

Short Paper

Improving 802.11 Wireless TCP Performance with Adaptive Reed-Solomon Codes: An Experimental Study

JUNG-SHIAN LI AND MING-WANG GUO
*Department of Electrical Engineering
Institute of Computer and Communication Engineering
National Cheng Kung University
Tainan, 701 Taiwan
E-mail: jsli@mail.ncku.edu.tw*

TCP is currently the dominant protocol for the Internet having originally been designed to perform congestion control and flow control in wired networks. Recently, wireless LAN has been very successful in extending the connectivity of the Internet. However, the performance of TCP in the wireless environment is worse than that in the wired one. Furthermore, the error rate changes with the time in the wireless environment due to EMI, movement, or auto-rate-selection. Especially, the popular 802.11 wireless LAN does not employ any FEC or ARQ to improve error rate performance. In order to improve the wireless TCP performance, TCP with adaptive Reed-Solomon codes (ARSC) is proposed. The performance of the proposed TCP with ARSC is verified by simulations and experiments in the test bed.

Keywords: FEC, wireless TCP, Reed-Solomon code, wireless TCP with ARSC, 802.11

1. INTRODUCTION

In order to improve the performance of TCP [2-4, 8] in changing networks, many new versions of TCP, such as NewReno and SACK [6], have been proposed. However, these have focused on wired networks, while wireless LAN is now in widespread use. Wireless environments are distinct from wired ones and random bit error rate should not be ignored. For simplicity and cost-effectiveness, wireless LAN does not employ ARQ (Automatic ReQuest retransmission) or FEC (forward error control) to handle the random bit error in the data link layer. In this paper, we implement a wireless TCP with adaptive Reed-Solomon codes (ARSC). The Reed-Solomon coding schemes are adaptive according to the analytical models proposed in [4]. Since ARSC can adjust the code rate to correct erroneous symbols according to the error conditions, the wireless TCP can achieve better performance. It is very suitable for deployment in the 802.11 wireless LAN, which is not equipped any error-correcting codes in the data link layer.

Received May 8, 2003; revised June 14, 2004 & February 21, 2005; accepted April 11, 2005.
Communicated by Chu-Sing Yang.

2. RELATED WORKS

2.1 Reed-Solomon Codes

Reed-Solomon (RS) codes are currently widely used for applications in digital communication and storage. As shown in Fig. 1, the RS encoder takes symbols from a data source and adds redundant bits to form a RS block. The errors may occur for a number of reasons, such as noise or interference, scratches on a CD, etc. The RS decoder processes each input RS block and attempts to determine errors and correct them. The number of errors that can be corrected depends on the characteristics of RS codes.

Symbol is a popular term used in RS codes. A symbol may contain a fixed number of bits, e.g., 7 bits. For a symbol with s bits, the maximum codeword length for a RS code is $2^s - 1$ symbols, that is, $s(2^s - 1)$ bits. Generally, a RS code is specified as $RS(n, k)$ with s -bit symbols. For a $RS(n, k)$ code, there are n symbols of s bits to form a RS block. In the block, k symbols belong to data symbols and the remaining $n - k$ symbols are parity symbols. A Reed-Solomon decoder can correct t symbols that contain errors in a codeword, where $2t = n - k$. Fig. 2 shows a RS code block.

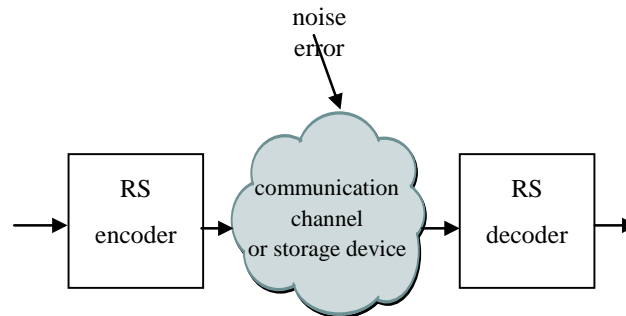


Fig. 1. A typical Reed-Solomon codes system.

