A Forward Error-Tolerant Reassembly Method for Receiving Network Data

Xiaomei Wang, Peng Yu, Xiaoyi Zhang, and Liang Fan
Information Science and Engineering Institute, Zhengzhou, China
Email: wangxiaomei.zz@163.com

Qiao Zhang
Henan Institute of Education, Zhengzhou, China

Abstract—Reassembly is an important operation for receiving network data. A fragment with errors is discarded by the conventional reassembly, which renders the information content of the whole fragment lost. The paper claims that the underlying associated relationships among the network data may bring the gain of Forward Error Correction (FEC). It describes the reassembly model based on multiple long-data and proposes a Forward Error-Tolerant Reassembly method based on the Associated Relationships among the network data (FETRAR). FETRAR is able to work alone without the explicit cooperation of the sender entity or other network entities. Simulation and test results illustrate that FETRAR outperforms the conventional reassembly method (short for CR) dramatically. FETRAR constructs much more complete long-data than CR in the one-way communication. It also remarkably deceases the average repeat times of fragments under ARQ with full use of the valid information in erroneous fragments.

Index Terms—Reassembly; Error-Tolerant; Network Data; Associated Relationships

I. INTRODUCTION

The network data means the data that pass through the network above the physical protocol layer in the networks. There may be different terms to describe the network data in different protocol layers. For example, a network data of the data link layer is generally called “frame”, and the one of the IP layer is called “packet”. TCP segment is used to name the network data of the TCP layer. TCP connection is another name from the connection point of view. ATM cell is the name of the network data of the ATM layer, and etc.

Bit errors may occur frequently among the network data due to noise and other issues in the wireless channel. Generally, some protection mechanisms are adopted in the network architecture and its different layers. For some of those layers, an error-detection code is used to protect network data. These error-detection codes can be CRCs or checksums [1, 2]. The conventional reassembly [3, 4] procedure drops the network data with errors, which renders the whole network data lost in the upper layer and the corresponding information content less acquired in some one-way communication scenarios.

To handle errors and achieve reliable transmission, existing approaches mainly focus on the data redundancy or the path redundancy. Channel coding [1, 5-8] is a well-known approach that adds redundancy to the information data. At the receiver, the channel decoder uses the data redundancy to recover errors. As for the path-redundancy approaches [9-13], multiple paths between the sender and the receiver are used to increase the delivery success probability of the network data without errors. ARQ [14] (Automatic Repeat reQuest) and its improved methods [15-17], which try to get better reliability through multiple repeat transmissions, may be implicitly regarded as data-redundancy approaches or path-redundancy approaches.

However, all the above approaches have some shortcomings. First, the retransmission-based recovery mechanisms bring about increased delay. Thus it is difficult for them to satisfy the latency sensitive applications, such as voice over IP. Second, they lead to more communication cost of the network, and thus be not suitable for the networks with the limited resources such as the wireless sensor networks. In those networks, energy efficiency is the key problem [18-21], thus the communication cost should be limited. Third, those solutions that need close cooperation between the sender and the receiver do not fit the one-way transmission, such as unidirectional broadcast.

Some studies [22-26, 29] focus on the receiver-based error tolerant solutions. Ref. [22] proposed a UDP Lite protocol that verifies the partial checksum and drops a packet only if it has errors in the transport or the application layer headers. UDP Lite based transport schemes ignore errors in the application layer payload, but drop all packets that have one or more bit-errors in the IP, UDP, or application layer headers. They require MAC layer checksum to be disabled so that corrupted packets can be passed to higher layers. Ref. [23] proposed a header estimation method to estimate the corrupted critical header fields in a packet while non-critical header fields are simply ignored. However it requires an accurate MAC layer bit-error channel model, which is difficult to acquire. A light TCP protocol which tolerates the errors in the payload data was proposed in [24]. It explores the possibility of accepting TCP packets with an erroneous
checksum, to improve network performance for those applications that can tolerate bit errors.

