PRUNED CONVOLUTIONAL CODES AND VITERBI DECODING USING THE LEVENSHTEIN DISTANCE METRIC APPLIED TO ASYNCHRONOUS NOISY CHANNELS

L. Cheng and H.C. Ferreira

Dept. of Electrical and Electronic Engineering Science, University of Johannesburg, P O Box 524, Auckland Park, 2006, South Africa

Abstract: For a convolutional encoding and Viterbi decoding system, two insertion/deletion/substitution (IDS) error correcting techniques are presented in this paper. In the first means, by using the pruned convolutional codes, a rate compatible encoding system can adapt the transmission according to the state of the channel having IDS errors. In the second means, a convolutional encoded sequence is decoded by a modified Viterbi decoding algorithm using the Levenshtein distance metric, in which the IDS errors can be corrected at the same time.

Key words: Viterbi decoding, Levenshtein distance, Insertion/deletion/substitution, Pruned convolutional codes.

1. INTRODUCTION

Synchronisation is an important issue for the reliable communication. Insertion and deletion errors are used to model the synchronisation errors when bits are gained and lost between the source and the receiver at unknown positions in the bit sequence, respectively.

A fixed rate encoder is always used in an error control coding system. However, since the error protection needs are not fixed and the transmission channel condition is changing, a more flexible scheme is required, in which the sender can adapt the encoding to the varying channel state and the varying significance of source. Punctured convolutional codes are widely used in the rate compatible encoding applications mentioned above, in which the bits are periodically deleted from encoded sequence. It is evident the bandwidth efficiency is improved, since the puncturing can increase the code rate. On the other hand, the pruning technique is studied as an inverse operation of puncturing in the convolutional encoding system, in which the rate is decreased by deleting branches connecting state nodes in the trellis diagram. In [1] Brink et al. presented a possible implementation of the pruned convolutional codes for insertion/deletion error correction. Based on this work, we will present an encoding scheme which can correct random insertion/deletion/substitution errors at the same time.

As shown in Fig. 1, we present a rate compatible transmission scheme, in which the rate is varying to adapt to the source significance information (SSI) and the channel state information (CSI). As Fig. 1 shows, the source carries significance information, which can indicate different error protection needs. As well, the channel can provide state information to the encoder. In the scheme shown in Fig. 1, we change the code rate in transmission to adjust the error correcting capability according to the source and the channel needs. For practical purposes, in a variable rate transmission, we do not want to switch between a set of encoders and decoders. Instead, we implement only one encoder and one decoder which can be modified without changing their basic structures. Therefore, the pruned convolutional codes can be the ideal candidates for the rate compatible encoding system. In this scheme, the Viterbi algorithm (VA) is chosen to be implemented in the decoder.

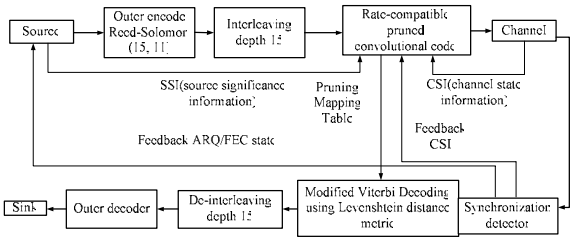


Figure 1: Rate compatible transmission scheme

In the second insertion/deletion/substitution error correcting technique presented in this paper, a modified Viterbi decoding algorithm using the Levenshtein distance metric is introduced. In [4], it has been proved that, in general, the Levenshtein distance (LD) metric can be applied in a VA decoding scheme. According to our research, VA can be used to detect the state of the channel and implement a hybrid ARQ/FEC scheme. Moreover, the channel state information detected in the VA can be used to adjust the rate of the encoder. In this decoding scheme, we use the minimum accumulated error (MAE) of the VA to construct a synchronization detector as shown in Fig. 1, by which the decoder can find out whether the disrupted received codeword can be corrected by the forward error correction (FEC).

Otherwise, the system needs to send an automatic repeat request (ARQ). At the same time, statistics of the rate of FEC/ARQ can be fed back to the encoder for the transmission rate control. Some related work can be found in [2].

