Welch approach, and the models without adaptation data could also be adapted by discriminating information from all its competitive utterances.

*Experimental results:* The above adaptation approaches were evaluated with both the dialect adaptation and speaker adaptation experiments using the TIMIT corpus. ML-based stochastic matching [2] of model-space transformation was used as the benchmark for comparisons. The experiments were designed such that phonemes defined in the TIMIT database were recognised. Each phoneme was modelled with a hidden Markov model with at most three states, and each state consisted of three mixture Gaussian densities. The number of states in each HMM was determined by the time duration distribution of its phoneme provided in the TIMIT database.

**Table 1**: Dialect-adapted phoneme correctness/accuracy with different number of adaptation sentences for Baum-Welch (B_M), MMD and stochastic matching (SM) approaches

| | 50 training sentences | | 100 training sentences | | 150 training sentences | | 200 training sentences | |
|---|---|---|---|---|---|---|---|---|
| | Correctness | Accuracy | Correctness | Accuracy | Correctness | Accuracy | Correctness | Accuracy |
| | % | % | % | % | % | % | % | % |
| B_W | 75.62 | 61.98 | 77.32 | 64.48 | 77.41 | 65.29 | 77.85 | 66.0 |
| MMD | 76.43 | 62.78 | 77.65 | 65.1 | 78.8 | 66.19 | 79.27 | 66.9 |
| SM | 75.46 | 61.74 | 76.55 | 64.66 | 76.56 | 64.8 | 77.3 | 65.5 |

The speaker-independent (SI) recogniser was built from the 852 si and sx sentences selected from the Dr1, Dr3–Dr6 and Dr8 (under the training section) in the TIMIT. All si and sx sentences in the Dr7 of the training section were used for dialect adaptation training and testing. The SI recogniser achieved a correctness rate of 75.09% with 61.37% accuracy. Table 1 shows the test performances with different amounts of adaptation speech data. It shows that the MMD has the highest performance improvement. The phoneme error rate was reduced by 16.78%/5.38% for 200/50 sentences, and the accuracy error reduction was 14.32%/3.65%. The Baum-Welch method is also better than the stochastic matching method. For example, for 200 sentences, the correctness error reductions were 11.08 and 8.87% for the Baum-Welch and stochastic matching methods, respectively.

**Table 2**: Speaker-adapted phoneme correctness/accuracy with different number of adaptation sentences for Baum-Welch, MMD and stochastic matching approaches

| For speaker MDAB0 (SI: %correctness = 73.1, %accuracy = 66.31) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | One sentence | | Two sentences | | Three sentences | | Four sentences | | Five sentences | |
| | Correct-ness | Accuracy | Correct-ness | Accuracy | Correct-ness | Accuracy | Correct-ness | Accuracy | Correct-ness | Accuracy |
| | % | % | % | % | % | % | % | % | % | % |
| B_W | 74.18 | 68.48 | 76.35 | 71.81 | 76.78 | 72.15 | 79.82 | 74.65 | 81.12 | 76.48 |
| MMD | 77.65 | 70.64 | 78.58 | 73.14 | 80.04 | 74.65 | 82.85 | 77.87 | 83.93 | 79.48 |
| SM | 75.92 | 69.64 | 76.35 | 70.84 | 77.65 | 72.55 | 78.52 | 74.15 | 79.82 | 74.99 |
| For Speaker FMDG0 (SI: %correctness = 75.27, %accuracy = 60.83) | | | | | | | | | | |
| B_W | 76.91 | 62.7 | 78.78 | 67.47 | 81.22 | 69.96 | 83.65 | 74.12 | 86.22 | 81.59 |
| MMD | 77.44 | 64.98 | 80.67 | 69.55 | 82.84 | 72.87 | 87.97 | 82.42 | 91.48 | 88.45 |
| SM | 76.78 | 62.43 | 78.38 | 67.34 | 80.65 | 69.23 | 81.34 | 72.0 | 84.64 | 80.08 |

