much when $per = 0{\cdot}001$, but increase remarkably when $per = 0{\cdot}01$, especially in the region larger than $u = 1{\cdot}0$. The volume of buffer memory to be prepared for resequencing becomes greater compared with the case of no transmission errors.

Conclusions: The probability of resequencing delay, the resequencing delay time, and the total delay time in multilink systems have been quantitatively calculated by simulation. It has been shown that they may increase remarkably when transmission errors occur over the link.

Y. YOSHIDA

Department of Electrical Engineering
Hosei University
Kajino-cho, Koganei-shi, Tokyo 184, Japan

14th September 1990

References

1 CCITT: 'Data communication networks: Services and facilities, Interfaces', Blue Book, Vol. VIII—Fascicle VIII.2 (Geneva, 1989), pp. 182–191
2 CCITT: 'Data communication networks: transmission and signalling and switching, network aspects, maintenance and administrative arrangements', Blue Book, Vol. VIII—Fascicle VIII.3, Geneva, 1989, pp. 175–185
3 NISHIZONO, T., and YOSHIDA, Y.: 'An analysis on multilink system', *Trans. IECE Japan*, 1984, **J66-B**, pp. 47–53

THROUGHPUT PERFORMANCE OF SOME CONTINUOUS ARQ SCHEMES WITH MEMORY

Indexing terms: Telecommunications, Data transmission, ARQ techniques, Throughput efficiency

A class of continuous ARQ schemes with memory, where multiple copies of each data block are sent contiguously instead of one single copy, and several copies of the same data block are combined to make a decision at the receiver, is considered. The throughput efficiency analysis is carried out based on the model proposed by Bruneel. Results show that the performance of the investigated ARQ schemes can be considerably improved if the receiver has some memory.

Introduction: Bruneel and Moeneclaey[2] analysed the throughput efficiency of a class of continuous ARQ schemes in which each data block is transmitted n_0 or less copies and, if necessary, retransmitted n_i or less copies in the ith retransmission to the receiver. This class of ARQ schemes preserves the ordering of data blocks just as in the classic go-back-N ARQ scheme. It is possible that an ACK for a copy of a given data block arrives at the transmitter before all the maximum allowed number of copies are sent. When this occurs, the transmitter will start to transmit the next data block instead of continuing to transmit the other copies of the data block. This is the reason the phase 'or less' is used in the above description. An error detection procedure is performed at the receiver on each received copy independently.[2]

In this letter, we allow the receiver to have some memory, i.e., the receiver can combine several copies to make a more reliable decision. The analytical model proposed by Bruneel[1] is used for performance evaluation. This analytical model is as follows: Whenever a data block is transmitted for the first time after no data blocks ahead of it will even be retransmitted, the erroneous decision probability is given by p. When two or more copies of the same data block are sent to the receiver, the information contained in all (or parts) of these copies can be collected to obtain a more reliable decision. In the model, this process results in a constant reduction factor, say r, of the erroneous decision probability for each additional received copy. In other words, the probability that the jth copy of a demodulated data block is erroneous given that its preceding $j-1$ copies are all erroneous is equal to pr^{j-1}, $0 < r \leq 1$. Notice that $r = 1$ corresponds to the memoryless case studied by Bruneel and Moeneclaey.[2] Small values of r can be achieved by using the soft error detection.[3,4]

Two different demodulation strategies proposed in Reference 1, namely the limited information strategy (LIS) and the full information strategy (FIS), are considered. In the case of LIS, the receiver makes a decision on a given data block by combining only the current set of copies of the data block regardless of all the information gained from the previous sets. The FIS combines all copies received thus far to make a decision on a given data block even though these copies were received from different sets of copies. Clearly, FIS requires more memory than LIS does. The throughput efficiency of both strategies will be derived and optimised, and compared with that of the corresponding schemes without memory studied by Bruneel and Moeneclaey in Reference 2.

