Re-Synchronization of Permutation Codes with Viterbi-Like Decoding

Ling Cheng, Theo G. Swart and Hendrik C. Ferreira
Department of Electrical and Electronic Engineering Science
University of Johannesburg, P.O. Box 524,
Auckland Park, 2006, South Africa
E-mail: lcheng@uj.ac.za, tgswart@uj.ac.za, hcferreira@uj.ac.za

***Abstract*— In this paper, we present a fast re-synchronization algorithm for permutation coded sequences. The new algorithm combines the dynamic algorithm and a Viterbi-like decoding algorithm for trellis codes. The new algorithm has a polynomial time complexity $\mathcal{O}(N)$, where N denotes the length of the sequence. A possible application to M-ary FSK for the CENELEC A band power-line communications (PLC) is considered.**

I. Introduction

Synchronization is a crucial issue for reliable communication. Most of the conventional communications systems are designed to work under very low signal-to-noise-ratios (SNR) and at high transmission rates, which demand more reliable timing recovery techniques to cooperate. In most cases, it is achieved with techniques such as phase-locked-loops (PLLs) or preamble sequences. Nevertheless, these techniques also have some drawbacks. For example, PLL needs a considerable delay and heavy dependence on the precision of hardware. Another obvious drawback of PLLs and preamble sequences is that both can only detect synchronization errors, thus two-way communication is always necessary so lost information can be sent again.

It is important to investigate new synchronization techniques that can overcome the drawbacks mentioned above. It prompted the research to combine coding techniques with conventional equalization and timing recovery [1]. In the previous work [2], we presented a self-synchronizable coding scheme that combines coding and modulation by using permutation codes. However, this coding scheme has a major drawback in that it has a decoding complexity $\mathcal{O}(N^2)$, where N denotes the length of sequences. In this paper, we present a new decoding algorithm that reduces the time complexity to $\mathcal{O}(N)$. The new algorithm has a polynomial time complexity, which normally is considered as an efficient algorithm. Moreover, we show a reliable coding scheme that can correct a combination of insertion, deletion and substitution errors. The simulation results shows this scheme can provide a decoding block error rate below 10^{-7} with block throughput rate 98.464%, when the channel has insertion, deletion and substitution error rates at 10^{-3} respectively.

The paper is organized as follows: To make this paper more self-contained, a brief introduction to the permutation codes and its application in M-ary FSK modulation is presented in Section II. In the same section, we will recall some existing works on insertion/deletion correction with permutation codes. A Viterbi-like algorithm in conjunction with the insertion/deletion/substitution error metric of permutation codes is presented in Section III. The system model of the simulation is stated in Section IV, as well as the simulation results. We conclude the paper with Section V.

It is worth mentioning that M-ary FSK modulation is preferred for narrow-band PLC in the CENELEC A band in our case, however the results will also be applicable to other M-ary schemes such as pulse amplitude modulation (PAM). Much research has been done in the field of broadband PLC, but little documentation can be found in the low frequency range (below than 100 kHz). In this range communication is considered with a low rate that can provide very high accuracy, for applications such as automatic meter reading and demand side management.

II. Preliminaries

A. Permutation Codes and M-FSK Modulation

The definition for a permutation code is as follows:

Definition 1: A permutation code $\mathcal{C}$ consists of $|\mathcal{C}|$ code words of length M, where every code word contains the M different integers $1, 2, \ldots, M$ as symbols.

Permutation codes combined with M-ary FSK modulation have previously been considered to combat additive noise, impulse noise and permanent frequency disturbances. In [3] and [4] the authors show that they are good candidates for narrow-band PLC. Some performance simulations for these codes can be found in [5].

In the M-ary FSK system every symbol corresponds uniquely to a frequency from an M-FSK modulator and the M-ary symbols are then transmitted in time as the corresponding frequencies. A more detailed explanation of the system and how different types of noise on the power-line affects it, can be found in [3] and [4].