For speaker adaptation, we selected speaker MDAB0 in Dr1 and FMGD0 in Dr6 of the test section from the TIMIT database as the target speakers. Each speaker had 10 sentences. Five si sentences could be used as adaptation data and the performance was tested for all 10 sentences. The SI recogniser's performance achieved 73.1% correctness, 66.31% accuracy for speaker MDAB0 and 75.27% correctness, 60.83% accuracy for speaker FMGD0. The test performance with different adaptation data is shown in Table 2. As in dialect adaptation, these results show that the MMD leads to the largest performance improvement for any amount of adaptation speech data, and the Baum-Welch method performs slightly better than the stochastic matching method in most cases. For speaker MDAB0, when five si sentences were used the MMD reduced correctness errors by 40.26% and accuracy errors by 39.09%. The Baum-Welch method improved the correctness and accuracy errors by 29.81 and 30.19%, and 24.98, 25.76% for stochastic matching. Even when only one adaptation sentence was used, the MMD also improved the correctness and accuracy errors by 16.91 and 12.85%. For speaker FMDG0, 44.28, 65.55, 37.89% correctness error reduction and 53.0, 70.41, 49.14% accuracy error reductions were achieved for the Baum-Welch, MMD and stochastic matching methods, respectively.


6 August 1999
*Electronics Letters Online No: 19991273*
*DOI: 10.1049/el:19991273*

S. Kwong, K.F. Man and K.S. Tang (*City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong*)

Q.H. He (*South China University of Technology, People's Republic of China*)

E-mail: cssamk@cityu.edu.hk

**References**

1 GAUVAIN, J.L., and LEE, C.H.: 'Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains', *IEEE Trans. Speech Audio Process.*, 1994, **2**, pp. 291–298

2 SANKER, A., and LEE, C.H.: 'A maximum-likelihood approach to stochastic matching for robust speech recognition', *IEEE Trans. Speech Audio Process.*, 1996, **4**, pp. 190–202

3 DIAKOLOUKAS, D., and DIGALAKIS, V.: 'Maximum-likelihood stochastic-transformation adaptation of hidden Markov models', *IEEE Trans. Speech Audio Process.*, 1999, **7**, pp. 177–187

4 KWONG, S., HE, Q.H., MAN, K.F., and TANG, K.S.: 'A maximum model distance approach for HMM-based speech recognition', *Pattern Recogn.*, 1998, **31**, (3), pp. 219–229

5 HE, Q.H., KWONG, S., MAN, K.F., and TANG, K.S.: 'Improved maximum model distance for HMM training', *Electron. Lett.*, 1999, **35**, (10), pp. 783–785

## Start-up transition behaviour of TCP NewReno

Changhee Joo and Saewoong Bahk

NewReno has been proposed as the sender to recover multiple packet losses within a window by responding to a partial ACK. Since a great deal of time is required to recover all losses, the behaviour of the TCP sender during fast recovery greatly affects the overall performance. The dynamics of TCP NewReno variants during fast recovery are analysed.

*Introduction:* The transition control protocol (TCP) NewReno has been proposed to improve the performance by preventing unnecessary timeouts due to multiple packet losses from a single window of data during fast recovery [1 – 4]. TCP NewReno can recover multiple packet losses without any changes at the receivers, and has some variants according to the recovery procedures. With the increase in network bandwidth and the use of applications with short TCP connections, TCP start-up dynamics become important [1]. In this case, the overall performance is largely affected by procedures during fast recovery. In this Letter, we analyse the dynamics of TCP NewReno variants during fast recovery.