Throughput evaluation and optimisation: Let $B(n_0, n_1, \ldots)$ denote the mean number of transmissions sent per correct data block delivered. The throughput efficiency of an ARQ scheme is $\eta = 1/B(n_0, n_1, \ldots)$. Let n_i^* denote the optimum value of the parameter n_i ($i \geq 0$). That is

$$B(n_0^*, n_1^*, \ldots) \leq B(n_0, n_1, \ldots) \qquad \forall(n_0, n_1, \ldots) \quad (1)$$

For simplicity, we assume that the round-trip delay is s, the number of data blocks that can be transmitted during the interval between the end of transmission of a copy and the receipt of its response is constant, and all the data blocks are of fixed length. Expressions for the quantity $B(n_0, n_1, \ldots)$ can be derived as follows:

LIS:

$$B(n_0, n_1, \ldots) = a(n_0) + p^{n_0} r^{n_0(n_0-1)/2}[s + B(n_1, n_2, \ldots)] \quad (2)$$

where

$$a(n) = \begin{cases} n & n \leq s+1 \\ s + \sum_{k=0}^{n-s-1} p^k r^{k(k-1)/2} & n \geq s+1 \end{cases} \quad (3)$$

One can easily prove that all the n_i^*s are equal to a common optimum value n^*. As a result, the optimum $B(n_0, n_1, \ldots)$ is given by

$$B(n^*, n^*, \ldots) = \frac{a(n^*) + p^{n^*} r^{n^*(n^*-1)/2} \cdot s}{1 - p^{n^*} r^{n^*(n^*-1)/2}} \quad (4)$$

FIS:

$$B(n_0, n_1, \ldots) = a_0(n_0) + p^{n_0} r^{n_0(n_0-1)/2}[s + B_1(n_1, n_2, \ldots)] \quad (5)$$

where

$$B_i(n_i, n_{i+1}, \ldots) = a_i(n_i) + p^{n_i} r^{n_i(2N_i+n_i-1)/2} \times [s + B_{i+1}(n_{i+1}, n_{i+2}, \ldots)] \quad (6)$$

$$a_i(n) = \begin{cases} n & n \leq s+1 \\ s + \sum_{k=0}^{n-s-1} p^k r^{k(2N_i+k-1)/2} & n \geq s+1 \end{cases} \quad (7)$$

and $N_i = \sum_{j=0}^{i-1} n_j$ ($i \geq 0$). Notice that all the n_i^*s are no longer equal to a common optimum value. In order to simplify the implementation, it is of interest to consider the special case when $n_0 = n_1 = \ldots \overset{\text{def}}{=} n$. For this special case, eqn. 5 reduces to

$$B(n, n, \ldots) = \begin{cases} (s+n)\sum_{j=0}^{\infty} p^{jn} r^{jn(jn-1)/2} - s & n \leq s+1 \\ s + \sum_{k=0}^{n-s-1} p^k r^{k(k-1)/2} \\ \quad + p^n r^{n(n-1)/2}[s + B_1(n, n, \ldots)] & n \geq s+1 \end{cases} \quad (8)$$

where

$$B_i(n, n, \ldots) = s + \sum_{k=0}^{n-s-1} p^k r^{k(2in+k-1)/2} + p^n r^{n(2in+n-1)/2}[s + B_{i+1}(n, n, \ldots)] \quad (9)$$

For convenience, we refer to the special case as the fixed FIS (FFIS). In the cases of both FIS and FFIS, it is difficult to obtain a further explicit expression of the optimum $B(n_0, n_1, \ldots)$, but we can still resort to computer searching.

Results and discussion: The results presented in Reference 2 can be found by substituting $r = 1$ into either eqn. 4 or 8. In the case of FIS (including FFIS), a practical computation of results requires an approximation of the infinite series in eqns. 5 and 8 by a finite sum.