The remainder of this paper is organized as follows. In Section 2, we will present the pruned convolutional codes. The Levenshtein distance metric and the modified Viterbi algorithm using this metric will be presented in Section 3. Section 4 concludes the paper with some results and discussions.

2. PRUNED CONVOLUTIONAL CODES

Punctured convolutional code is obtained by periodically deleting encoded bits from ordinary encoded sequences [3]. The aim is to obtain good convolutional codes in terms of the free distance. Inversely, we can periodically delete branches from the trellis diagram to obtain good convolutional codes. According to our research, the pruning on a convolutional code can be implemented by mapping the information and register states into the final input of the encoder. This mapping can result in a specified codeword structure, as well as a better free distance property, at the cost of decreased rate.

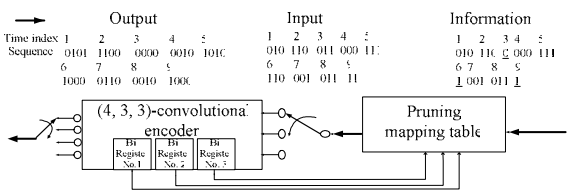


Figure 2: Example of a pruned convolutional encoding scheme. The underlined information bits are mapped according to Table 1

Table 1: Pruning mapping table

States	Information	Input
000	0	000
000	1	100
001	0	011
001	1	111
010	0	001
010	1	101
011	0	010
011	1	110
100	0	001
100	1	101
101	0	010
101	1	110
110	0	000
110	1	100
111	0	011
111	1	111

Example 1:

In order to clarify the pruned convolutional codes, we start with the example shown in Fig. 2, where a rate R = 3/4 convolutional code with memory m = 3 is pruned periodically. In this example, the encoder keeps the R = 3/4 rate in the first two intervals of every three intervals, and implements pruning operation only in the third interval. Thus, within every three intervals, six bits have been encoded at the original rate R = 3/4, and one bit has been encoded at the rate R = 1/4. As shown in Table 1, one bit is mapped into three bits and fed into the encoder. As shown in Fig. 2, as the least modification of an encoding system, the mapping component is implemented right before the original 3/4-rate convolutional encoder. It is evident that the pruning process reduces the code rate, and increases the free distance of the convolutional code.

3. VITERBI ALGORITHM USING LEVENSHTEIN DISTANCE METRIC

Definition 1:

Let X and Y be two finite sequences of symbols from a given alphabet. If X can be transformed into Y by substitution of K_i symbols, the insertion of M_i symbols, and the deletion of N_i symbols, then the Levenshtein distance (LD) [4] from X to Y is defined as:

$$LD(X,Y) = \min_i (K_i + M_i + N_i). \tag{1}$$

In Viterbi decoding schemes using the Hamming distance on an asynchronous noisy channel, it is impossible to continue decoding without re-synchronization. However, with the Levenshtein distance metric, Viterbi decoding can be used to keep synchronization, since the Levenshtein distance metric can be implemented for the variable length branch comparisons.

The regular Viterbi algorithm can be presented in a two dimensional trellis diagram, which is shown in Fig. 3.

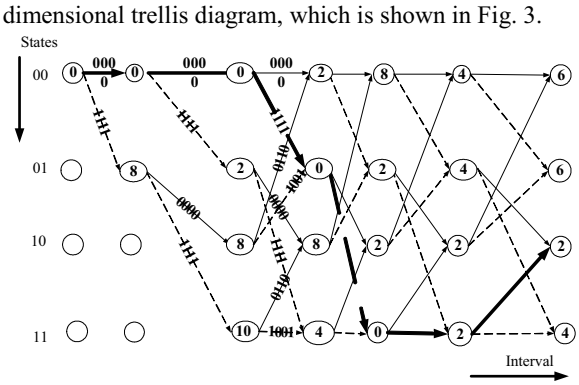


Figure 3: Trellis diagram of a regular Viterbi decoding algorithm

The horizontal coordinate represents the intervals, and the vertical coordinate represents the states. Since the regular Viterbi algorithm assumes the sequence is synchronized,

the branch comparisons of each interval starts from the same origin.