*TCP NewReno:* TCP NewReno has been proposed with many variants [4]. We have used the NewReno variants that have the following properties during fast recovery:
(i) Retransmit a single packet for each partial ACK
(ii) Reset the retransmit timer after receiving each partial ACK (slow-but-steady variant)
(iii) Avoid multiple fast retransmits (less careful variant)

One variant of TCP NewReno is whether the partial window deflation (PWD) algorithm is used or not. With the PWD algorithm, if a sender receives a partial ACK, TCP NewReno retransmits the first unacknowledged packet, then decreases the congestion window (*cwnd*) by the amount of new data acknowledged and adds one packet back. This algorithm attempts to prevent packet burstiness when fast recovery eventually ends by making an approximate slow-start threshold (*ssthresh*) amount of data outstanding in the network. Conversely, NewReno without PWD reduces the *cwnd* to the *ssthresh* for each partial ACK and inflates *cwnd* by one for each subsequent duplicate ACK until the next partial ACK is received. When NewReno (with or without PWD) receives an ACK, it sends a new packet if the new *cwnd* permits. We will analyse NewReno both with and without PWD.

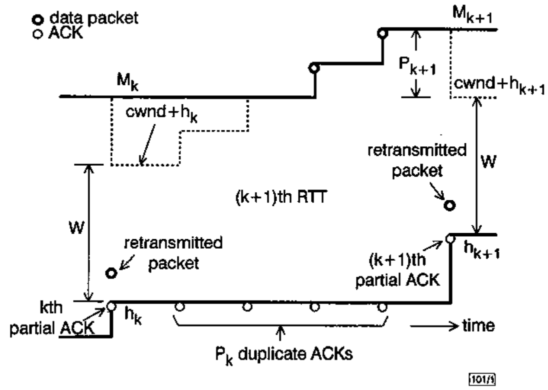


**Fig. 1** *NewReno behaviour without PWD during fast recovery*

*Analysis of transition behaviour:* Once a TCP connection is established, TCP usually makes the first phase transition from slow-start to congestion avoidance by a loss detection before the *cwnd* is over the *ssthresh*. The sender detects the loss by receiving three duplicate ACKs.

$W$ denotes the sender window size when the sender sends a packet that will be dropped first. Therefore when the sender detects the loss, i.e. after one round trip time (RTT), it will have a *cwnd* of approximately $2W$.

We have assumed the following for our analysis:
(i) $n$ (> 1) packets are dropped from a single window of $2W$
(ii) The receiver sends an ACK for each received packet
(iii) The advertised receiver window size is sufficiently large not to limit the sender window size ACKs and retransmitted packets are not dropped.

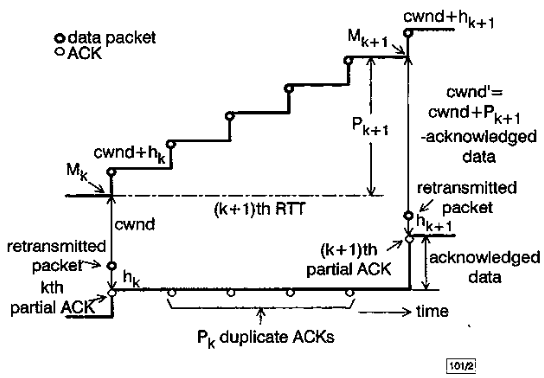


**Fig. 2** *NewReno behaviour with PWD during fast recovery*

*NewReno without PWD:* We define some variables for the analysis during fast recovery. $h_k$ is the highest sequence number acknowledged, and $M_k$ is the highest sequence number sent so far. Both are measured at the end of the $k$th RTT, which is just after the $k$th partial ACK is received. $P_k$ is the number of new packets sent during the $k$th RTT.

If the sender sends $P_k$ new packets during the $k$th RTT, the sender will receive $P_k$ duplicate ACKs during the $(k + 1)$th RTT, as shown in Fig. 1. After the sender receives the $k$th partial ACK, the sender sets $h_k$ to the acknowledged sequence number of the partial ACK (lower solid line), and *cwnd* to *ssthresh*, i.e. $W$. The *cwnd* is increased for each duplicate ACK received and can reach up to $W + P_k$ during the $(k + 1)$th RTT.