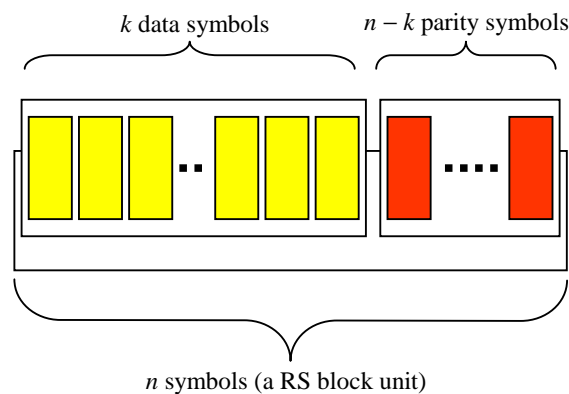


Fig. 2. A Reed-Solomon code block.

2.2 Existing Wireless TCP Algorithms

In order to improve the performance of TCP, many new versions of TCP have been proposed. For example, NewReno, standardized in RFC 2582, introduces a modification to solve the problem of multiple packet losses. TCP SACK, standardized in RFC 2018, introduces a selective acknowledgement option in the TCP header to let the sender retransmit the lost packets more efficiently. However, characteristics of the wireless environment are quite different from the wired one. Therefore, many new mechanisms for the wireless environment have been proposed. In [1], the authors proposed a fast retransmission solution, which focused on the handoff problem. Moreover, many new mechanisms split the TCP connection into two portions. The first portion, from fixed host (FH) to base station (BS), is in the wired environment. The second portion, from base station (BS) to mobile host (MH), is in the wireless environment. In other words, the split connection mechanisms have modified the end-to-end semantic. Indirect-TCP (I-TCP) [1] was the first approach using the split connection model. In addition, M-TCP [5] is based on I-TCP to improve the TCP performance in wireless communication. The goal of these two mechanisms is that the packets sent by the fixed host can be acknowledged by the base station without waiting for the mobile host. WTCP [7] is one of the well-known split mechanisms used to enhance the performance of TCP in wireless communication. The WTCP mechanism does not modify either the fixed host or the mobile host, but instead, the base station includes two stacks, TCP and WTCP. As the base station receives a packet or an ACK from the fixed host, it saves the packet or ACK and then sends it to the mobile host. Hence, as the base station detects packet losses and ACK losses, it can retransmit them immediately. Therefore, in the split mechanisms, the base station has to store the packets for each TCP flow. Consequently, the split solutions often suffer from high overhead and, in addition, they are not scalable. Forward error correcting (FEC) is one solution to reduce the frame error rate. However, it also wastes bandwidth when the wireless channel is in a good state. Therefore, "adapting" the FEC code is a desirable solution to improve performance without wasting wireless bandwidth. We try to build adaptive FEC codes in the transport layer just under the TCP protocol stack. The solution is flexible for the various link layer technologies with or without implementation of FEC codes in the data link layer.

3. ARSC SYSTEM

3.1 Decoding Time for Various RS Block Size

Here, we show the decoding time for various RS codes. Figs. 3 and 4 show the decoding time for RS code with 127 symbols and 63 symbols, respectively. The y -axis represents the decoding time and the x -axis represents number of data symbols for a block (X). As shown here, the decoding time increases linearly when X decreases, although, when the data ratio decreases, the error correcting ability is better.

The data ratio of both RS(127, 119) and RS(63, 59) is 0.936. However, the decoding time for RS(127, 119) is about 0.046 msec, and the decoding time for RS(63, 59) is about 0.0125 msec. This demonstrates that a large RS block needs more decoding time than a

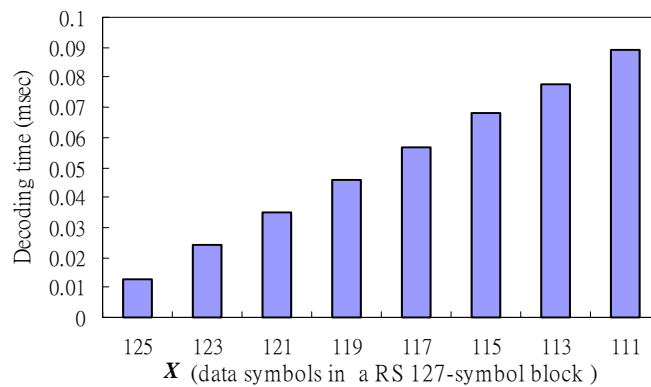


Fig. 3. Decoding time for Reed-Solomon codes with 127-symbol block.

