An Adaptive FEC with QoS Provisioning for Real-Time Traffic in LEO Satellite Networks

Sungrae Cho†, Ana Goulart†, Ian F. Akyildiz†, and Nikil Jayant‡

†Broadband and Wireless Networking Laboratory (BWN-Lab) ‡ Georgia Tech. Broadband Institute (GTBI)

School of Electrical and Computer Engineering

Georgia Institute of Technology, Atlanta, GA 30332 USA

Email: {srcho, ana, ian, njayant}@ece.gatech.edu

Abstract— **This paper presents an adaptive forward error correction (AFEC) protocol that provides a reliable communication service for real-time traffic over low-earth orbit (LEO) satellite networks. In time-varying wireless links, such as LEO satellite networks, a reliable channel estimation scheme with an appropriate code selection technique is essential for adaptive error-control systems. This paper proposes a new channel estimation scheme that uses receiver-initiated messages (ACKs, NAKs, and INCs). These messages feed back to the transmitter which then selects the code rate for sending packets. In addition, the scheme uses the concept of a dynamic transmittable code set by adjusting the maximum code rate in the set of possible codes from which the transmitter can select. In terms of throughput (from network provider's viewpoint) and packet error rate (from subscriber's viewpoint), performance results show that the proposed scheme guarantees the quality of service (QoS) requirements of real-time applications.**

Keywords— **Error-Control, Adaptive FEC, Real-time Traffic, LEO Satellite Network.**

I. Introduction

Global information access can be provided by different satellite systems. Because of their relatively lower propagation delay and power consumption in comparison with geostationary (GEO) satellite counterparts, *low-earth orbit* (LEO) satellite systems will play an important role in communication services in the near future.

In LEO satellite systems, the wireless channels have higher error rates and burstier error patterns than wireline links. Sources of channel impairments include satellite mobility, fading, interference, and atmospheric conditions. Consequently, error statistics on the satellite channel are time-varying. Efficient error-control for a time-varying channel can be realized with an adaptive coding scheme [5], [11], in which the amount of redundant bits increases when the channel is bad, or decreases when the channel is good.

Moreover, the design of error-control protocols must take into account the tolerance to packet losses in multimedia applications. For instance, video and voice transmissions, unlike data, can tolerate certain error rates (less than 10^{-4}–10^{-2} packet error rates), but they have stringent delay requirements [6].

Adaptive FEC (AFEC) schemes have been widely proposed for multimedia traffic with real-time constraints [1], [4], [8]. However, their performances will not be satisfactory for time-varying wireless channel conditions without an appropriate channel estimation scheme. Hence, the channel estimation scheme, in conjunction with the code selection mechanism, is a core component of the protocols.

In this paper, we propose a link-level AFEC protocol that includes a novel channel estimation scheme. The proposed scheme addresses two important issues in the transmission of real-time traffic on LEO satellite channels:

- *Adaptability to channel conditions, through prompt reactions to channel variations and*
- *QoS provision, especially throughput and packet error rate.*

In the proposed AFEC protocol, the novel channel estimation scheme exploits the concept of a dynamic transmittable code set, where each element of the set is a concatenated code. The size of the code set varies in order to ensure that the packet error rate meets the application's QoS requirement. Receiver-initiated feedback messages[1] provide channel estimation by controlling the code selection mechanism and the code set size. In this way, the dynamic transmittable code set together with the feedback messages provide a fast and low-complexity channel estimation scheme that adjusts to the QoS requirements of real-time applications.

The presentation of our error-control scheme is organized as follows: the AFEC protocol is presented in Section II; the simulation model is described in Section III, followed by the performance evaluation in Section IV; then, conclusions are drawn in Section V.

II. Adaptive FEC Protocol

We propose a link-layer adaptive FEC scheme that allows for a more dynamic adaptation to LEO satellite channel conditions through a new channel estimation scheme and code selection mechanism. By link-layer, we mean that our protocol is applicable as a link-by-link (or hop-by-hop) error-control scheme.

The block diagram of the proposed AFEC scheme is presented in Fig. 1. Real-time traffic from the upper layer is encoded by a concatenated FEC encoder [5]. Encoded packets are then sent through the forward channel to the receiver, which in turn decodes the messages. Additional modules are introduced in the receiver and in the transmitter: the *Control Packet Generator* and the *Code Selector*, which are the main components of our scheme.