Whenever the sender receives an ACK, it checks whether the sum of *cwnd* and $h_k$ (dotted line) is above $M_k$ (upper solid line). If $(cwnd + h_k) > M_k$, the sender sends as many new packets as there are excess. Since *cwnd* is increased up to $W + P_k$ during the $(k + 1)$th RTT, the number of new packets sent is given by:

$$P_{k+1} = \max\left(0, (W + P_k) + h_k - M_k\right) \quad 1 \le k < n \quad (1)$$

Also $P_1$ can be obtained from eqn. 1. In this case $P_0$ is the number of duplicate ACKs during the first RTT, i.e. $2W - n$. Then $P_1 = \max(0, 3W - n + h_0 - M_0)$. Here $M_0 - h_0$ represents the window size when the drop is detected, i.e. $2W$. Hence, $P_1 = \max(0, W - n)$.

As $M_{k+1} = M_k + P_{k+1}$, we can write $M_k$ as the sum of $P_k, P_{k-1}, ..., P_1$, and $M_0$. $M_k = M_0 + \Sigma_{i=1}^{k} P_i = h_0 + W + \Sigma_{i=1}^{k} P_i$. Now we can rewrite eqn. 1 as

$$P_{k+1} = \max\left(0, (h_k - h_0) - W - \sum_{i=1}^{k-1} P_i\right) \quad 2 \le k < n$$
$$P_2 = \max\left(0, (h_1 - h_0) - W\right) \quad (2)$$

Since $h_k - h_0 < 2W$

$$P_k < W \qquad 1 \le k \le n \quad (3)$$

An important problem regarding multiple losses in NewReno is packet burstiness which occurs just after fast recovery ends. A burst with the size of *ssthresh* often causes a buffer overflow in the network and an unfairness problem with other existing connections. The burst size is determined by the number of packets outstanding in the network when a positive ACK is received (the positive ACK informs the sender that all packets sent before fast retransmit have been successfully transferred), i.e. $cwnd - P_n$, where *cwnd* is decreased to *ssthresh*, $W$. However, since we cannot obtain an exact $P_n$ in NewReno without PWD, we cannot estimate the burst size exactly. For the same reason, the total number of new packets sent during fast recovery cannot be obtained exactly, but we can obtain the maximum value. If $\Sigma_{i=1}^{k-1} P_i \ge W$, $P_{k+1} = 0$ from eqn. 2 and $h_k - h_0 < 2W$. Therefore we can choose a parameter $j$ such that $\Sigma_{i=1}^{j-1} P_i < W$ and $\Sigma_{i=1}^{j-1} P_i + P_j > W$, resulting in $P_{j+1} = \max(0, h_j - W - \Sigma_{i=1}^{j-1} P_i)$ and $P_i = 0$ for $i > j + 1$. If $P_{j+1} = 0$, $\Sigma_{i=1}^{n} P_i = \Sigma_{i=1}^{j-1} P_i + P_j < W + W < 2W$ from eqn. 3. If $P_{j+1} > 0$, $\Sigma_{i=1}^{n} P_i = P_j + h_j - W < h_j < 2W$. Hence

$$\sum_{i=1}^{n} P_i < 2W \quad (4)$$

*NewReno with PWD:* Fig. 2 shows the fast recovery behaviour of TCP NewReno with PWD. When the sender receives a partial ACK, it decreases *cwnd* by the amount of data acknowledged and increases it by one. This behaviour makes the sender increase the right (upper) edge of the window by one without affecting how much data is acknowledged by a partial ACK.

If non-zero $P_k$ new packets are sent during the $k$th RTT, $P_k + 1$ new packets will be sent during the $(k + 1)$th RTT, as shown in Fig. 2. This is because the sender increases *cwnd* by $P_k + 1$ due to $P_k$ duplicate ACKs and one partial ACK. But if $P_1 = 0$, the sender cannot send any new packets until the increased right edge of the window exceeds $M_k = M_0$. Hence, if $W + (2W - n) + k \le 2W$, $P_{k+1} = 0$, and otherwise $P_{k+1} = P_k + 1$, $1 \le k < n$.