Since decoding of permutation codes can be difficult in this scenario, an approach is used whereby the convolutional code's error correcting capabilities are mapped to the permutation codes. In [4] it is described how permutation trellis codes can be created by using distance-preserving mappings. Briefly, the outputs of a binary convolutional encoder are mapped to the code words from a permutation code in such a way that the Hamming distance between any two permutation sequences

978-1-4244-3790-0/09/$25.00 ©2009 IEEE

is at least as large as the distance between the corresponding convolutional code's output sequences which they are mapped to. This results in a permutation trellis that can be decoded using the Viterbi algorithm.

In this paper we will focus on insertion/deletion errors, and the insertion/deletion error correcting capability of the permutation codes is investigated. A real-time synchronization scheme is designed, based on the permutation codes, which can correct single insertion/deletion errors in each code word. This limits the error propagation and reduces the delays of the resynchronization process.

Using some of the properties that we will discuss shortly, permutation trellis codes can be designed that have insertion/deletion correcting capabilities as well.

B. Insertion and Deletion

Most of the investigations on error correcting codes only consider additive errors. Nevertheless, when messages are transmitted through an asynchronous channel, the message received may have a different size compared to that of the message sent. Thus, insertion and deletion errors are defined for a channel having synchronization problems.

Definition 2: An insertion is the transform whereby one symbol is added at an unknown index in the message during transmission, which results in the increase of the message size by one.

Definition 3: A deletion is the transform whereby one symbol is dropped off the message during transmission, which results in the decrease of the message size by one.

An example of insertion/deletion channels in the context of the PAM system can be found in [2]. Evidently, even a single insertion/deletion error can give rise to severe error propagation.

C. Permutation Codes and Insertion/Deletion Correction

By using the properties of the permutation codes, we can detect or correct insertion/deletion errors. Constructing a subset of permutation codes by using the rules of Tenengolts' nonbinary single insertion/deletion correcting codes [6], we obtain single insertion/deletion correcting permutation codes. The cardinalities of these codes are $(M-1)!$ [2]. Nevertheless, the error correcting capability of a single insertion/deletion and the known boundary assumption still limit the code to a research interest only. In [2], the authors provide an algorithm to resynchronize a permutation coded sequence with multiple insertion/deletion/substitution errors. The algorithm is based on a dynamic algorithm, which has a decoding complexity of $\mathcal{O}(N^2)$. Whereas the authors showed that insertion/deletion/substitution error correction is possible with permutation codes in [2], no results were provided.

III. Viterbi-Like Algorithm

A. Lattice Representation

A stochastic insertion/deletion/substitution channel can be illustrated by a lattice diagram. In Fig. 1, we show symbols over a channel can be affected by three types of errors, i.e., insertion, deletion or substitution. Each vertex in the graph represents a time index. Vertically, the vertices have the same time indices. Three edges departing from a vertex represent symbols/sequences affected by three different types of errors respectively. There are multiple paths connecting any two vertices. These paths represent the possible symbols/sequences that occur over an insertion/deletion/substitution channel.

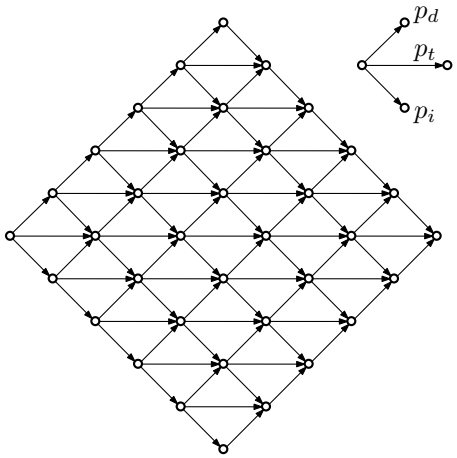

Fig. 1. Lattice diagram for the insertion/deletion/substitution channels

As shown in Fig. 1, the arrows pointing towards the top right corner represent possible deletions, the arrows pointing towards the bottom right corner represent possible insertions, and the horizontal arrows represent symbols/sequences that are insertion/deletion error-free. The idea of the lattice representation of insertion/deletion/substitution channel was first presented in [7]. It was first used to show a method to correct garbled words based on the Levenshtein metric. Inspired by this work, the decoding of permutation sequences can also be presented on a modified lattice graph.

B. Error Function on Permutation Sequence

For permutation sequences affected by insertion, deletion or substitution errors, an error function $d(\mathbf{x})$ can be defined to quantify the number of errors [2].