In the proposed AFEC scheme, two subsystems can be clearly identified: the concatenated FEC encoder/decoder and the channel estimation scheme. Since the concatenated coding scheme is a well accepted approach, we introduce it as the *fixed* part of our protocol. Instead, our focus is on the *adaptive* nature of the protocol.

A. Concatenated FEC

Concatenation is a scheme in which two codes, an *inner code* and an *outer code*, are used in tandem. The inner code corrects most errors and spreads out burst errors,

This work is supported by NASA–Ames (NAG2–1262), Yamacraw (E21-105), and Georgia Tech. Broadband Institute (E21-A52).

[1]Feedback information in the proposed AFEC protocol is not intended for retransmission as in ARQ protocols.

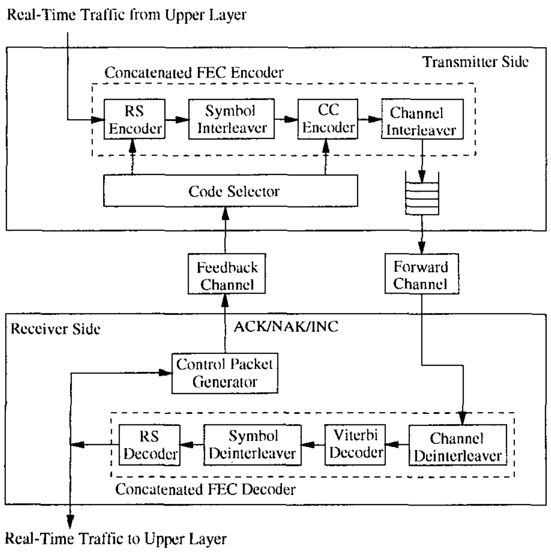

Fig. 1. System Block Diagram of the Proposed AFEC.

then the outer code corrects the small block errors that remain. In our concatenated FEC, the inner coding scheme is a rate-compatible convolutional code (RCPC), which is a family of convolutional codes. RCPC codes generate different code rates from an original rate-1/2 convolutional code. Higher or lower code rates can be obtained with rate-tables by puncturing or repetition [12]. The outer coding scheme is the Reed-Solomon (RS) code, which is particularly effective at correcting short bursts of errors in a data stream. As shown in Fig. 1, we also have interleaver/deinterleaver pairs in order to break up burst errors introduced by the channel. The symbol interleaver disperses bursts errors out of the inner decoder at the symbol level, while the channel interleaver randomizes channel burst errors at the bit level [12].

B. Channel Estimation Scheme

Most of the channel estimation techniques in TDMA and CDMA systems [8] use feedback channel information. Bit error rate (BER) and frame error rate (FER) are examples of feedback information that can be used to estimate the channel conditions. For bit-level error estimates, under the assumption of memoryless channels, a reasonable estimation of BER can be done after a certain period of data collection at the receiver. However, the latency of the data collection can be a drawback for high throughput applications.

Moreover, we are dealing with time-varying channels, which are not memoryless, and errors are statistically dependent. For this reason, the accuracy and reliability of a BER estimate depends on key parameters, such as the confidence interval and confidence level of the estimate [7]. Also, burst errors may be very short and depending on the time window of the data collection, the good channel conditions may be predominant in the computed BER. Then, it is quite possible that the adaptive FEC protocol will not react properly to the bad channel conditions.

Our objective is to provide an approximately accurate picture of the satellite channel conditions, so that the transmission of real-time data is bandwidth efficient and reliable. We propose a scheme that conveys per-packet basis feedback channel information. They influence the code selection mechanism at the transmitter through changes in the code rate and the code set size.

B.1 Control Packet Generation

The control packet generator, which is located at the receiver side, creates control packets based on the results of the decoding process and on the monitored packet error rate (PER). The control packets are transmitted through a feedback channel that we assume to be commonly available in LEO environments. In the case of a symmetric traffic exchange, the receiver can always insert in the data packets a piggybacked control packet.