Before the first partial ACK is received, the behaviours of TCP NewReno with and without PWD are identical. Hence $P_1$ is also $\max(0, W - n)$ for NewReno with PWD.

We can write $P_k$ as

$$P_k = \max(0, W - n + k - 1) \qquad 1 \le k \le n \quad (5)$$

Since $P_k$ has a maximum value at $k = n$, then $\max(P_k) = P_n = W - 1$. Therefore the PWD algorithm resolves the packet burstiness problem. As the sender transmits $P_n$ of $W - 1$ packets during the $n$th RTT, no packet burst occurs after fast recovery.

From eqn. 5, the total number of new packets that NewReno with PWD sends during fast recovery is given by

$$\sum_{i=1}^{n} P_i = \frac{1}{2}W(W - 1) - \max\left(0, \frac{1}{2}(W - n)(W - n - 1)\right) \quad (6)$$

*Conclusions:* We have analytically compared two popular variants of TCP NewReno in terms of their start-up transition behaviours. We have focused on the number of new packets sent during fast recovery, and the packet burstiness that may occur just after fast recovery ends. In NewReno with PWD, these values were determined by the number of drops. In NewReno without PWD, since these values varied with drop patterns, we could obtain the maximum values.


*1 September 1999*
*Electronics Letters Online No: 19991229*
*DOI: 10.1049/el:19991229*

Changhee Joo and Saewoong Bahk (*School of EE, Seoul National University, Seoul, Korea*)

E-mail: sbahk@netlab.snu.ac.kr

**References**

1 HOE, J.C.: 'Improving the start-up behavior of a congestion control scheme for TCP'. ACM SIGCOMM'96, 1996
2 FALL, K., and FLOYD, S.: 'Simulation based comparisons of Tahoe, Reno and SACK TCP', *Comput. Commun. Rev.*, 1996
3 ALLMAN, M., PAXSON, V., and STEVENS, W.: 'TCP congestion control'. RFC 2581, April 1999
4 FLOYD, S., and HENDERSON, T.: 'The NewReno modification to TCP's fast recovery algorithm RFC 2582, April 1999

# Upper bound on the bit error probability of TCM over frequency-selective Rayleigh fading channel

P. Tarasak and N. Rajatheva

A new upper bound on the bit error probability using Chernoff's bound of trellis code modulation (TCM) is introduced. It can be applied to a frequency-selective Rayleigh fading channel. The bound indicates a new design criterion, which is the product of the intersymbol interference (ISI) distance.

*Introduction:* In a frequency flat fading channel, the effective code length (ECL) and the minimum product distance (MPD) play an important role in the performance of TCM [1, 2]. For a frequency-selective fading channel, the performance of a combined equaliser-decoder has been shown in [3 – 5]. The upper bound was also discussed in [4, 5], but it was derived based on the complement error function and therefore could not provide further insight into the design methodology.

In this Letter, Chernoff's bounding technique is applied to pairwise error probability. The channel is modelled as an independent Rayleigh fading channel. This results in easily evaluated bit error probability.

*System model:* The output of the encoder is a sequence of coded symbols drawn from an alphabet of size $M$, denoted by $\mathbf{x} = (x_1, ..., x_k, ...)$ transmitted to the channel. The channel is assumed to be frequency-selective Rayleigh fading with a memory denoted by $L$. The discrete time channel model is employed. The received signal sequence is denoted by $\mathbf{y} = (y_1, ..., y_k, ...)$, where each $y_k$ can be represented by:

$$y_k = \sum_{j=0}^{L} g_{j,k} x_{k-j} + n_k \qquad (1)$$

where $n_k$ is complex white Gaussian noise with variance $2\sigma_N^2$. $g_{j,k}$ is a complex Gaussian variable with zero mean and variance $2\sigma_{gj}^2$ corresponding to path $j$, representing the channel impulse response of path $j$ at time $k$. They are assumed to be independent for each path $j$ and also for each time $k$.