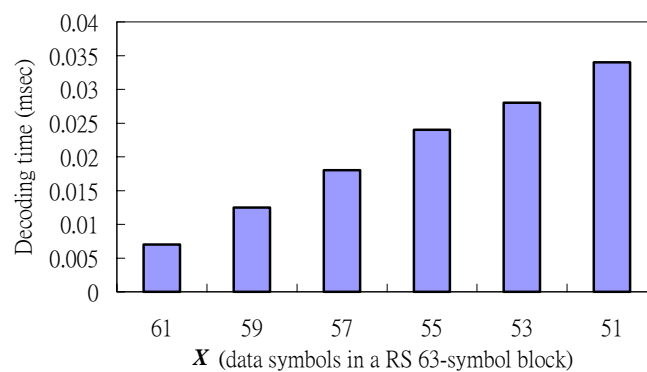


Fig. 4. Decoding time for Reed-S codes with 63-symbol block.

small block when the data ratio is the same. Even so, the large RS block still has advantages. RS(127, 119) can correct a 4-bit error in a block, whereas RS(63, 59) can correct only a 2-bit error. Though the two schemes have same data ratio, RS(127, 119) is better at resisting burst bit errors. Therefore, for the same data ratio, the larger RS block exhibits better error correction ability, whereas the smaller RS block spends less time on decoding. Furthermore, the decoding time of all the RS schemes is less than 0.1 ms in our experiments. That is, to equip the RS coding schemes in the existing mobile hosts is possible.

3.2 TCP with Adaptive Reed-Solomon Codes

In order to improve the performance of wireless TCP, the adaptive Reed-Solomon codes (ARSC) are adopted. Here, we describe the ARSC system and explain each of its components in detail. Then, the experiments for wireless TCP with ARSC are performed with different distances for the wireless link. The wireless TCP with ARSC system provides better performance than TCP with a fixed RS code or TCP SACK.

3.2.1 ARSC design

3.2.1.1 Adaptive Reed-Solomon code in client site (ARSCCS)

As shown in Fig. 5, the functional components of an adaptive Reed-Solomon code in the client site (ARSCCS) are shown. There are four components in ARSCCS, including performance estimator, RS parameter setter, RS parameter negotiator and RS decoder. The performance estimator has two sub-components, which are rate estimator and error estimator. The rate estimator evaluates ARSC throughput and the error estimator calculates the RS block number, which may have error symbols after decoding. The RS parameters setter uses the results of the performance estimator to dynamically set the suitable parameters for the RS decoder to optimize the throughput of ARSC. Based on RS parameters setter, the RS decoder retrieves the input RS data block. The RS parameter negotiator sends the negotiation message to the adaptive Reed-Solomon code in the server site (ARSCSS). This allows ARSCCS and ARSCSS to use the same Reed-Solomon parameter settings.

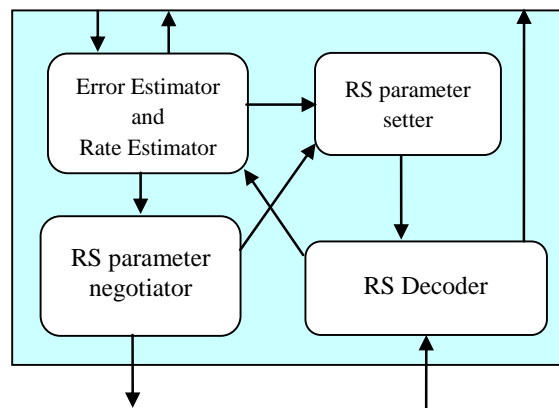


Fig. 5. Adaptive Reed-Solomon codes in client site (ARSCCS).

3.2.1.2 Adaptive Reed-Solomon code in server site (ARSCSS)

The functional components of the adaptive Reed-Solomon code in the server site (ARSCSS) are shown in Fig. 6. There are three components in ARSCSS, including RS parameter negotiator, RS parameter setter and RS encoder. The RS parameter negotiator receives the negotiation message from the client site and places the message in the RS parameter setter. Based on the message, which the RS parameter negotiator has just sent, the RS parameter setter dynamically sets the same parameters as ARSCCS. Then, RS data blocks, which are calculated by the RS encoder, are sent to the client.