By using the Levenshtein distance metric, a variable length branch comparison metric, the Viterbi decoding algorithm can be presented in a three dimensional trellis diagram considering the loss of synchronization.

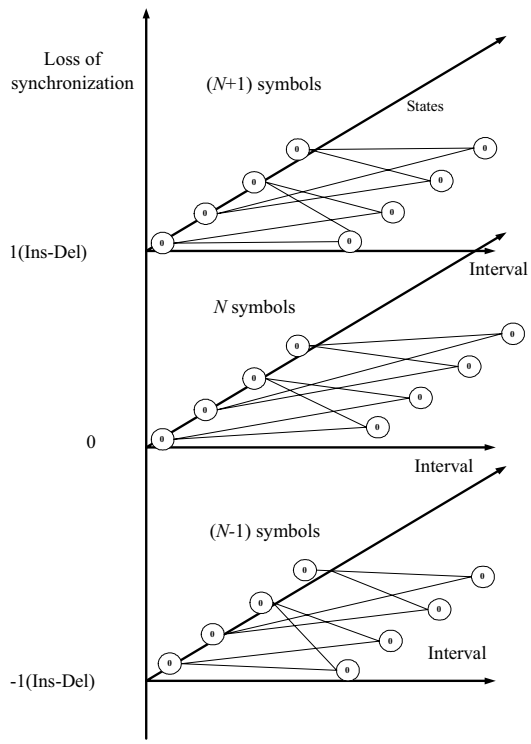


Figure 4a: Trellis diagram of Viterbi decoding algorithm considering loss of synchronization. Ins: number of insertion errors. Del: number of deletion errors

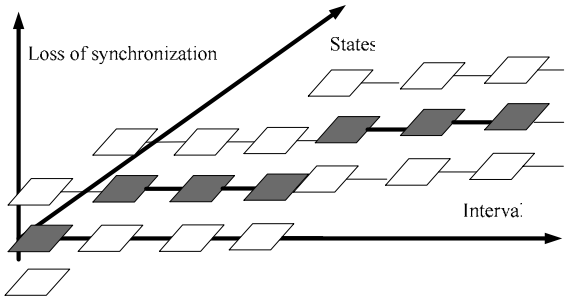


Figure 4b: Trace back of Viterbi decoding algorithm considering the loss of synchronization

As shown in Fig. 4a, the third coordinate quantifies the number of bit shifts, which is the net consequence of the insertion and deletion errors, in the received sequence. Given a value of bit shifts, the comparisons of the Viterbi decoding are between the fixed length trellis branches and a sequence of variable length. For instance, in the example shown in Fig. 4a, branch comparisons are implemented in three different trellis diagrams starting at the same origin. These trellis diagrams are named as the decoding planes. However, in each decoding plane, the sequence has different length corresponding to the value of the bit shifts. Let N denote the length of the codeword generated in every interval. In the 1(Ins-Del) plane, $(N + 1)$ received bits are compared with the branches in the trellis diagram. In the 0(Ins-Del) plane, N received bits are compared, and $(N - 1)$ received bits are compared in the -1(Ins-Del) plane. In the 1(Ins-Del) plane, since the received sequence in this interval is assumed to have one bit shift synchronization error, we therefore compare $(N + 1)$ bits with the branches. After the branch comparisons, in the add/compare/select step of the Viterbi algorithm, we need to choose the survivor path and the survivor plane, where the survivor decoding plane corresponds to the corrected synchronization. As well, the trace-back step will go through the survivor branches in the survivor planes. This step is shown in Fig. 4b.

The flowchart of the modified Viterbi decoding algorithm is shown in Fig. 5.