Studies on wireless multimedia have revealed that the forward error correction on corrupted packets yields better bandwidth utilization and lower delay than retransmissions [25, 26]. Some multimedia standards have introduced enhanced error-resilience and concealment features, e.g., slices in JVT/H.264 [27] and reversible VLC in MPEG-4 [28]. Ref. [29] proposed an I-frame error recovery approach for H.264 video transmission based on data hiding. At the encoder, for an H.264 intra-coded I frame, the important data for each macro block (MB) are extracted adaptively and hidden into the RTP header of the next frame by the proposed data hiding method. At the decoder, if the important data for a corrupted MB can be correctly extracted, the extracted important data will be used for error recovery; otherwise, a simple algorithm is used to recover the corrupted MB.

Based on the idea that the applications may benefit from trading bit errors to get better performance, the paper focuses on the reassembly problem and provides a receiver-based error tolerant solution.

We introduce a term “long-data” which includes a series of fragments to pass through a link with a smaller maximum transmission unit. A long-data may be regarded as a data stream. The reassembly entity reassembles the fragments to the original long-data. The reassembly entity may exist at the receiver or intermediate nodes according to the concrete network setting, the function requirements, and etc.

The network data can be classified into control data and service data. The control data, such as the headers or tailors of the packets (frames), carry the control information. For example, the control data of the TCP segment is the TCP header. A TCP header consists of the destination port, the source port, the sequence number, the acknowledgement number, the checksum and other fields. Generally, the control data is much more susceptible to the errors than the service data, especially those for routing, multiplexing, reassembling, and etc., since bit errors in the control data may render the fragments unrecognized, the corresponding long-data unrebuilt in the upper layer and even the service data useless to the users. These are unacceptable to the users of the network.

It is worth noting that there are potential relationships among the control data. These relationships are not from the agreements made for reliability beforehand, but from the users’ statistical behaviors of using the network. The design and implementation of the network protocols that realize other network functions may also create the associated control data.

The associated relationships among the network data are helpful for the reassembly entity to acquire the gain of forward error correction. They are underlying and should be deeply analyzed and mined. In the paper, a reassembly method is proposed, which is called the Forward Error-Tolerant Reassembly method based on Associated Relationships (FETRAR). FETRAR explicitly focuses on the errors with impact on the control data. It also plays an essential role in recovering the service data.

With FETRAR, the reassembly entity works alone without retransmissions of corrupted fragments from the sender. FETRAR is suitable for the one-way communication. It can also be applied into the two-way communication to support efficient communication schemes. The simulation and test results illustrate that FETRAR dramatically reassembles more complete long-data than the conventional reassembly method in the one-way communication and decreases the average retransmission times of fragments in the two-way communication.

The paper is organized as follows. Section II presents the reassembly model for the network data. FETRAR is detailed in Section III. Section IV shows the simulation results to illustrate the feasibility and the effectiveness of FETRAR. Section V shows the test results based on the real network data. Finally, the concluding remarks are given in Section VI.

II. REASSEMBLY MODEL FOR THE NETWORK DATA

The reassembly model based on multiple long-data is shown as Fig. 1. The model consists of the mixed source, the generalized channel, the reassembly module and the mixed sink.

![Reassembly model for the long-data](image)

A long-data seems to be a stream that starts from only one source (the creator of this long-data) and ends at only one sink. The mixed source contains one or several sources. Each source has a fragmentation module, which is responsible for dividing its incoming long-data into fragments and passing them to the generalized channel. The output data of the mixed source are a series of fragments that come from different sources.

The generalized channel includes the physical channel and some protocol layers under the layer that the long-data belongs to. For example, if we are interested in reassembling AAL PDUs, the generalized channel may include the physical layer, ATM layer and the physical channel as well. The fragments from the generalized channel may have errors.
The reassembly module should classify erroneous fragments into the corresponding long-data, sort them in terms of their original sequence, and provide the reassembled long-data to the pure sink in the mixed sink.

III. FETRAR

Based on the reassembly model above, this section details FETRAR.