3.2.2 Adaptive Reed-Solomon codes

Here, we use TCP without RS code together with six RS codes, including seven

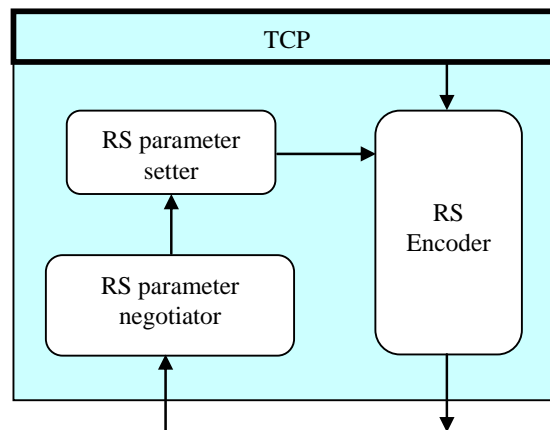


Fig. 6. Adaptive Reed-Solomon codes in server site (ARSCSS).

states in the ARSC system. Each client site uses RS(127, 125) as the initial code rate, which provides basic error correcting ability. In addition, we can stipulate the thresholds, which are used to change the code rate to improve the wireless TCP performance according to the mathematical analysis developed in [4, 16].

4. SIMULATIONS

4.1 A Single Congested Router

The topology of a single congested router is shown in Fig. 7. The bandwidth and delay for the congested link are 4 Mbps and 10 ms, respectively. There are one wired-cum-wireless TCP flow and 20 wired TCP flows passing the congested router. The range of round trip times is 60 to 100 ms. For the wired-cum-wireless TCP, the total transmission and propagation time is 80ms. The wired-cum-wireless TCP's sender and receiver are node S0 and node D0, respectively.

Here we use RS(127, X), where $X = 125, 123$ and 121 , to perform simulations for the topology in Fig. 7. In Fig. 8, we use $(127, a, b)$ to represent each curve, where " a " denotes the original message length in the RS code and " b " could be A (Padhye's analysis [4]) or S (simulation). As Figs. 8 and 10 indicate, all of the analyses are accurate for wireless link bit error rates between 10^{-4} and 10^{-3} , which are common in the wireless environment. In addition, when the bit error rate is less than 0.0001, sending rate for these three codes are very close. In other words, when the bit error rate is less than 0.0001, the RS(127, 125) is selected since it adds the fewest redundant bits. However, when the bit error rate is larger than 0.0003, it is better to choose the RS(127, 123) code. Similarly, when the bit error rate is larger than 0.0007, it is better to choose the RS(127, 121) code. Consequently, the analysis results could be used to choose an RS coding rate to improve the performance of wireless TCP in different error conditions. If the wireless environment is varied, the error rate changes. TCP with adaptive coding schemes can achieve better performance than TCP using a fixed coding scheme.

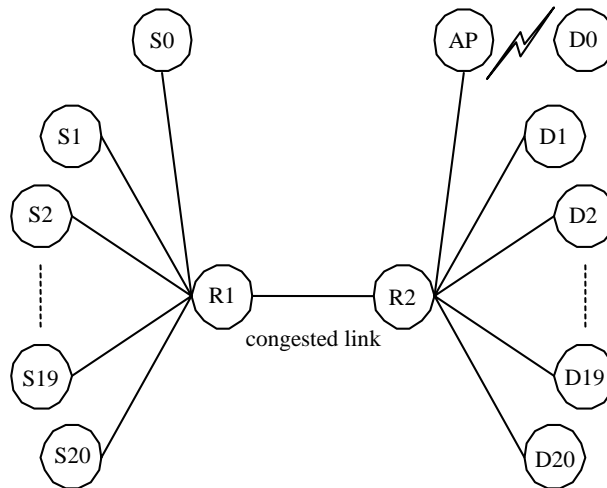


Fig. 7. The topology of one congested link.

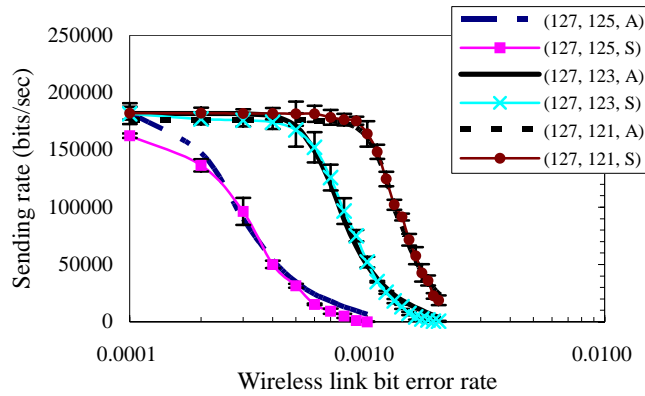


Fig. 8. The sending rate of wired-cum-wireless TCP with RS code passing one congested router.