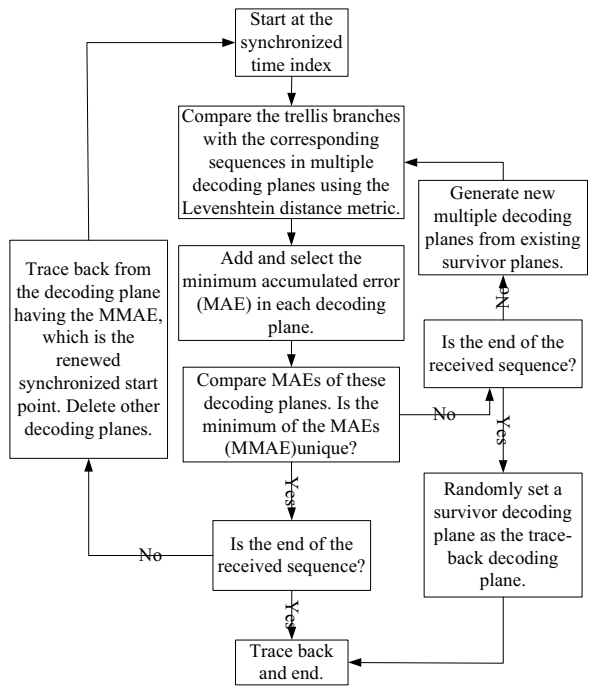


Figure 5: Flowchart of the modified Viterbi decoding algorithm using the Levenshtein distance metric

4. RESULTS

Computer simulations were used to evaluate the performance. The high rate convolutional encoder is a (4, 3, 3) optimum free distance (OFD) convolutional encoder. The pruning mapping table is shown in Table 1.

For convenience, our simulations only consider a simple binary symmetric channel (BSC) model with only substitution and deletion errors. Each packet of the encoded sequence has 3600 bits. We assume that the

boundaries of the packets are known. In practice, this assumption can be realized by using marker codes.

First, we define the following notations:
BER: bit error rate;
PLR: packet loss rate;
P_s: substitution error rate;
P_d: deletion error rate.

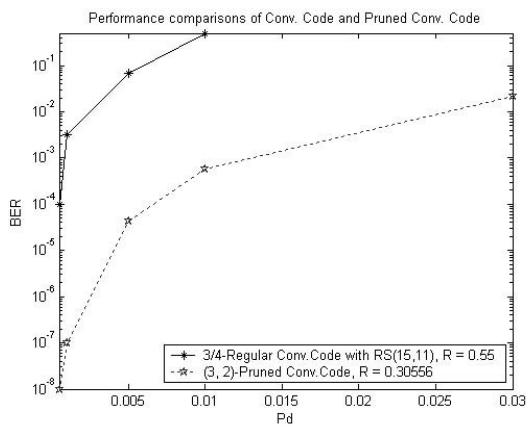


Figure 6: Decoding bit error rate comparisons of the regular convolutional code and the pruned convolutional code. $P_s = 0.001$, $P_d \geq 0$.

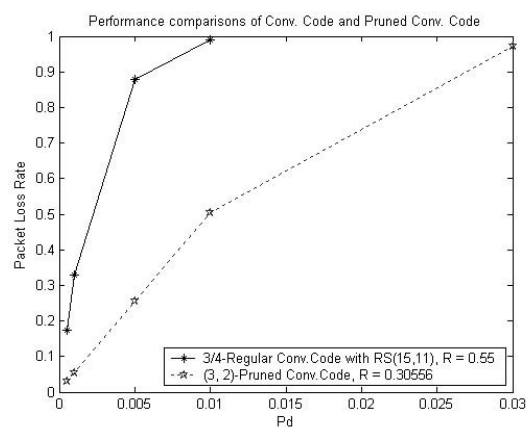


Figure 7: Packet loss rate comparisons of the regular convolutional code and the pruned convolutional code. $P_s = 0.001$, $P_d \geq 0$.

As Fig. 6 and Fig. 7 show, this scheme achieves good performance on channels with $P_d < 0.002$ and $P_s = 0.001$. These results prove that the modified Viterbi decoding algorithm based on the Levenshtein distance metric can be applied on channels with substitution and deletion errors. Meanwhile, the pruned convolutional codes can improve the performance of this scheme.