The decoder performs maximum likelihood sequence decoding on the combined equaliser and the code trellis. The state of the combined trellis is given by [3]

$$\mu_k = (x_{k-L}, x_{k-L+1}, \ldots, x_{k-1}; \sigma_k) \qquad (2)$$

where $\sigma_k$ is the state of the encoder trellis. Hence, if the encoder trellis has $S$ states and the number of input bits per encoding interval is $m$, the combined trellis has $Sm^L$ states with $m$ transitions emerging from each state.

The branch metric evaluated in the combined trellis is

$$\lambda(x_k, y_k | g_{0,k}, g_{1,k}, \ldots, g_{L,k}) = \left| y_k - \sum_{j=0}^{L} g_{j,k} x_{k-j} \right|^2 \qquad (3)$$

The additive path metric is defined as

$$\Gamma(\mathbf{x}, \mathbf{y} | \mathbf{g}) = \sum_{k=1}^{N} \lambda(x_k, y_k | g_{0,k}, g_{1,k}, \ldots, g_{L,k}) \qquad (4)$$

The decoder selects the path with the minimum additive path metric as the decoded sequence.

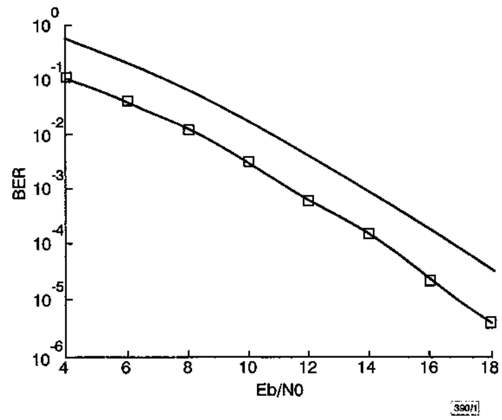


**Fig. 1** *Performance of four-state 8-PSK TCM with combined equaliser-decoder*

—□— simulation
——— upper bound

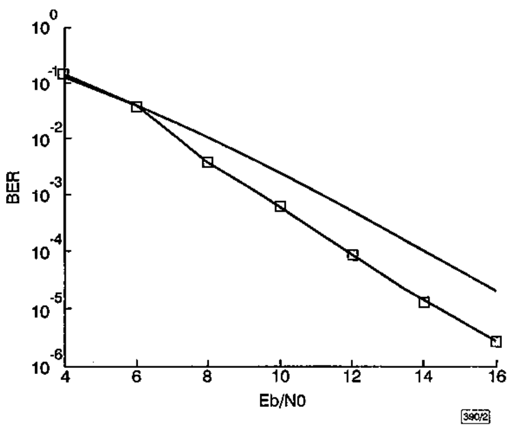


**Fig. 2** *Performance of eight-state 8-PSK TCM with combined equaliser-decoder*

—□— simulation
——— upper bound

*Performance analysis:* The error event of length $l$ is the interval where the incorrect path diverging from the transmitted path remerges with it in the combined trellis. The bit error probability can be estimated from [5]

$$P_b \approx \frac{1}{m} \sum_{l} b_{\mathbf{x}\to\hat{\mathbf{x}}}(l) P_{\mathbf{x}\to\hat{\mathbf{x}}}(l) \qquad (5)$$

where $m$ is the number of input bits per encoding interval, $P_{\mathbf{x}\to\hat{\mathbf{x}}}(l)$ is the pairwise error probability of error event of length $l$, i.e. the probability that the decoder selects the sequence $\hat{\mathbf{x}}$, and $b_{\mathbf{x}\to\hat{\mathbf{x}}}(l)$ is the number of erroneous bits along the event $\hat{\mathbf{x}} \neq \mathbf{x}$ of length $l$.