We first need to define a function $u(\mathbf{x})$ to count the number of unique symbols in a sequence, where $\mathbf{x} = x_1x_2\ldots x_i\ldots x_n$, $x_i \in \mathcal{A}$ and $\mathcal{A} = \{a_1, a_2, \ldots, a_M\}$ is the alphabet of the permutation code.

Definition 4: For an alphabet $\mathcal{A}$ of a permutation code and a sequence $\mathbf{x} \in A^n$, let

$$u(\mathbf{x}) := |\{j|a_j = x_i\}|, \tag{1}$$

where $1 \le i \le n$, $1 \le j \le M$ and $|\cdot|$ denotes the cardinality of a set.

Then we can define a function to quantify the number of errors of a corrupted permutation code word as follows:

Definition 5: Let $\mathbf{x}' = x'_1x'_2\ldots x'_{M'}$ be a corrupted sequence of $\mathbf{x} = x_1x_2\ldots x_M$ as a result of substitution, deletion and insertion errors, the minimum error function $d(\mathbf{x}')$ is defined as

$$d(\mathbf{x}') := \begin{cases} M' - u(\mathbf{x}'), & \text{for } M' \ge M, \\ M - u(\mathbf{x}'), & \text{for } M' < M. \end{cases} \tag{2}$$

C. Modified Decoding Algorithm

In the original lattice model [2], in each code word we consider there are probabilities to have one insertion, deletion or substitution error. When the number of code words increments, so does the number of errors that the algorithm can correct. Nevertheless, the original algorithm has a time complexity of $\mathcal{O}(N^2)$.

If we assume the probabilities of the insertion, deletion and substitution error over a channel are i.i.d., and each type of error has a probability p, the probability that a sequence of length n has more than s errors can be shown as follows:

$$\Pr\{\#\text{Errors} > s\} = 1 - \sum_{i=0}^{s} \binom{n}{i} (1-p)^{n-i} p^i. \tag{3}$$

When n is large, it has a Poisson distribution such that

$$\Pr\{\#\text{Errors} = s\} = \frac{\lambda^s e^{-\lambda}}{s!}, \tag{4}$$

where $\lambda = np$.

We have

$$\Pr\{\#\text{Errors} > s\} = 1 - \sum_{i=0}^{s} \Pr\{\#\text{Errors} = i\}. \tag{5}$$

It is evident that the probability of large s could be very small. When

$$\Pr\{\#\text{Errors} = s\} \gg \Pr\{\#\text{Errors} = s+1\}, \tag{6}$$

the probabilities of number of errors more than s can be ignored.

According to this result, we can start to present a modified algorithm that can provide polynomial time complexity of $\mathcal{O}(N)$ as follows. (Fig. 2 shows the indices that will be used in the algorithm.)

Algorithm 1: If the sequence $\mathbf{x}$ is received within T intervals, a sequence of length MT was sent.

Step 1: Given a design bit error rate ϵ of a reliable transmission system, according to (4) we obtain the first value of s that satisfies (7), when s increments,

$$\frac{\lambda^s e^{-\lambda}}{s!} \leq \epsilon. \tag{7}$$

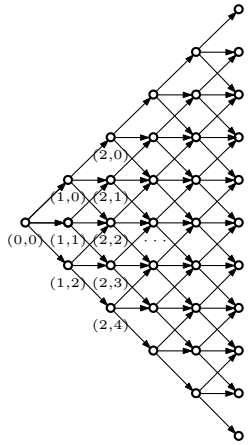

Fig. 2. Indices of a truncated lattice

Step 2: Let $v_{t,j}$ denote a vertex. Here t is the time index that increments horizontally, and j indexes the vertices vertically in the lattice graph. Obviously, we can use t and j to index any vertex uniquely.

Then define

$$v_{t,j} := (l_{t,j}, e_{t,j}), \tag{8}$$

where $l_{t,j}$ denotes the index of the vertex in the $t-1$'th (last) interval which gives rise to the least accumulated errors $e_{t,j}$ at vertex $v_{t,j}$. Note that, according to the structural property of the lattice graph, for any $v_{t,j}$, we have $0 \leq j \leq 2t$. For the trellis part of the graph, in other words when $t \geq s$, we have $0 \leq j \leq 2s$.