Positive acknowledgments (ACKs) or negative acknowledgments (NAKs) control the code selection mechanism at the transmitter. An ACK indicates a successful packet transmission, and it is a command to increase the code rate. In contrast, a NAK indicates that a packet was discarded and the transmitter must decrease the current code rate. As a response to a NAK, the transmitter also reduces the maximum code rate, decreasing the code set by 1. The size of the code set decreases in order to ensure that the PER meets the application's QoS requirement.

The receiver updates the PER metric for every packet that arrives, thus obtaining a moving average estimate of the PER. When the PER meets the QoS requirement, a third type of acknowledgment is sent to the transmitter: an *increase* acknowledgment (INC). INC messages compensate for the shrinking in the code set caused by NAK packets. Upon the reception of an INC, the code selector increases the code set size by 1. An INC also represents a positive acknowledgment that indicates to the transmitter that the next higher code rate must be used to encode the next packet. The importance of INC messages in the channel estimation scheme is that they trigger increases in the code set size as a function of the current PER. In this way, besides the per-packet-based feedback provided by ACKs and NAKs, we also have a QoS-based feedback provided by INCs.

B.2 Code Selection Algorithm

The code selector, which is located at the transmitter side, determines the code rate for sending packets, and controls the size of the dynamic transmittable code set. Its operation is based on the following algorithm.

- *Algorithm Description*

At the transmitter, we define the *complete code set* as $C = \{\text{code } 1, \text{code } 2, \ldots, \text{code } N\}$, where the code rate of code i is greater than that of code $(i-1)$ for all $i \in \{2, 3, \ldots, N\}$, and N is the maximum size of the code set. Based on the complete code set, the code selector dynamically updates the highest code rate, which may vary from code N to code 1 during its operation. As a result, we have a *dynamic transmittable code set*, that can have a minimum of one code to a maximum of N possible codes from which the transmitter can select.

Fig. 2 shows how the code selection mechanism operates through dynamic variations in the code set. Each circle in the figure represents a state in the code selection process. The number inside the circle, on the top, indicates the code rate which the transmitter is using to encode packets. Below this number, the size of the code set is given. Feedback messages, which are shown through arrows, produce changes in both the code rate and the code set size. With reference to Fig. 2, let us assume that after a certain period of time, the current code set has L codes, i.e., $C = \{\text{code } 1, \text{code } 2, \ldots, \text{code } (L-1), \text{code } L\}$, where $1 < L < N$. If the transmitter sends a packet with code i $(1 < i < L)$

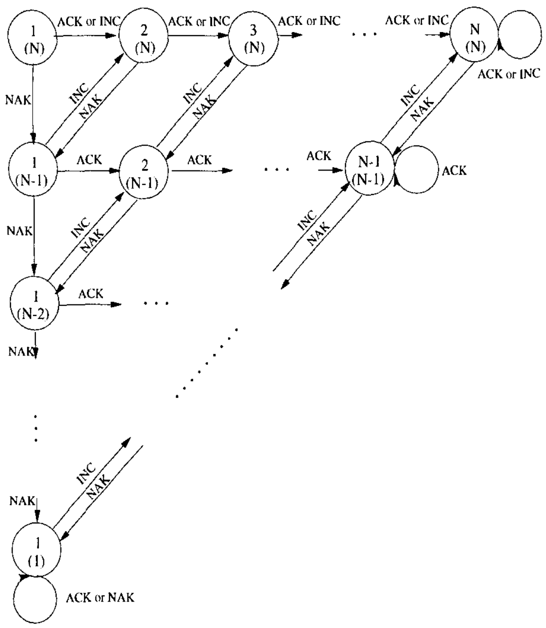

Fig. 2. The Dynamic Transmittable Code Set.

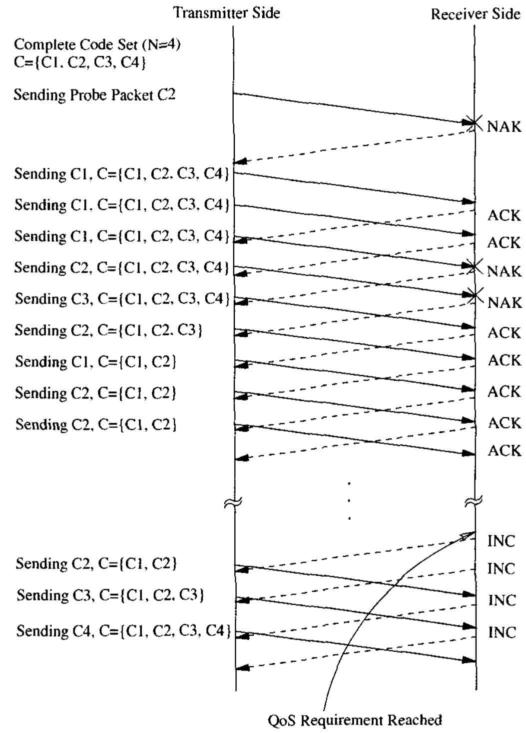

Fig. 3. Example on the Operation of the Code Selection Algorithm.