4.2 A Network of Multiple Congested Routers

Fig. 9 illustrates the topology of a network with multiple congested routers. The bandwidth and delay for these congested links are 2 Mbps and 10 ms, respectively. One wired-cum-wireless TCP flow competes with 20 wired TCP flows. For the wired-cum-wireless TCP, the total transmission and propagation time is 80 ms. The wired-cum-wireless TCP’s sender and receiver are node S and node R, respectively. On each congested link, there are 6 wired connections with different *RTT*s (Round Trip Times) passing by. The range of round-trip delays is 50, 100 ms.

Similar to the settings in section 4.1, here we also use RS(127, *X*), where *X* = 125, 123 and 121, to perform simulations in the topology shown in Fig. 9. In Fig. 10, (127, *a*, *b*) indicates each curve, where “*a*” represents the original message length in RS code

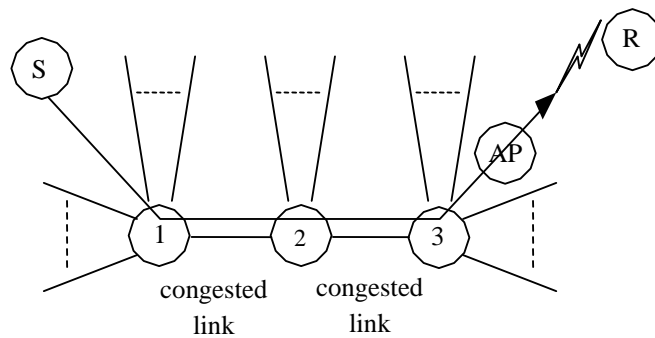


Fig. 9. The Topology of Multiple Congested Links.

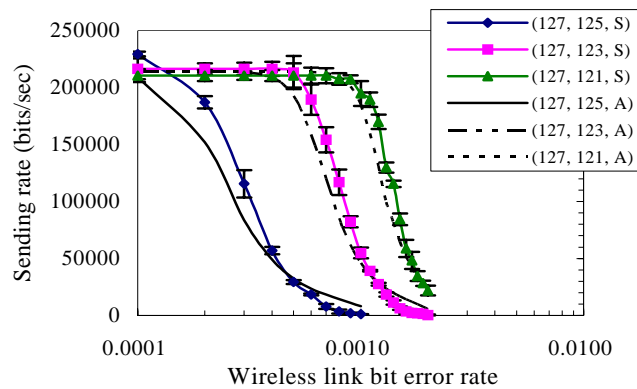


Fig. 10. The sending rate of wired-cum-wireless TCP with RS code passing multiple congested routers.

and “ b ” can be A (analysis) or S (simulation). Similar to the results in section 4.1, all of the analyses are accurate for wireless link bit error rates between 10^{-4} and 10^{-3} . The analysis results here can be used to adapt the RS coding rate to obtain better performance of wireless TCP in an error prone environment for multiple congested routers.

As described above, we can find that Padhye’s analysis is accurate, and so, can be used to estimate the thresholds for an ARSC system. An ARSC system adapts a code rate to correct error symbols. However, how to set the suitable code rate is an important issue. Since it trades off between code rate (error-correcting ability) and bandwidth utilization, TCP performance depends on the thresholds used by the analysis model [16]. With the calculated thresholds, we can find suitable settings for the code rate for various wireless error rates.

Though the throughput of wireless TCP passing one congested router is different from that passing multiple congested routers, the falling edge of each curve in Fig. 8 is close to that in Fig. 10. Consequently, no matter whether for one congested router or multiple congested routers, the thresholds can be set as follows. In the experiments in section 5, when the wireless bit error rate is less than 0.00005, TCP without RS is adopted; when it is between 0.00005 and 0.0002, RS(127, 125) is adopted; when it is

between 0.0002 and 0.0003, RS(127, 123) is adopted; when it is between 0.0003 and 0.0005, RS(127, 121) is adopted; when it is between 0.0005 and 0.0007, RS(127, 119) is adopted; and when it is between 0.0007 and 0.001, RS(127, 117) is adopted. Otherwise, the ARSC system adopts RS(127, 115).