Every fragment of the long-data carries two types of information for reassembly.

One is the stream identifier information telling which long-data a fragment belongs to. For example, the source IP address, the destination IP address, the source port and the destination port can identify a TCP connection. They are the identifier of the TCP connection. VPI VCI is another example used to identify the AAL PDU.

The other is the location information showing where the fragment locates in the long-data. Based on the switch modes, different means exist to direct the location information. In packet-switched networks, the sequence number field is used to direct a fragment’s location. And in the virtual circuit networks, the cells from the same long-data retain their original order in terms of time, so the cell’s position is not directed by a special filed in the control data, but by the arrival time of the cell.

Passing through the generalized channel, the fragments may contain errors. A fragment that fails to pass the verification is discarded by the conventional reassembly, which will make a hole in the corresponding long-data and thus create an incomplete long-data. Suppose in Fig. 2, the third fragment in the long-data between source 1 and sink 1 contains errors. It causes the long-data incomplete in the conventional reassembly process under the one-way communication.

![Diagram of reassembly](image)

Figure 2. An example of three pieces of long-data over the reassembly model

To overcome this problem, the reassembly module should have the forward error-tolerant capacity. It aims to make the incomplete long-data complete. Here a complete long-data means whose every fragment locates the correct position, regardless of the errors in the fragments.

There are associated relationships among the control data. They exist in multiple domains, such as the time domain and the protocol domain. They are helpful for the reassembly module to increase the error-tolerant performance, so should be fully utilized. The paper focuses on the virtual circuit networks, and summarizes four pieces of associated relationships among the network data.

A. Definition

(1) $F$ is the set of the long-data, $F=\{f_i | i \in \{1,2,...,K\}\}$, $f_i$ represents long-data $i$. $start(f_i)$ is the start time of $f_i$. $end(f_i)$ is the end time of $f_i$. $id(f_i)$ is the stream identification of $f_i$.

(2) $C_i$ is the set of fragments of the long-data $f_i$, $C_i=\{C'_j | j \in \{1,2,...,J_i\}\}$. $C'_j$ is the fragment $j$ in the long-data $f_i$. $J_i$ is the total number of the fragments that belong to the long-data $f_i$. $C'_i(1)$ is the arrival time of the fragment $C'_i$, which implicitly indicates the position of the fragment $C'_i$ in the long-data $f_i$. $C'_i(2)$ is the stream identification of $f_i$.

B. Associated Relationships among Network Data

R1–R4 are four pieces of associated relationships among the network data. R1 and R2 are the associated relationships in terms of the time domain. R3 and R4 are the ones in terms of the protocol domain.

R1:

$\forall i \in \{1,2,...,K\}, \forall j \in \{1,2,...,J_i\}$

$\exists start(f_i) \leq C'_i(1) \leq end(f_i)$

R2:

$\forall i \in \{1,2,...,K\}, \forall p,q \in \{1,2,...,J_i\}$, if $p < q$ then $C'_i(1) \leq C'_i(2)$

R3:

$\forall i \in \{1,2,...,K\}, \forall p,q \in \{1,2,...,J_i\}$

$\exists id(f_i) = C'_i(2) = C'_i(2)$

R4:

$X = \{x_1, x_2, ..., x_r\}$ is the set containing the values of the stream identification. Generally $X$ accords with some statistical distribution, i.e. $X \sim P(X = x)$.

C. Forward Error-Tolerant Reassembly

The fragments of the long-data in $F$ pass through the generalized channel and arrive at the reassembly module. The fragments with errors create incomplete long-data. These bad fragments are put in the set $Y$. The incomplete long-data are put in the set $F'$. $F' = \{f'_k | k \in \{1,2,...,K'\}\}$. Suppose $C'_i$ contains the right fragments of incomplete long-data $f'_i$. $C'_i = \{C'_j | j \in \{1,2,...,J'_i\}\}$ ($k = 1,2,...,K'$).