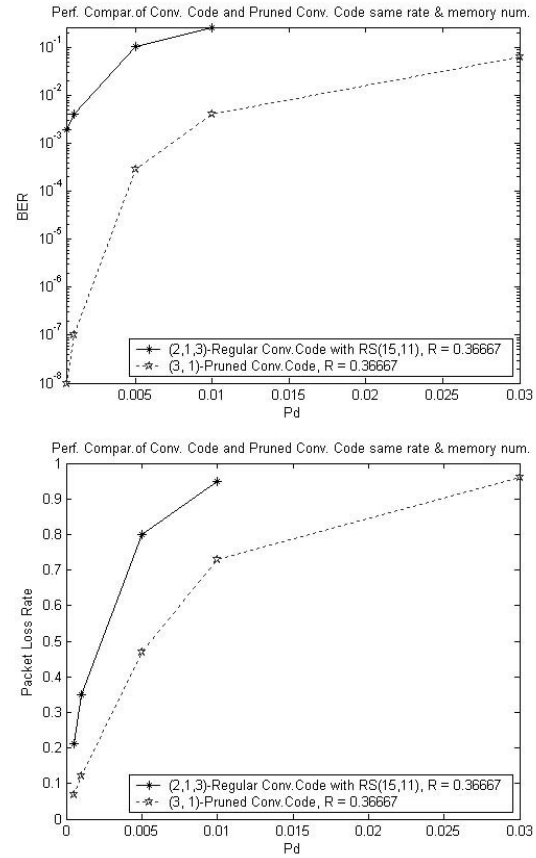
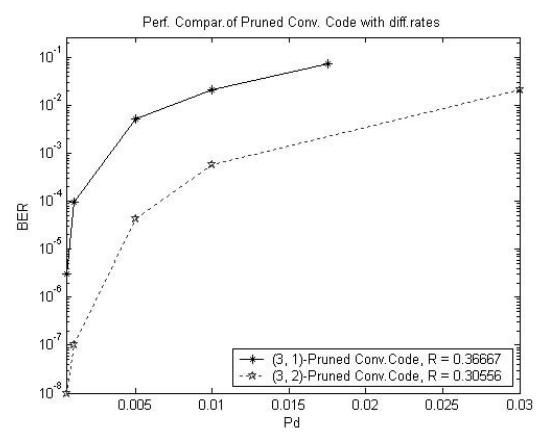


Figure 8: Performance comparisons of convolutional code and pruned convolutional code with the same rate and the same number of memories. The regular convolutional encoder is a (2, 1, 3) OFD. $P_s = 0.001$, $P_d \geq 0$.

According to the results shown in Fig. 8, we can conclude that the pruned convolutional codes obtain a better performance than regular OFD with the same rate and the same number of memories.



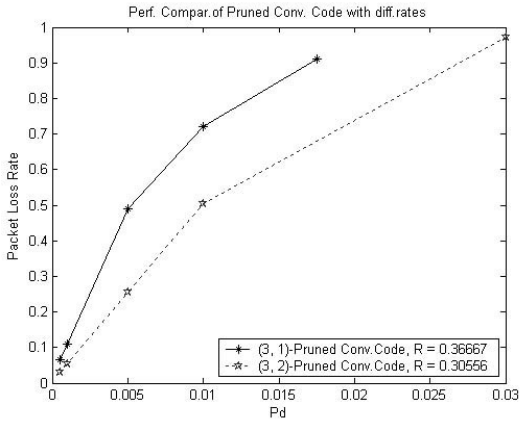


Figure 9: Performance comparisons of the pruned convolutional codes with different rates. $P_s = 0.001$, $P_d \geq 0$.

As shown in Fig. 9, the pruned convolutional codes with different pruning rate have different error correcting capabilities. It also shows that increasing pruning rate can improve the performance, and the pruned convolutional codes can be the optimum candidates for the rate compatible transmission schemes.

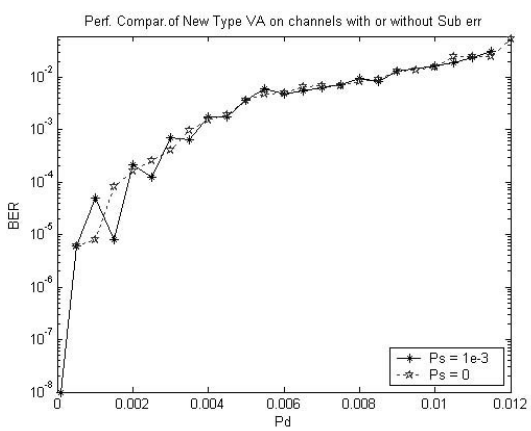


Figure 10: Performance comparisons on different channels. Solid line $P_s = 0.001$, $P_d \geq 0$; dotted line $P_s = 0$, $P_d \geq 0$.