We define functions $\alpha(\cdot)$, $\beta(\cdot)$ and $\gamma(\cdot)$ to facilitate the description as follows:

$$\begin{aligned}
\alpha(t,j) &:= e_{t-1,j-1} + d(x_{(t-1)(M-1)+j-1} \dots x_{t(M-1)+j-1}) \\
\beta(t,j) &:= e_{t-1,j} + d(x_{(t-1)(M-1)+j} \dots x_{t(M-1)+j-1}) \\
\gamma(t,j) &:= e_{t-1,j-2} + d(x_{(t-1)(M-1)+j-2} \dots x_{t(M-1)+j-1})
\end{aligned} \tag{9}$$

Note that the $\alpha(\cdot)$ function is used to calculate the error accumulation when the last block received has no insertion or deletion error, the $\beta(\cdot)$ function is used in the case that the last block has a deletion error, and the $\gamma(\cdot)$ function is used when the last block has an insertion error.

We furthermore define the functions $\Lambda(\cdot)$ and $\Delta(\cdot)$ as follows:

$$\Lambda(t,j) := \min(\alpha(t,j), \beta(t,j), \gamma(t,j)). \tag{10}$$

$$\Delta(t,j) = \begin{cases} -1, & \text{if } \Lambda(t,j) = \alpha(t,j), \\ 0, & \text{if } \Lambda(t,j) = \beta(t,j), \\ -2, & \text{if } \Lambda(t,j) = \gamma(t,j), \end{cases} \tag{11}$$

when $0 \leq t \leq s$. For $s < t \leq T$,

$$\Delta(t,j) = \begin{cases} 0, & \text{if } \Lambda(t,j) = \alpha(t,j), \\ 1, & \text{if } \Lambda(t,j) = \beta(t,j), \\ -1, & \text{if } \Lambda(t,j) = \gamma(t,j). \end{cases} \tag{12}$$

Here $\Lambda(t,j)$ compares the error accumulations from three connected vertices and chooses the minimum error accumulation at the vertex $v_{t,j}$, and the index offset is given by $\Delta(t,j)$ accordingly. Note that if more than one condition in (11) or (12) is satisfied, we can choose one of them randomly.

Initialization:

$$\begin{aligned}
v_{0,0} &= (0,0) \\
v_{1,0} &= (0, d(x_0 x_1 \dots x_{M-2})) \\
v_{1,1} &= (0, d(x_0 x_1 \dots x_{M-1})) \\
v_{1,2} &= (0, d(x_0 x_1 \dots x_M))
\end{aligned}$$

Iteration: Repeat the following steps for $t = 2$ to s:

1) $l_{t,0} = 0$;
2) $e_{t,0} = e_{t-1,0} + d(x_{(t-1)(M-1)} \dots x_{t(M-1)-1})$;
3) $l_{t,1} = 1 + \Delta(t,1)$, where $\Lambda(t,1) = \min(\alpha(t,1), \beta(t,1))$;
4) $e_{t,1} = \Lambda(t,1)$;
5) $l_{t,2t-1} = 2t - 1 + \Delta(t, 2t-1)$, where $\Lambda(t, 2t-1) = \min(\alpha(t, 2t-1), \gamma(t, 2t-1))$;

6) $e_{t,2t-1} = \Lambda(t, 2t-1)$;
7) $l_{t,2t} = 2(t-1)$;
8) $e_{t,2t} = e_{t-1,2(t-1)} + d(x_{(t-1)(M+1)} \cdots x_{t(M+1)-1})$;
9) $l_{t,j} = j + \Delta(t,j)$, when $2 < j < 2t-1$;
10) $e_{t,j} = \Lambda(t,j)$, when $2 < j < 2t-1$;

Repeat the following steps for $t = s+1$ to T:

1) $l_{t,0} = \Delta(t,0)$, where $\Lambda(t,0) = \min(\alpha(t,0), \beta(t,0))$;
2) $e_{t,0} = \Lambda(t,0)$;
3) $l_{t,2s} = 2s + \Delta(t,2s)$, where $\Lambda(t,2s) = \min(\alpha(t,2s), \gamma(t,2s))$;
4) $e_{t,2s} = \Lambda(t,2s)$;
5) $l_{t,j} = j + \Delta(t,j)$, when $1 \le j \le 2s-1$;
6) $e_{t,j} = \Lambda(t,j)$, when $1 \le j \le 2s-1$;

Trace-back: At the interval T, we find $e_{T,j_{\min}} = \min(e_{T,0}, e_{T,1}, \ldots, e_{T,2s})$. According to $l_{T,j_{\min}}$ of the survivor vertex $v_{T,j_{\min}}$, trace-back through the lattice and obtain the shortest path.

As shown in Fig. 3, the new Viterbi-like decoding algorithm is actually a combination of the original lattice decoding with trellis decoding. It is evident that the new algorithm has a time complexity close to $\mathcal{O}(N)$, if $s \ll N$.

IV. Simulation and Performance

A. Channel Model

It is worth to specify the channel model we used in the computer simulations.

The Davey-MacKay channel model [8] can be illustrated by Fig. 4. At interval t_i, the sent symbol has a probability of p_d to be deleted. This symbol cannot reach the t_{i+1} interval. At interval t_i, there is also a probability of p_i that a random symbol is inserted, and a probability of p_t that the symbol is transmitted. However, after the symbol is transmitted, the probability of error-free transmission is $1 - p_s$. We have

$$p_i + p_d + p_t = 1, \tag{13}$$

and thus the probability of error-free transmission from the t_i interval to t_{i+1} is $p_t(1-p_s)$.

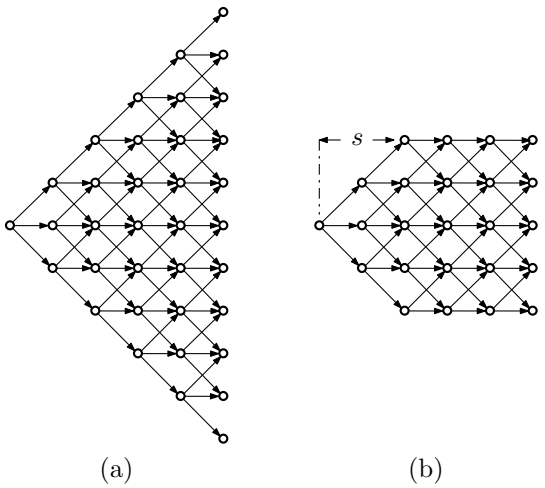

Fig. 3. Comparison of (a) original lattice algorithm, and (b) new trellis Viterbi-like decoding

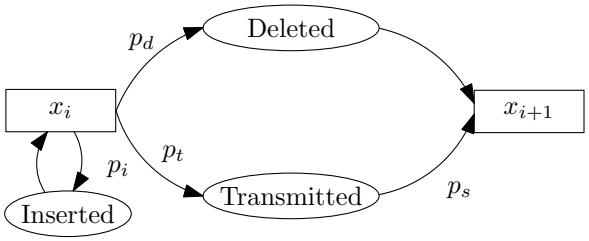

Fig. 4. Davey-MacKay insertion/deletion/substitution channel model

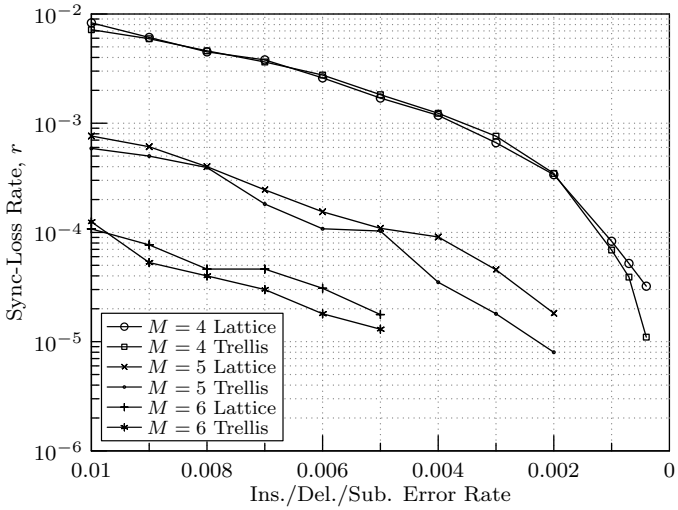

Fig. 5. Comparisons of re-synchronization rate of lattice decoding and trellis decoding

B. Results

We use computer simulations to demonstrate the performance of the new algorithm. We first compare the re-synchronization performance of the new algorithm with the lattice decoding. As shown in Fig. 5, compared to the lattice decoding, the trellis Viterbi-like decoding has no significant loss in terms of re-synchronization loss.