FETRAR aims to make the incomplete long-data in $F'$ complete based on R1–R4. That is, for each $y$ that belongs to $Y$, FETRAR deduces its stream identification and position identification. From the associated relationship R2, the position information can be
implicitly acquired by the arrival time. So deducing the stream identification becomes the key problem. According to R4, it is clear that the Bayesian classifier is suitable to solve this problem. What’s more, R1 is useful to reduce the possible long-data scope where y possibly belongs.

The following details how FETRAR deduces the stream identification.

For each y (y ∈ Y), FETRAR executes three steps:

1. First, to compute the start time and the end time of $f_k$, $f_k \in F^k (k = 1, 2, ..., K^*)$.

   \[
   \text{start} \left( f_k \right) = C^i_k \left( 1 \right) - \Delta 
   \]
   \[
   \text{end} \left( f_k \right) = C^i_k \left( 1 \right) + \Delta 
   \]  

   Here \( \Delta = \alpha \times \frac{1}{k - 1} \sum_{j=1}^{k} \left( C^j_i - C^j_i \right) \). \( \alpha \) is the coefficient for adjusting the start time and the end time of the long-data.

2. Second, to achieve $F^*$ based on R1 according to (3) below.

   \[
   F^* = \left\{ f \mid f \in F \text{ and } \text{start} \left( f \right) < y(1) < \text{end} \left( f \right) \right\} 
   \]  

   Over the above two steps, a reduced scope $F^*$ where y possibly belongs is achieved.

3. Third, to conclude the stream identification of y based on Bayesian classifier, shown as (4).

   \[
   \text{id} \left( f^* \right) = \max_{f^* \in \text{X}_k} \left\{ P \left( \text{id} \left( f^* \right) / y \left( 2 \right) \right) \right\} 
   \]  

In R4, the stream identification value in X can be achieved from the complete long-data. Suppose that the generalized channel is binary symmetric channel (BSC) and the stream identification values accord equivalent probability distribution, the Bayesian classification turns to be the minimum distance classification.

IV. SIMULATION RESULTS AND ANALYSIS

We use MATLAB to simulate and evaluate the performance of FETRAR.

In order to simulate the mode that the network data transfers, it is assumed that 50 long data are divided into fixed length fragments. And these fragments will be mixed before passing through the generalized channel. The mixing will not change the sequences of the fragments belonging to the same long data. When transferred in the channel, the fragments may be annoyed by the noise according to BSC and several bits of them may be changed. At the receivers, the fragments will be reassembled into the long-data as their flow signs.

In conventional reassembly, if any fragment of a long data is erroneous, the long data will be taken as an incomplete long-data. While in FETRAR, if the control data of a fragment is correct, the fragment will be taken as a correct fragment, and if all of the fragments in a long data are correct, the long-data will be taken as a complete long-data.

Fig. 3 shows the transfer mode of the long-data in the simulation. Si stands for the ith long-data to be transmitted. The fragments of all the long-data are chosen to be transmitted by roulette selection. The area of the sectors stands for the probability that a fragment of Si may be chosen. That means, a larger area of the sector of Si means a higher probability that a fragment of it will be chosen.

In the simulation, we first generate a fixed probability distribution for each Si and use the roulette to choose the fragments, thus the sending sequences are generated.

![Figure 3. The simulation transmitting mode of the long-data.](image)

The reassembly ratio (RR for short) in the one-way communication is what we are concerned about. RR is defined as the ratio of the complete long-data from the reassembly module to the long-data derived from the mixed source.

We are also concerned about the performance in the two-way communication with ARQ. If there are holes in the long-data, the fragments corresponding to these holes have to be retransmitted according to ARQ. This will increase the communication cost and delay, for which the average retransmission times (ART for short) of the fragments are taken as the metrics to be evaluated. In the simulation, the conventional Selective Repeat ARQ is adopted, that means, only the fragments that cannot pass through the check will be retransmitted.

The evaluations are based on 50 long-data that start randomly. The fragments of the long-data contain the service data (which is regarded as the payload of the fragment) and the control data used only for the reassembly.