Fig.10 shows that the performance of decoding on a channel without substitution errors is very close to that of a channel with substitution errors. Since the Levenshtein distance metric is used as the branch comparison metric in the decoding algorithm, the substitution errors are treated equally as the insertion/deletion errors.

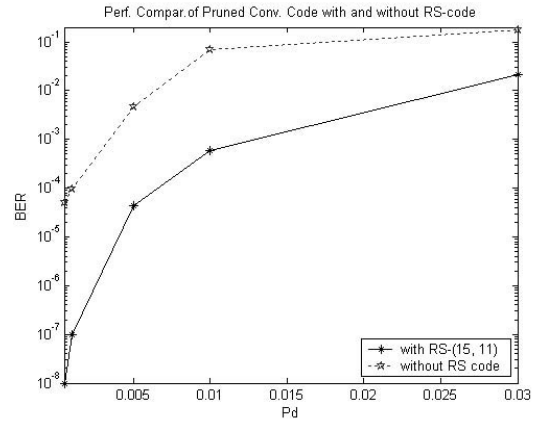


Figure 11: Performance comparisons of Reed-Solomon codes. $P_s = 0.001$, $P_d \geq 0$.

Fig. 11 shows that the RS codes improve the decoding performance significantly.

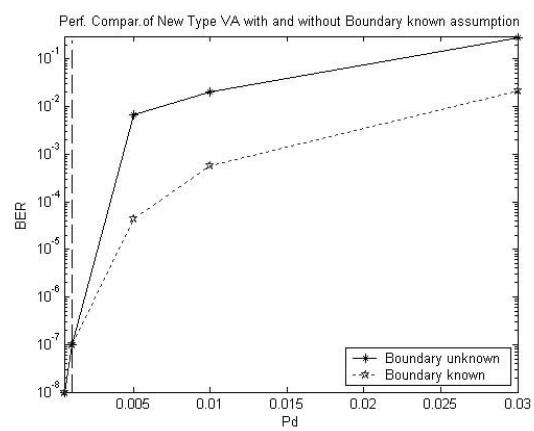


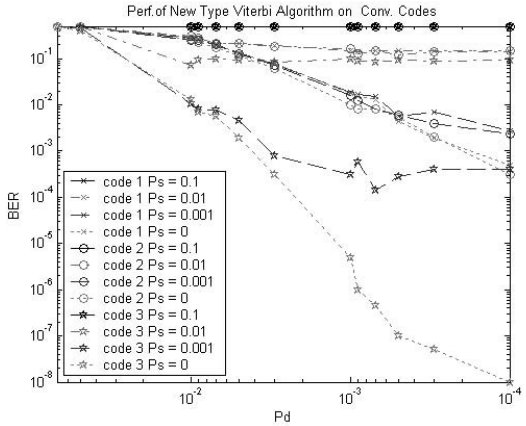
Figure 12: Performance comparisons of boundary assumption. $P_s = 0.001$, $P_d \geq 0$.

Fig. 12 shows, when $P_d < 0.001$, that decoding scheme can detect the boundaries of packets independently.

Table 2: Convolutional codes used in the simulations

Index	n	K	m	$G_p(D)$
Code1	2	1	2	$[1+D^2, 1+D+D^2]$
Code2	2	1	3	$[1, 1+D+D^3]$
Code3	4	1	3	$[1+D^2+D^3, 1+D+D^3, 1+D+D^3, 1+D+D^2+D^3]$

This modified Viterbi decoding algorithm can also work independently on the channels with insertion/deletion/substitution errors. Some results on the regular convolutional codes listed in Table 2 are given in Fig. 13.