then the code selection mechanism performs as follows:

- If an ACK arrives at the code selector, the transmitter sends the next packet with code $(i+1)$.
- If a NAK arrives at the code selector, the transmitter sends the next packet with code $(i-1)$, and the code set becomes $C = \{\text{code } 1, \ldots, \text{code } (L-1)\}$.
- If an INC arrives at the code selector, the transmitter sends the next packet with code $(i+1)$, and the code set becomes $C = \{\text{code } 1, \ldots, \text{code } (L), \text{code } (L+1)\}$.

When the packet is transmitted with the highest code rate within the code set (i.e., $i = L$), the code rate increases only when an INC arrives. For the lower boundary condition (i.e., $i = 1$), NAKs decrease the code set size and the code rate remains code 1.

Additionally, we define a *timer* value which is determined by the maximum round-trip delay. If no acknowledgments are received until the timer is expired, the transmitter sends the next packet with code 1, which is the lowest code rate for the worst case channel conditions. In this case, the size of the transmittable code set remains the same.

We describe next how the initial code is selected, and provide a simple example on the code selection algorithm.

- *Initial Code Selection*

When a transmitter sends an initial packet, the channel conditions are unknown until the first control packet arrives. Therefore, we need to define the code rate for the initial transmission. We solve this problem by sending a probe packet (i.e., a dummy data packet) before the actual data transmission. Because sending more probe packets is not desirable for real-time applications, the initial code rate of the probe packet is set to code $\lfloor N/2 \rfloor$. If the first control packet indicates a NAK, the actual packet is transmitted with code 1; otherwise, it is transmitted with code $\lfloor N/2 \rfloor + 1$. In this case, we adopt a conservative approach to select the initial code in just one probe packet. In addition, the size of the code set is initially set to N; thus, when actual packet transmission starts, feedback messages allow the code selector to adapt to the real channel conditions.

- *Example*

As shown in the example in Fig. 3, the initial code set consists of the *complete code set*, which has four codes. At first, a probe packet is sent and the initial code rate is set to $C1$. After packet transmission starts, the control packet generator, at the receiver, generates two ACKs which cause the code selector, at the transmitter, to increase the code rate from $C1$ to $C3$. Then, the arrival of NAKs reduces the code set to $C = \{C1, C2\}$ and the code rate becomes $C1$. Next, after several ACKs, the maximum code rate is $C2$. Note that the code set size increases only when the QoS requirement (PER) is achieved.

Later on, when the computed PER indicates the target is achieved, the receiver sends an INC message. It triggers an increase in the code set, and the selection of the next higher code rate. Since the overall PER remains below the target, continuous INC messages are generated after each successful packet. As a result, the code set becomes $C = \{C1, C2, C3, C4\}$, and the code rate $C4$. The code set size and code rate will change as soon as the channel condition changes.

C. Packet Format

Data and control packets should have adequate fields in order to support our protocol design. We assume that the transmitter uses a framing mechanism that accommodates variable packet sizes, as is the case for most link-layer framing mechanisms used in practice. Moreover, packet fields and corresponding sizes are subject to network design implementation. Our suggestion for data and control packets is shown in Fig. 4.

TI	TOS	SN	LN	M	Rate	Data	FEC
(1bit)	(1bit)	(8bits)	(16bits)	(1bit)	($\log_2 N$bits)	(variable)	(variable)

(a) Data Packet Format

TI	TOS	ACK/NAK/INC	FEC
(1bit)	(1bit)	(2bits)	(4bits)

(b) Control Packet Format

Fig. 4. Suggested Packet Formats.