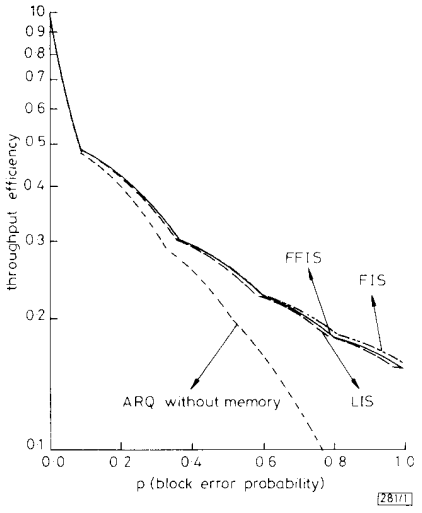

Fig. 1 *Throughput efficiency against p*
$s = 10; r = 0.8$

Figs. 1 and 2 show the throughput efficiency against p for $s = 10$, $r = 0.8$, and $s = 50$, $r = 0.6$, respectively. Here the throughput efficiency is achieved by using all the optimum values of n_i, and the logarithmic scale is used in the vertical axis of both Figures. It can be seen that, under a wide range of high error rate conditions, considerable improvements in throughput performance can be achieved in both cases if the receiver has some memory. One can also observe that as expected, FIS performs better than FFIS, and LIS is the most inefficient technique. The performance difference among FIS, FFIS and LIS can be neglected especially in environments with a large round-trip delay. The curve of the optimum number of copies as a function of p for $s = 20$ and $r = 0.8$ is plotted in Fig. 3. This Figure shows that the optimum copy number can be reduced significantly for high values of p by adding memory in the ARQ schemes. We conclude that LIS policy should be a good choice in practice.

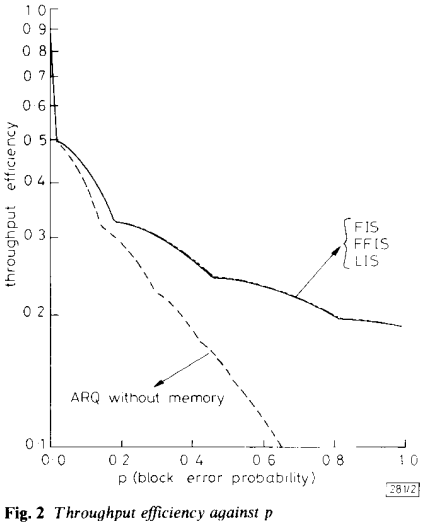

Fig. 2 *Throughput efficiency against p*
$s = 50; r = 0.6$

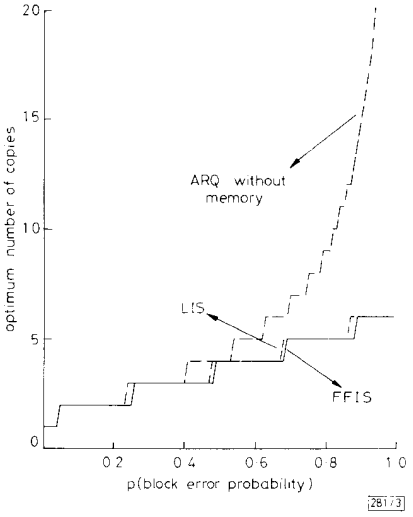

Fig. 3 *Optimum number of copies against p*
$s = 20; r = 0.8$

T.-H. LEE
J.-K. HU*

Department of Communications Engineering
**Institute of Electronics*
National Chiao Tung University
Hsinchu, Taiwan 30050
Republic of China

28th September 1990

References

1 BRUNEEL, H.: 'Throughput comparison for stop-and-wait ARQ schemes with memory', *Electron. Lett.*, 1988, **24**, pp. 531–533
2 BRUNEEL, H., and MOENECLAEY, M.: 'On the throughput performance of some continuous ARQ strategies with repeated transmission', *IEEE Trans.*, 1986, **COM-34**, pp. 244–249
3 FANTACCI, R.: 'Generalised stop-and-wait ARQ scheme with soft error detection', *Electron. Lett.*, 1986, **22**, pp. 882–883
4 LAU, C., and LEUNG, C.: 'Performance analysis of a memory ARQ scheme with soft decision detectors', *IEEE Trans.*, 1986, **COM-34**, pp. 827–832

SPLIT-RADIX FHT ALGORITHM FOR REAL-SYMMETRIC DATA

Indexing term: Transforms

A fast algorithm for computing the discrete Hartley transform of a real-symmetric data sequence is introduced. The number of computations required is significantly less than that required by the usual split-radix fast Hartley transform.

Introduction: A number of different algorithms have been proposed for computing the discrete Hartley transform (DHT) which are called fast Hartley transform (FHT) algorithms.[1,2]