5. EXPERIMENTS

When a bit-error wireless channel is considered, the symbol size used for RS coding, the distance between mobile host and access point, and the TCP packet length may all affect the transmission efficiency. In order to observe the relationship between application-layer forward error control and the distance between the mobile host and access point in a wireless channel, the transport-layer payload (TCP packet length) is fixed at 1016 bytes, and contains eight 127-symbol RS codewords in our experiments.

Here, we present the experiments for wireless TCP with adaptive Reed-Solomon codes, considering the effect of the distance between the access point (AP) and the client site. Seven states, TCP without RS code and TCP with RS(127, i), $i = 125, 123, 121, 119, 117$ and 115, are used to adjust the code rate in our experiments. In order to show the efficacy of the ARSC system, we performed experiments for different distances to let the ARSC system dynamically choose the suitable code rate. All experiments were performed in an environment EMI disturbance. In contrast to TCP with ARSC, we performed experiments with TCP RS(127, 125) and TCP SACK. Furthermore, the Reed-Solomon codes in our experiments do not protect the IP header. If an error occurs in the header, the packet is dropped. Though some vendors of 802.11 implement auto-rate-selection, the AP uses only fixed 11 Mb/sec in our experiments. Basically, ARSC can work well with auto-rate-selection. When ARSC cooperates with auto-rate-selection, the AP sets the 802.11 transmission rate, e.g., reduces the speed from 11Mbit/s to 5.5 and 2Mbits/s when too many errors are experienced, and ARSC can adapt the code rate by sensing the error condition change. Furthermore, clients at different distances from the AP can use different code rates as needed.

As shown in Fig. 11, TCP with ARSC achieves better performance than TCP with RS(127, 125) or TCP SACK in various wireless link distances. In the 20-meter wireless link case, TCP with ARSC has the same performance as TCP with RS(127, 125) since the error condition remains the same during the experiments. However, in the 30-meter or 40-meter condition, we can see that TCP with ARSC achieves better performance than TCP with a fixed RS coding scheme. As can be seen from Fig. 12 of the average delays for transmitting a 1-MB file, TCP with ARSC has a smallest delay in most cases. As Fig. 13 indicates, the code rate for ARSC system adapts to the wireless environment with changing error rate. It shows that an ARSC system adapts a code rate according to the error conditions which change with time.

6. CONCLUSIONS

TCP with ARSC is proposed in this paper. The motivation of ARSC is the ability of today's mobile hosts to perform Reed-Solomon coding. Basically, the RS code with a

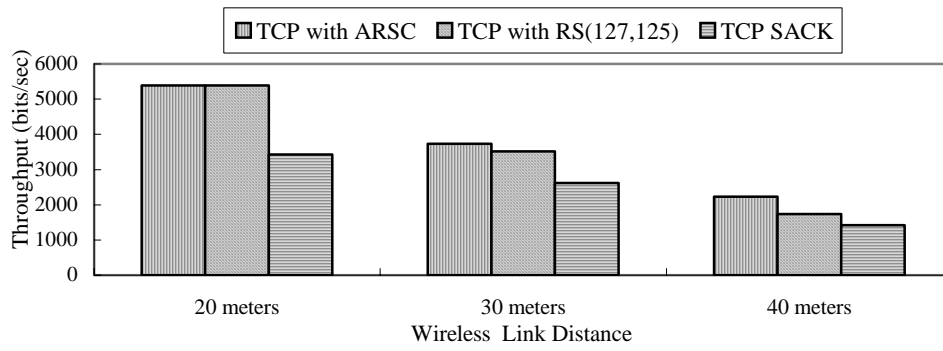


Fig. 11. Throughput in various wireless link distances.