The format of the data packet is shown in Fig. 4 (a). *TI* is *type indication* which indicates data or control packets. *TOS* is *type of service* which determines if the application is real-time or non-real-time. If it indicates a real-time application, our AFEC scheme is performed; otherwise, an alternative scheme (such as an ARQ protocol) should be used. The data packet has a *sequence number SN* in order to identify out-of-sequence packets at the receiver. Since the data size is variable, we also need the *LN* field which is the *length* of the data field in octets. When there is more data to transmit, the *more* (M) bit is set to 1; otherwise, it is reset to zero. *Rate* indicates the code rate. The length of this field depends on the size of the complete code set. *Data* is the payload with variable size. *FEC* is the adaptive error-control code. Its size is variable, according to the *Rate* field.

Fig. 4 (b) shows the control packet format. The *ACK/NAK/INC* field indicates the type of acknowledgment. Since the feedback channel is also subject to errors, we include an *FEC* field at the end of the control packet. The size of the *FEC* field depends on design implementation. For simplicity, we suggest an *FEC* field of 4 bits.

III. Simulation Model

The performance of the AFEC protocol is evaluated through simulations. In our simulation environment, two hosts represent the communicating nodes, e.g., a LEO satellite and an earth station. They exchange information and control packets through a forward and a feedback channel, respectively. The time-varying wireless channel model is implemented in two other separate hosts, which are connected to the communicating nodes. Since our AFEC protocol is packet-based, we use UDP sockets in the simulations.

The LEO satellite channel is modeled as a three-state Markov bursty channel model [11], [14]. For satellite channels, experimental results have shown that the channel conditions are stationary over a short time interval [11]; thus, these channels are in general represented by M-stationary bursty channel models. A reasonable number of states M for experimental channels has been reported to vary from 2 to 4 [11].

In typical Markov models [9], [14], each state has a fixed bit error probability; however, this cannot reflect the real fading environment. In our three-state Markov model, each state represents a frequency non-selective slow-fading Rician channel. Fluctuations in signal strength occur as time passes, and the channel quality can be *bad* (state 0), *intermediate* (state 1), or *good* (state 2). For each state, we generate different levels of Rician fading by using Jake's simulator [10].

Finally, we consider the motion of LEO satellites which introduces a Doppler shift for the transmitted signals. Assuming an orbital satellite velocity v of approximately 26,600 km/h, and a carrier frequency ω_0 of 1.6 MHz, the maximum Doppler frequency shift f_d is 39.407 Hz ($f_d = v \cdot \omega_0 / c$ where c is the speed of light). The round-trip delay is set to vary between 2.5 and 7.5 msec [13].

TABLE I
Transmittable Code Set

	Concatenated Code	Code Rate
Code1	RS(255,124) x CC(Rate=1/2)	0.243
Code2	RS(255,124) x CC(Rate=2/3)	0.324
⋮	⋮	⋮
Code21	RS(182,124) x CC(Rate=5/6)	0.568
Code22	RS(182,124) x CC(Rate=6/7)	0.584
Code23	RS(140,124) x CC(Rate=2/3)	0.590
⋮	⋮	⋮
Code44	RS(140,124) x CC(Rate=14/15)	0.827
Code45	RS(140,124) x CC(Rate=15/16)	0.830

TABLE II
Channel Models

Model		State 0	State 1	State 2
1	P_{ii}	$1-5\times10^{-7}$	$1-5\times10^{-7}$	$1-7.1\times10^{-8}$
	π_i	0.1	0.2	0.7
2	P_{ii}	$1-5\times10^{-7}$	$1-10^{-6}$	$1-3.75\times10^{-7}$
	π_i	0.3	0.3	0.4

IV. Performance Evaluation

In this section, our objective is to show, through simulations, how the proposed AFEC protocol performs, in comparison with other static and adaptive FEC schemes. The input traffic is real-time application at a constant bit rate of 64 kbps. Our performance metrics are *throughput* (ratio of the number of successfully transmitted information bits to the total number of transmitted bits) and *packet error rate* (ratio of the number of erroneous packets to the total number of transmitted packets).

The transmittable code set, as shown in Table I, contains 45 concatenated codes with code rates ranging from 0.243 to 0.830. For example purposes, we simulate a wide range of code rates to better illustrate the concept of the dynamic transmittable code set, and to clarify that there are no fixed code rates associated with a certain channel condition. However, in a real system, the code set should be selected according to the reliability that the system requires, the available bandwidth, and the channel characteristics.