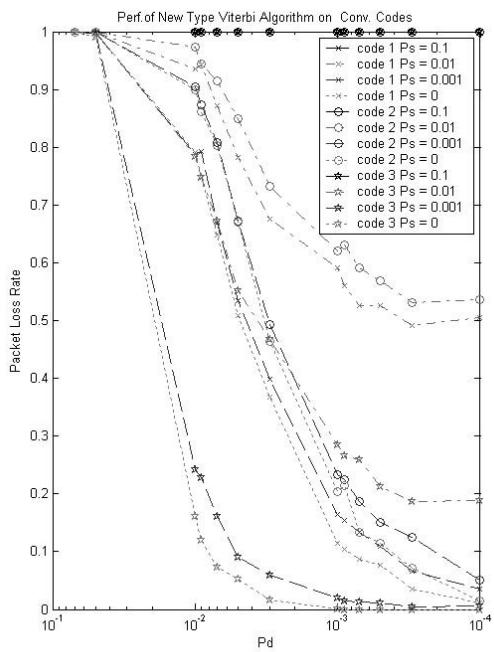


Figure 13: Performance comparisons of the modified Viterbi decoding algorithm on different regular convolutional codes (packet boundary unknown).

At the end of this section, we show the differences of the modified Viterbi decoding algorithm and the regular VA on the channels only having substitution errors. We can see from Fig. 14 that, while $P_s > 0.0005$, the performance of the modified Viterbi decoding algorithm is inferior to the regular one. Fig. 15 shows that the decoding time of the modified Viterbi decoding algorithm is greater than that of the regular one. These are the tradeoffs of using the modified Viterbi decoding algorithm.

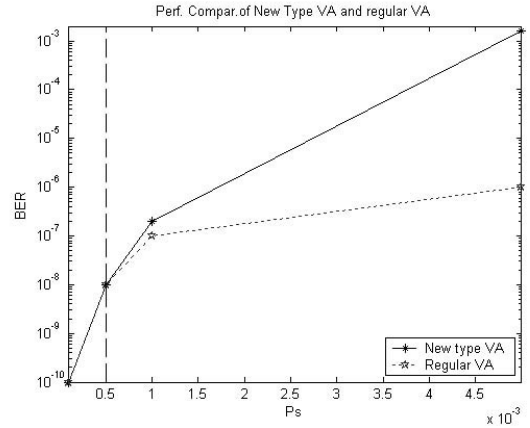


Figure 14: Performance comparisons of the modified Viterbi decoding algorithm and the regular VA. $P_s \geq 0$, $P_d = 0$.

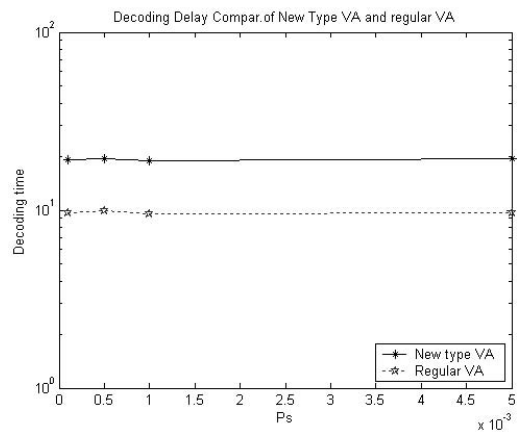


Figure 15: Decoding time comparisons of the modified Viterbi decoding algorithm and the regular VA. $P_s \geq 0$, $P_d = 0$.

5. REFERENCES

[1] B Brink, HC Ferreira and WA Clarke, "Pruned convolutional codes for flexible unequal error protection against insertion/deletion/reversal errors", Proceedings of the IEEE International Symposium on Information Theory, Sorrento, Italy, pp. 260, June 25-30, 2000

[2] M dos Santos, WA Clarke and HC Ferreira, "Correction of Insertions/Deletions using Standard Convolutional Codes and the Viterbi Algorithm", Proceedings of the 2003 IEEE Information Theory Workshop, Paris, France, pp. 187-190, March 31-April 4, 2003.

[3] J. Hagenhauer, "Rate-compatible punctured convolutional codes and their applications," IEEE Transactions on Communications, Vol. 36, No. 4, pp. 389-400, April 1988.

[4] T. Mori, H. Imai, "Viterbi Decoding Considering Insertion/deletion Errors", Proc. ISIT '95, Whistler, B. C. Canada, pp. 145, Sept. 1995.