The error model of the generalized channel is assumed to be BSC. As for R4 mentioned in Section III, we assume that X follows an equivalent probability distribution. The reassembly without any error-tolerant processing is called the conventional reassembly (CR for short) in the following.

Fig. 4, Fig. 5 and Fig. 6 illustrate the difference between FETRAR and CR in the one-way communication environment in terms of RR. The RR difference between FETRAR and CR is used to evaluate the forward error-correction gain of FETRAR. As the bit error probability of the generalized channel increases, the reassembly ratio of CR deteriorates more quickly than FETRAR, so FETRAR shows good forward error-correction gain. Because FETRAR utilizes associated
relationships among the control data to place these
fragments with errors in the right place of the
corresponding long-data, it is able to bring out more
complete long-data than CR which drops these bad
fragments.

![Diagram](image)

Figure 4. The effect of the number of the fragments in one long-data
on RR in the one-way communication environment

![Diagram](image)

Figure 5. The effect of the length of the service data in the fragment on
RR in the one-way communication environment

![Diagram](image)

Figure 6. The influence of the length of the control data on RR in the
one-way communication environment

Fig. 4 shows how the number of the fragments in one
long-data influences the reassembly ratio. 15 fragments,
30 fragments and 45 fragments in one long-data are
illustrated. The length of the control data is 8 bits, and the
length of the service data is 48 bytes. Both FETRAR and
CR are influenced by the number of the fragments in the
long-data. The more fragments a long-data has, the lower
reassembly ratio it shows.

FETRAR mainly depends on the vital control data to
achieve the reassembly, so the length of the service data
has little effect on its reassembly ratio. But CR is
different from FETRAR. Fig. 5 illustrates the effect of the
length of the service data on reassembly ratio of CR. The
lengths of the service data are respectively 48 bytes, 200
bytes and 1000 bytes. The length of the control data is 8
bits. One long-data contains 15 fragments. The simulation
results show that the longer payload results in lower
reassembly ratio of CR.

The influence of the length of the control data on RR is
worth noticing. Fig. 6 compares FETRAR with CR when
the lengths of the control data are respectively 8 bits, 16
bits and 28 bits. The length of the service data is 48 bytes.
As shown in the figure, the length of the control data
plays an important role in FETRAR. The longer the
control data is, the higher RR FETRAR has. The length
of the control data determines the size of the space. With
the fixed number of the classes, the classifier used in
FETRAR has better performance in the space with bigger
dimensionality. The length of the control data is far
smaller than that of the service data. Its change has little
effect on the length of the fragment. So the length of the
control data has little influence on CR.

![Diagram](image)

Figure 7. The RR comparisons between FETRAR and CR in the two-
way communication environment

In Fig. 7 and Fig. 8, FETRAR is compared with CR in the
two-way communication environment. The maximum
of the fragment retransmission of ARQ is 5. One long-
data has 15 fragments. The length of the control data in
the fragment is 8 bits.

Fig. 7 shows the reassembly ratio. The length of
service data is 48 bytes. As the error probability of the
generalized channel remains low, both FETRAR and CR
are able to reassemble nearly all the long-data. When the
bit error probability of the generalized channel deteriorates, the performance of CR falls down while
FETRAR keeps well in terms of RR.

Fig. 8 shows the effect of the length of the service data.
Fig. 8 (a) illustrates the RR curves. Fig. 8 (b) present the
The Reassembly Ratio

The Reassembly Ratio

The Average Retransmission Times

Fig. 9 and Fig. 10 compare FEETRAR with CR when different maximum retransmission times are set in ARQ.

Fig. 9 illustrates RR curves. Fig. 10 shows ART curves. As shown in the figures, although the maximum of retransmission times is set to 1 in FEETRAR and 5 in CR, FEETRAR outperforms CR both in the reassembly ratio and average retransmission times of the fragments.

V. TEST RESULTS BASED ON THE REAL NETWORK DATA

ATM cells received from the real network are adopted to do the tests.