We perform two types of experiments. First, we compare the proposed AFEC protocol to three other static FEC (SFEC) schemes, which correspond to code 1, code 22, and code 45 in our transmittable code set (Table I). Second, we compare the performance of our scheme with the AFEC proposed in [11], to which we will refer as the *conventional* AFEC. We choose this conventional AFEC because it has similarities with ours, such as the use of concatenated codes and the assumption of time-varying Rician channels.

By varying the parameters of the three-state Markov model, the experiments are based on two wireless channel models, *Model 1* and *Model 2*. The transition probabilities P_{ii} and corresponding steady-state probabilities π_i are given in Table II. Comparing the two models, *Model 2* has a higher probability of being in state 0, i.e., the *bad* state.

A. Experiment 1

Fig. 5 shows PER over average SNR for our AFEC and the SFEC schemes, with *Model 1* as the channel model. The results confirm that our AFEC can guarantee the QoS requirement (with target PER $= 10^{-2}$) for all SNR values.

For the schemes that achieve an acceptable PER, which are SFEC code 1 and the proposed AFEC, we compare throughput results. As shown in Fig. 6, throughput dou-

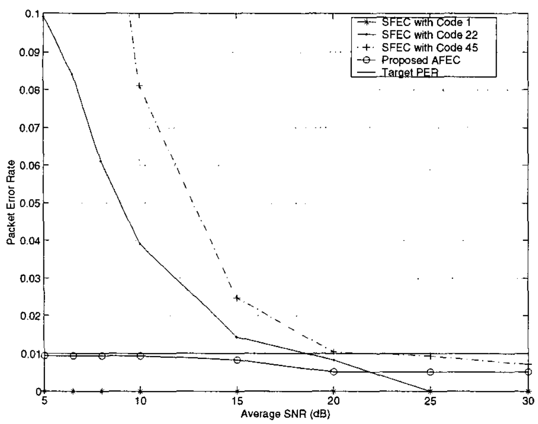

Fig. 5. Experiment 1: Packet Error Rate vs. Average SNR.

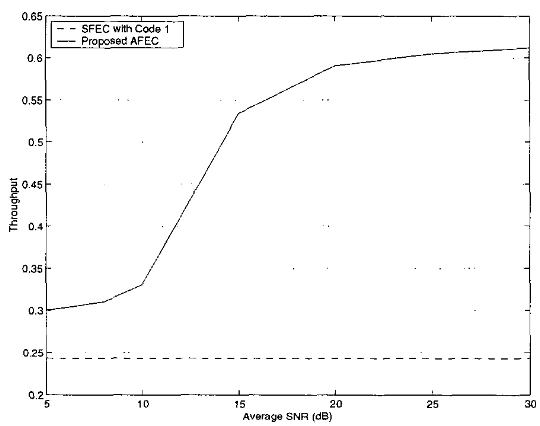

Fig. 6. Experiment 1: Throughput vs. Average SNR.

bles from the low-end to the high-end values of channel SNR for our AFEC. Note that the maximum throughput achievable in the transmittable code set is 0.830.

As shown in Fig. 6, when the SNR values are lower than 10 dB, the PER is held below the target PER at the expense of high bandwidth consumption. Then, from 10 to 15 dB, the throughput has a considerable increase, which corresponds to the initial decrease in the PER in Fig. 5. The reason is that higher code rates are selected as soon as the channel quality improves.

B. *Experiment 2*

Fig. 7 shows PER over average SNR for our AFEC and the conventional AFEC of [11]. The results are obtained after running two sets of experiments, with *Model 1* and *Model 2*, respectively.

The results show that the conventional AFEC cannot guarantee the target PER for all SNR values. When the fading environment becomes worse (*Model 2*), the PER of the conventional AFEC even becomes larger. However, the proposed AFEC has PER below the target for all SRN values, independent of the two channel models. This type of control is obtained by explicitly adjusting the dynamic transmittable code set when the channel conditions change; thus, ensuring that the expected QoS is achieved.