The comparisons of the decoding time for different sequence lengths are shown in Table I. The comparisons for different values of s are illustrated by Fig. 6.

Moreover, in conjunction with a low-density parity-check (LDPC) outer code, the new algorithm shows that reliable transmission is possible for a hybrid ARQ/FEC system. As shown in Fig. 7, for a channel with a combination of insertion/deletion/substitution error rate at 3×10^{-3}, the decoding block error rate is below 10^{-7} with throughput rate 98.464%. In this ARQ/FEC system, we use a (1023, 781)-LDPC code as the outer code. The message sent has a CRC-32b ($EDB88320$) check. The block lengths for $M = 4$ and $M = 5$ are 4096 and 5115 respectively. In Fig. 8, we further show that when the insertion/deletion/substitution error rate reaches 3×10^{-4}, the packet-loss rate of the transimmision decreases to 1.64×10^{-4} with decoding block error rate below 10^{-7}.

V. Conclusion

In this paper, a trellis Viterbi-like decoding algorithm is investigated. The new algorithm has a polynomial time com-

TABLE I

COMPARISONS OF DECODING TIMES OF LATTICE DECODING WITH TRELLIS DECODING FOR $s = 10$

Length	Lattice t [sec.]	Trellis t [sec.]
100	2.174e-3	1.421e-3
200	9.658e-3	3.842e-3
300	2.453e-2	6.139e-3
400	4.808e-2	9.627e-3
500	8.128e-2	1.283e-2
600	1.524e-1	2.077e-2
700	2.199e-1	2.561e-2
800	3.151e-1	3.236e-2
900	4.102e-1	4.028e-2
1000	5.379e-1	4.361e-2

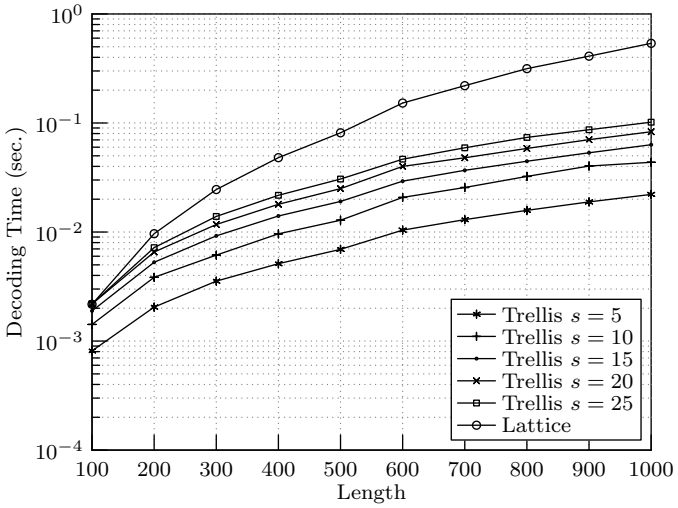

Fig. 6. Comparisons of decoding times for lattice decoding and trellis decoding: Microsoft Windows XP SP2, Intel P4 2.80GHz 1GB RAM, and VS6.0 C++

