Irregular Puncturing for Convolutional Codes and the Application to Unequal Error Protection

Chung-Hsuan Wang, Shih-Chieh Wang, and Yun-Liang Chang
Department of Communication Engineering, National Chiao Tung University
Hsinchu 30056, Taiwan
E-mail: chwang@mail.nctu.edu.tw

***Abstract*— In this paper, convolutional codes are studied for puncturing with irregular puncturing periods. Irregular puncturing can generate punctured codes with more available rates and better bit-error-rate performance compared with the conventional scheme with a single puncturing period. For the application to unequal error protection, a new multiplexing scheme is also proposed for rate-compatible punctured convolutional (RCPC) codes which can guarantee smooth transition between rates without extra overheads. Finally, families of good RCPC codes with irregular puncturing tables are given by a computer search.**

I. Introduction

Punctured convolutional codes were first introduced in [1] by periodically deleting some coded bits of ordinary convolutional codes. Later in [2], the puncturing process was further regulated by a rate-compatible criterion to guarantee smooth transition between different rates. Owing to flexible choices of code rates and ease for decoding all children* codes by a single decoder of their parent code, rate-compatible punctured convolutional (RCPC) codes have been extensively employed in various applications. However, literatures on puncturing were based on the scenario with a single puncturing period for all output streams of convolutional encoders. Although good RCPC codes have been searched in [1]–[5], only a few of rates are available for the code families with small puncturing periods, and not all of them can achieve the optimal free distances as the general convolutional codes with the same memories and code rates.

To improve the puncturing performance, in this study, we generalize the conventional scenario by choosing irregular puncturing periods for different encoder outputs. Given a parent code with a maximum period for puncturing, irregular puncturing can generate children codes with rates and free distances which are unobtainable by the conventional puncturing. Combining both of the puncturing schemes, we can hence construct code families with more flexible choices of code rate and more powerful error-correcting capability. In addition, irregular puncturing with small periods can be shown to achieve similar puncturing effect as the conventional scheme with large periods. Our design thus provides a low-complexity alternative for searching good high-rate punctured convolutional codes which are originally obtainable by conventional puncturing with extremely large periods.

*For convenience, the original convolutional code to be punctured is called the parent code and the resulting punctured code is called the child code.

For the application to unequal error protection (UEP), irregular puncturing is further combined with the rate-compatibility to guarantee smooth transition between different children codes. A new multiplexing scheme is then presented for RCPC codes which can achieve similar UEP performance as the conventional one in [2] but requires no additional zero-padding for packet termination. Finally, families of good RCPC codes with irregular puncturing tables are given by a computer search.

The rest of this paper is organized as follows. Puncturing with irregular periods is described in Section II. In Section III, we study irregular puncturing with the rate-compatibility and its application to UEP. Remarks are then given in Section IV to conclude this work.

II. Puncturing with Irregular Periods

Consider an $(n,\ k)$ parent code $\boldsymbol{C}$ with $c_{i,t}$ denoting the coded bit of the ith output stream of encoder at time t, $\forall\ 0 \le i < n$. Conventionally, a puncturing table $\boldsymbol{A}$ with puncturing period p is defined to be an $n \times p$ matrix with the (u, v)th entry $a_{u,v}$ taking value from $\{0, 1\}$. Suppose $\boldsymbol{C}$ is said to be punctured by $\boldsymbol{A}$. It follows that $c_{i,t}$ is allowed for transmission if $a_{i,t \bmod p} = 1$; otherwise, $c_{i,t}$ is deleted from the output stream. Accordingly, children codes with the following code rates can be obtained:

$$kp/(kp+l),\ \forall\ 1 \le l \le (n-k)p. \tag{1}$$

If a small p is used for puncturing, only a few of rates are available by (1). However, suppose $c_{i,t}$'s are allowed for puncturing with irregular periods as described below; more choices of code rate can be provided even with small puncturing periods.