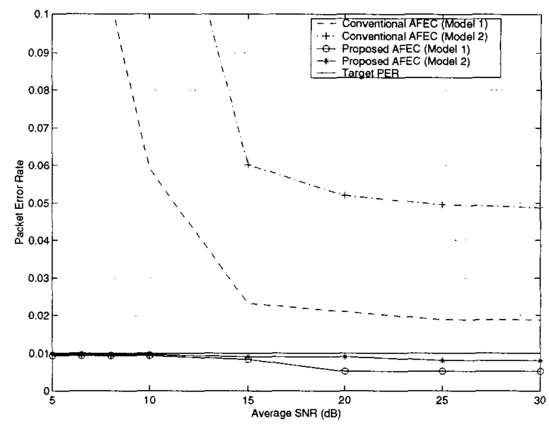

Fig. 7. Experiment 2: Packet Error Rate vs. Average SNR.

V. Conclusions

In this paper, we propose an AFEC protocol with a new channel estimation scheme and code selection mechanism. The advantages of our channel estimation technique are the provision of QoS adaptability and simple implementation. The proposed AFEC is evaluated for real-time traffic over LEO satellite channels. Taken the tradeoffs between throughput and PER into consideration, our AFEC outperforms the other static and AFEC protocols.

References

[1] J. Bolot, S. F. Parisis, and D. Towsley, "Adaptive FEC-based Error Control for Internet Telephony," *in Proc. of IEEE INFOCOM '99,* New York, USA, pp. 1453–1460, Mar. 1999.

[2] D. J. Costello Jr., J. Hagenauer, H. Imai, and S. B. Wicker, "Applications of Error-Control Coding," *IEEE Trans. on Information Theory,* vol. 44, no. 6, pp. 2531–2560, Oct. 1998.

[3] S. Cho, "Rate-Adaptive Error Control for Multimedia Multicast Services in Satellite-Terrestrial Hybrid Networks," *in Proc. of IEEE ICC 2000,* New Orleans, USA, pp. 446–450, Jun. 2000.

[4] M. Elaoud and P. Ramanathan, "Adaptive Use of Error-Correcting Codes for Real-time Communication in Wireless Networks," *in Proc. of IEEE INFOCOM '98*, San Francisco, USA, pp. 548–555, Jun. 1998.

[5] I. Joe, "An Adaptive Hybrid ARQ Scheme with Concatenated FEC Codes for Wireless ATM," *in Proc. of ACM/IEEE MOBICOM '97,* Budapest, Hungary, pp. 131–138, Sep. 1997.

[6] H. Liu and M. E. Zarki, "Performance of H.263 Video Transmission over Wireless Channels Using Hybrid ARQ," *IEEE J. on Select. Areas in Comm.,* vol. 15, no. 9, pp. 1775–1786, Dec. 1997.

[7] M. D. Knowles and A. I. Drukarev, "Bit Error Rate Estimation for Channels with Memory," *IEEE Trans. on Comm.,* vol. 36, no. 6, pp. 767–769, Jun. 1988.

[8] S. Nanda, K. Balachandran, and S. Kumer, "Adaptation Techniques in Wireless Packet Data Services," *IEEE Comm. Mag.,* vol. 38, no. 1, pp. 54–64, Jan. 2000.

[9] D. Qiao and K. G. Shin, "A Two-Step Adaptive Error Recovery Scheme for Video Transmission over Wireless Networks," *in Proc. of IEEE INFOCOM 2000,* Tel Aviv, pp. 1698–1704, Mar. 2000.

[10] G. L. Stüber, *Principles of Mobile Communication,* Kluwer Academic Pub., Massachusetts, 1996.

[11] B. Vucetic, "An Adaptive Coding Scheme for Time-varying Channels," *IEEE Trans. on Comm.,* vol. 39, no. 5, pp. 653–663, May 1991.

[12] S. B. Wicker, *Error Control Systems for Digital Communication and Storage,* Prentice-Hall Inc., New Jersey, 1995.

[13] W. W. Wu, E. F. Miller, W. L. Pritchard, and R. L. Pickholtz, "Mobile Satellite Communications," *Proc. of the IEEE,* vol. 82, no. 9, pp. 1431–1448, Sep. 1994.

[14] M. Zorzi and R. R. Rao, "On the Impact of Burst Errors on Wireless ATM," *IEEE Personal Comm. Mag.,* vol. 6, no. 4, pp. 65–76, Aug. 1999.