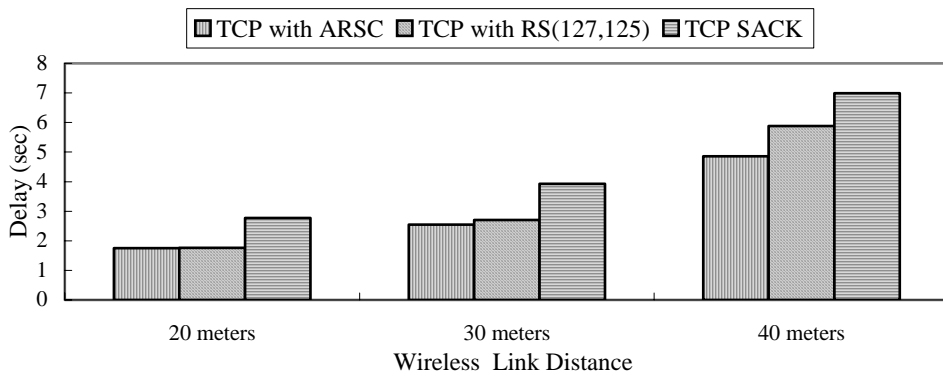


Fig. 12. Average delay for transmitting a 1-MB file.

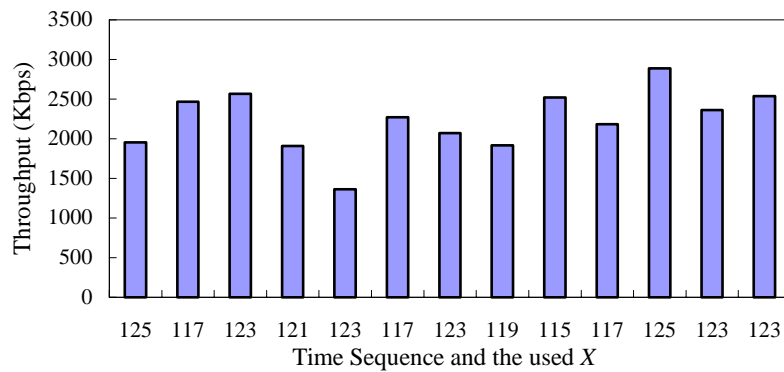


Fig. 13. RS coding adaptation in time for ARSC system (wireless link distance 40 meters).

smaller data ratio exhibits better error correcting ability. However, an RS code with many parity symbols wastes valuable wireless bandwidth. To equip the suitable error correcting ability is important for TCP performance. Therefore, to adapt the

Reed-Solomon codes is a desirable way to improve the performance. The thresholds, estimated by the TCP analytical model in [16] and verified by simulations, are used for the adaptation of ARSC. Experiments show that TCP with adaptive Reed-Solomon codes (ARSC) achieves better performance. The improvement is more obvious when error conditions change with time, regardless if it is due to EMI, mobility, or auto-rate-selection.

REFERENCES

1. A. Bakre and B. R. Badrinath, "I-TCP: indirect TCP for mobile hosts," in *Proceedings of 15th International Conference on Distributed Computing Systems*, 1995, pp. 136-143.
2. D. Lin and H. T. Kung, "TCP fast recovery strategies: analysis and improvements," in *Proceedings of INFOCOM*, 1998, pp. 263-271.
3. J. Postel, "Transmission control protocol," *RFC793*, 1981.
4. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP reno performance: a simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, Vol. 8, 2000, pp. 133-145.
5. K. Brown and S. Singh, "M-TCP: TCP for mobile cellular networks," *ACM Computer Communication Review*, Vol. 27, 1997, pp. 19-43.
6. M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgement options," *RFC2018*, 1996.
7. P. Sinha, N. Venkitaraman, R. Sivakumar, and V. Bharghavan, "WTCP: a reliable protocol for wireless wide-area networks," in *Proceedings of 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, 1999, pp. 231-241.
8. W. Stevens, "TCP slow start, congestion control, fast retransmit, and fast recovery algorithm," *RFC2001*, 1997.

Jung-Shian Li (李忠憲) joined National Cheng Kung University, Taiwan, in the spring of 1999 as an Assistant Professor in the Department of Electrical Engineering. Now, he is an Associate Professor in the same department. He graduated from the National Taiwan University, Taiwan, with a B.S. degree in Electrical Engineering in 1990, and a M.S. in 1992. In 1998, he obtained his Ph.D. at the Technical University of Berlin, Germany. He teaches communications courses and his research interests include QoS, network performance, and traffic management. He is currently involved in funded research projects dealing with performance in IEEE 802.17 RPR, sensor networks, network security testbed and IP QoS architectures. He is a member of IEEE.

Ming-Wang Guo (郭明旺) received M.S. degree from the Department of Electrical Engineering, National Cheng Kung University, Taiwan. His current research interests include wireless ad hoc networks and high speed IP QoS networks design.