by dynamically adjusts its redundancy factor according to packet loss rate measurement. Simulations indicate that ATCS greatly improves the throughput performance and file download delay performance. Simulation results also show it is more effective than conventional TCP and it is more suitable to use in satellite IP networks.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, “Network information flow,” *IEEE Trans. on Info. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000. Article (CrossRef Link).

[2] Y. Wu, P. A. Chou and S.-Y. Kung, “Information exchange in wireless networks with network coding and physical-layer broadcast,” *Microsoft Research*, *Tech. Rep. MSR-TR-2004-78*, Aug. 2004.

[3] C. Fragouli, J. Widmer and J.-Y. L. Boudec, “On the benefits of network coding for wireless applications,” in *Proc. of WiOpt*, pp. 1–6, Apr. 2006. Article (CrossRef Link).

[4] J. Liu, D. Goeckel and D. Towsley, “Bounds on the gain of network coding and broadcasting in

wireless networks," in *Proc. of IEEE INFOCOM*, pp. 724–732, May 2007. Article (CrossRef Link).

[5] S. Deb, M. Effros, T. Ho, D. R. Karger, R. Koetter, D. S. Lun, M. Medard and N. Ratnakar, "Network coding for wireless applications: A brief tutorial," in *Proc. of International Workshop on Wireless Ad-hoc Networks*, May 2005.

[6] A. A. Hamra, C. Barakat and T. Turletti, "Network coding for wireless mesh networks: A case study," in *Proc. IEEE WoW MoM*, pp. 103–114, Jun. 2006. Article (CrossRef Link).

[7] S. Katti, H. Rahul, W. Hu, D. Katabi, M. M. M′edard and J. Crowcroft, "XORs in the air: Practical wireless network coding," in *Proc. of ACM SIGCOMM*, pp. 243-254, Sep. 2006. Article (CrossRef Link).

[8] P. A. Chou, Y. Wu and K. Jain, "Practical network coding," in *Proc. of Allerton Conference on Communication*, Control, and Computing, pp. 40-49, 2003. Article (CrossRef Link).

[9] P.S. David and A. Kumar,"Network coding for TCP throughput enhancement over a multi-hop wireless network," in *Proc. of COMSWARE*, pp. 224–233, Jan. 2008. Article (CrossRef Link).

[10] Y. Tian, K. Xu and N. Ansari, "TCP in Wireless Environments: Problems and Solutions," *IEEE Radio Communications*, pp. 27-32, Mar. 2005. Article (CrossRef Link).

[11] D. S. Lun, M. Medard and M. Effros, "On coding for reliable communication over packet networks," in *Proc. of Allerton*, Urbana Champaign, Sep. 2004.

[12] S. Katti, D. Katabi, W. Hu, H. Rahul and M. Medard, "The Importance of Being Opportunistic: Practical Network Coding for Wireless Environments," in *Proc. of Allerton Conference on Communication, Control, and Computing*, 2005.

[13] T. Ho, R. Koetter, M. Medard, D. Karger and M. Effros, "The Benefits of Coding over Routing in a Randomized Setting," in *Proc. of International Symposium on Information Theory (ISIT 2003)*, 2003. Article (CrossRef Link).

[14] T. Ho, M. Medard, J. Shi, M. Effros and D. Karger, "On Randomized Network Coding" in *Proc. of Allerton Conference on Communication, Control, and Computing*, 2003.

[15] M. Ghaderi, D. Towsley and Jim Kurose, "Reliability benefit of network coding" in *Tech. Report 07-08*, Computer Science Department, University of Massachusetts Amherst, Feb. 2007. Article (CrossRef Link).

[16] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inform. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006. Article (CrossRef Link).

[17] D. Lun, M. Medard and R. Koetter, "Network coding for efficient wireless unicast," *IEEE International Zurich Seminar on Communications*, pp. 74 – 77, Jul. 2006. Article (CrossRef Link).

[18] J. K. Sundararajan, D. Shah, M. M′edard, M. Mitzenmacher and J. Barros, "Network coding meets TCP," in *Proc. of IEEE INFOCOM*, 2009. Article (CrossRef Link).

[19] S.-Y. R. Li, R. W. Yeung and N. Cai, "Linear network coding," *IEEETrans. Inform. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[20] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, 2003. Article (CrossRef Link).

[21] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inform. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006. Article (CrossRef Link).

[22] S.M. Bellovin, "Security problems in the TCP/IP protocol suite," *ACM SIGCOMM Computer Communication Review*, 1989. Article (CrossRef Link).

[23] Akyildiz, I.F., G. Morabito and S. Palazzo, "Research Issues for Transport Protocols in Satellite IP Networks," *IEEE Personal Communications*, vol. 8, no. 3, pp. 44–48, Jun. 2001. Article (CrossRef Link).

Wei Dong received the M.S degree in communication and systems from Electronic Science and Technology of China, Chengdu. He is currently a Ph.D. student at the School of Computer Science of Sichuan University. His research interests cover a wide variety of topics in wireless, and satellite networks, with emphasis on seamless handover schemes, and design of transport layer protocols for wireless networks.

Junfeng Wang was born in 1976. He received the M.S. degree in Computer Application Technology from Chongqing University of Posts and Telecommunications, Chongqing in 2001 and Ph.D. degree in Computer Science from University of Electronic Science and Technology of China, Chengdu in 2004. From July 2004 to August 2006, he held a postdoctoral position in Institute of Software, Chinese Academy of Sciences. From August 2006, Dr Wang is with the College of Computer Science, Sichuan University, Chengdu as a professor. His recent research interests include spatial information networks, network and information security, and intelligent transportation system.

Minhuan Huang is currently a professor in Beijing Institute of System Engineering, China, leading the Information Systems Security Lab. His research interests are in the areas of network coding and information security.

Jian Tang was born in 1981. He received Ph. D. degrees in Computer Science from Graduate University of Chinese Academy of Sciences, Beijing, China in 2009. He is a research assistant in Beijing Institute of Systems Engineering. China. His current research efforts are related to the wireless network and information security.

Hongxia Zhou was born in 1976. She received a Master Degree from College of Electronic Engineering, University of Electronic Science and Technology of China. She is with the Sichuan Military District, Chengdu as an associate professor. Her research interests include wireless ad-hoc networks and information security.