IP over ATM is chosen as the transfer mode for the test. IP over ATM is the common communication mechanism in the backbone network. The basic principle of IP over ATM is that the IP packets are packaged into ATM cells and transferred in ATM network. An IP packet is first encapsulated into an AAL5 PDU in the ATM adapter layer and then divided and packaged into several ATM cells, as shown in Figure 11.

Fig. 12 shows the details that how an IP packet is transferred in ATM network. In the reverse process we can get IP packets by reassembling ATM data.

| Service Data |
| TCP/IP |
| AAL5 |
| ATM |
| SDH/WDM |

Figure 11. The protocol stack of IP over ATM

Figure 12. The transfer flow of IP over ATM
We program and implement FETRAR and CR respectively to reassemble the AAL5 PDU in the one-way communication. We are also interested in the IP packets contained in the AAL5 PDUs. The following metrics are adopted in the tests.

(1) The number of the reassembled AAL5 PDUs;
(2) The number of the IP packets contained in the reassembled AAL5 PDUs;
(3) The number of the IP packets that pass the IP header checksum;
(4) The total bytes of IP packets contained in the reassembled AAL5 PDUs;
(5) The total bytes of the IP packets that pass the IP header checksum.

The test results are shown in TABLE I. FETRAR improves 136.9% over CR in terms of the number of the assembled AAL5 PDUs, 138.3% over CR in terms of the number of the IP packet, and 89.2% in terms of the number of the IP packets that pass the IP header checksum. FETRAR improves 185.9% over CR in terms of the total bytes of the IP packets and 124.0% in terms of the total bytes of the IP packets that pass the IP header checksum.

All the results prove that FETRAR is able to provide far more service data than CR to the users by using forward error-tolerant reassembly. It significantly improves the performance of the reassembly compared with the conventional reassembly method in the one-way communication.

### VI. Conclusions

The fragments with errors are generally dropped in the conventional reassembly, which decreases the efficiency of the network communication. In fact the underlying associated relationships among the network data may bring the gain of forward error correction and should be fully utilized in the reassembly. This paper introduces the reassembly model for the long-data and proposes a unique reassembly method called FETRAR. FETRAR rebuilds the long-data based on the associated relationships among the network data in terms of the time domain and the protocol domain. Simulation and test results illustrate that FETRAR outperforms CR. It not only reassembles dramatically more complete long-data than CR in the one-way communication, but also reduces the network cost and delay in terms of the average repeated times of the fragment transmission under ARQ.

### Acknowledgment

We would like to extend my sincere thanks to Professor Yu Hongyi. He suggested the research topic and helped us make this paper better.

This work was supported in part by a grant from the Important National Science & Technology Specific Project-Research on Key Technology of the Sensor Networks for Radio Frequency Monitoring (2010ZX0300602-001).

### References


Xiaomei Wang was born on May 23, 1976 in China. She received a Bachelor degree in computer hardware and Master degree in computer software and science respectively in 1998 and 2001. During the next 4 years, she had been researching on the wireless multi-hop networks at information Science and Engineering Institute in Zhengzhou, China, and got her Ph.D degree in communication in 2005. Now she is teacher in Information Science and Engineering Institute and interested in the wireless sensor networks, collaborative network data processing, and etc.

Peng Yu was born in 1988 in China. He obtained a Bachelor degree in communication engineering at Tsinghua University. Now he is studying for master postgraduates at Information Science and Engineering Institute Zhengzhou, China. His research interests include wireless communication and collaborative network data processing.

Xiaoyi Zhang was born in 1966 in China. In 1997 he graduated in Xidian University with a Master degree in communication engineering in Xian, China. He is a professor in Information Science and Engineering Institute in Zhengzhou, China. His research interests mainly include personal communication, signal processing, wireless sensor networks, and so on.

Liang Fan was born in 1994 in China. He obtained a Bachelor degree in communication engineering at information Science and Engineering Institute in Zhengzhou, China in 2013. Now he is studying for master postgraduates in the same institute. His research interests include wireless communication and collaborative network data processing.