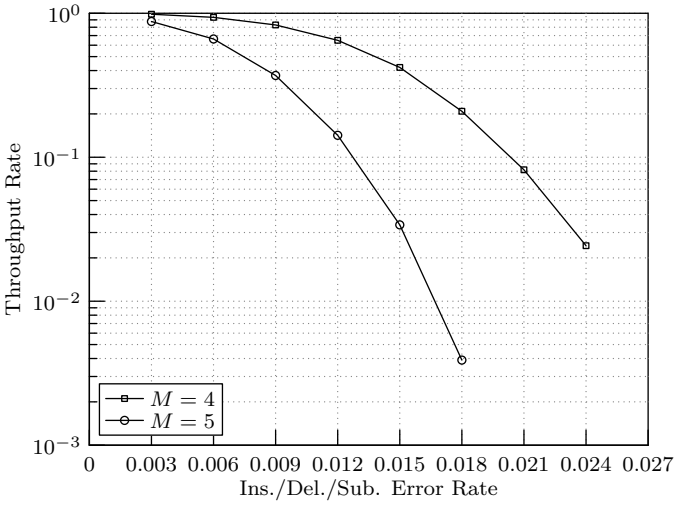

Fig. 7. Throughput rate performances as a function of the insertion/deletion/substitution error rate

plexity. Performances of the new algorithm based on computer simulations are provided. For a hybrid ARQ/FEC scheme, in conjunction with a LDPC outer code, the permutation coding system can provide for a reliable transmission.

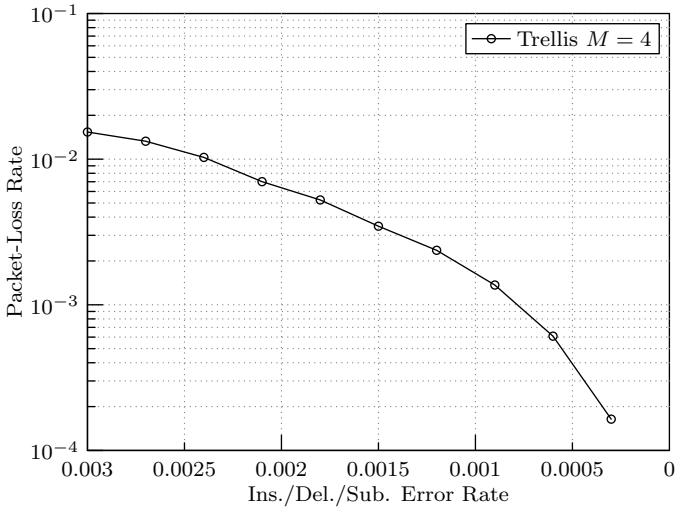

Fig. 8. Packet-loss rate performances as a function of the insertion/deletion/substitution error rate

REFERENCES

[1] J. R. Barry, A. Kavcic, S. W. McLaughlin, A. Nayak, and W. Zeng, "Iterative timing recovery," *IEEE Signal Processing Mag.*, pp. 89–102, Jan. 2004.

[2] L. Cheng, T. G. Swart, and H. C. Ferreira, "Synchronization using insertion/deletion correcting permutation codes," in *Proc. Int. Symp. on Power Line Commun. and its Applications*, Jeju Island, Korea, Apr. 2–4, 2008, pp. 135–140.

[3] A. J. H. Vinck, "Coded modulation for powerline communications," *Proc. Int. J. Elec. Commun.*, vol. 54, no. 1, pp. 45–49, 2000.

[4] H. C. Ferreira, A. J. H. Vinck, T. G. Swart and I. de Beer, "Permutation trellis codes," *IEEE Trans. Commun.*, vol. 53, no. 11, p. 1782–1789, Nov. 2005.

[5] T. G. Swart, I. de Beer, H. C. Ferreira, and A. J. H. Vinck, "Simulation results for permutation trellis codes using M-ary FSK," in *Proc. Int. Symp. on Power Line Commun. and its Applications*, Vancouver, Canada, Apr. 6–8, 2005, pp. 317–321.

[6] G. M. Tenengolts, "Nonbinary codes, correcting single deletion or insertion," *IEEE Trans. Inform. Theory*, vol. 30, no. 5, pp. 766–769, Sept. 1984.

[7] T. Okuda, E. Tanaka, and T. Kasai, "A method for the correction of garbled words based on the Levenshtein metric," *IEEE Trans. Comput.*, vol. c-25, no. 2, pp. 172–176, Feb. 1976.

[8] M. C. Davey and D. J. C. MacKay, "Reliable communication over channels with insertions, deletions and substitutions," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 687–698, Feb. 2001.