Let $p_0,\ p_1,\ \cdots,\ p_{n-1}$ be the puncturing periods corresponding to n output streams of the encoder. The irregular puncturing table $\boldsymbol{A}$ is defined in a similar way as above except that its ith row now consists of p_i columns, $\forall\ 0 \le i < n$, and hence $c_{i,t}$ is transmitted only if $a_{i,t \bmod p_i} = 1$. Let $\phi_0,\ \phi_1,\ \cdots,\ \phi_{n-1}$ be the numbers of non-zero entries in rows of $\boldsymbol{A}$. In general, the child code generated by $\boldsymbol{A}$ has code rate

$$k/\sum_{i=0}^{n-1}\frac{\phi_i}{p_i}. \tag{2}$$

For example, consider a (2,1) parent code $\boldsymbol{C}$ with the following codeword matrix:

$$\begin{pmatrix} c_{0,0} & c_{0,1} & c_{0,2} & c_{0,3} & c_{0,4} & c_{0,5} & c_{0,6} & c_{0,7} & c_{0,8} & c_{0,9} & c_{0,10} & c_{0,11} & \cdots \\ c_{1,0} & c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} & c_{1,5} & c_{1,6} & c_{1,7} & c_{1,8} & c_{1,9} & c_{1,10} & c_{1,11} & \cdots \end{pmatrix}$$

in which the (u,v)th entry indicates the coded bit of the uth encoder output at time v. Puncturing $\boldsymbol{C}$ with a single period $p=4$ can only generate children codes of code rate

$$4/5,\ 2/3,\ 4/7,\ \text{and}\ 1/2 \tag{3}$$

by (1). However, suppose $\boldsymbol{C}$ is now punctured by

$$\boldsymbol{A}=\left(\begin{array}{cccc} 1 & 0 & 1 & \\ 0 & 1 & 1 & 1 \end{array}\right) \tag{4}$$

with (p_0,p_1) = $(3,4)$ and (ϕ_0,ϕ_1) = $(2,3)$. The codeword matrix after puncturing by $\boldsymbol{A}$ turns to be

$$\left(\begin{array}{cccccccccccc} c_{0,0} & \times & c_{0,2} & c_{0,3} & \times & c_{0,5} & c_{0,6} & \times & c_{0,8} & c_{0,9} & \times & c_{0,11} & \cdots \\ \times & c_{1,1} & c_{1,2} & c_{1,3} & \times & c_{1,5} & c_{1,6} & c_{1,7} & \times & c_{1,9} & c_{1,10} & c_{1,11} & \cdots \end{array}\right) \tag{5}$$

where coded bits marked by $\times$ are deleted from transmission. The resulting child code has code rate 12/17 by (2). Moreover, for all possible irregular puncturing tables with (p_0,p_1) = $(3,4)$, the available rates of children codes are

$$12/13^*,\ 6/7^*,\ 4/5,\ 3/4^*,\ 12/17^*,\ 2/3,\ 3/5^*,\ 4/7,\ \text{and}\ 1/2$$

where the rates marked by $*$ are unavailable in (3). (12/13, 6/7, and 12/17 are even unattainable by (1) for all $p \le 4$.) Note that we have $\max(p_0,p_1)$ = p in this case. Compared with the conventional puncturing, it thus requires no extra hardware/computation overheads to implement the puncturing process with irregular periods.

In addition, as shown in Example 1, irregular puncturing may generate children codes with the optimal free distances as general convolutional codes which are unobtainable by the conventional puncturing with the same puncturing complexity. Combining both of the puncturing schemes, we can hence construct code families with more flexible choices of code rate and more powerful error-correcting capability even for the case of small puncturing periods.

Example 1: Consider a parent code $\boldsymbol{C}$ with generator matrix $\left[D^4+D+1 \;\; D^4+D^3+D^2+1\right]$ (i.e., [23 35] in octal). Based on the conventional puncturing with $p=6$, the optimal rate-3/4 child code of free distance 3 is obtained by puncturing $\boldsymbol{C}$ with [3]

$$\boldsymbol{A}_1=\left(\begin{array}{cccccc} 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \end{array}\right).$$

However, suppose $\boldsymbol{C}$ is punctured by

$$\boldsymbol{A}_2=\left(\begin{array}{cccccc} 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & & \end{array}\right)$$

with irregular periods (p_0,p_1) = $(6,4)$, which requires the same puncturing complexity as $\boldsymbol{A}_1$ since $\max(p_0,p_1)$ = p. The resulting child code also has code rate 3/4 but surprisingly achieves a larger free distance 4, which is the same as the optimal free distance that general $(4,3)$ convolutional codes with memory 4 can provide [6]. ∎

Recall the irregular puncturing table in (4). Repeating its first row four times and the second row three times, we obtain the following puncturing table with $p=12$:

$$\left(\begin{array}{cccc} 1 & 0 & 1 & \\ 0 & 1 & 1 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{cccccccccccc} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \end{array}\right). \tag{6}$$

Suppose a (2,1) parent code is punctured by the above puncturing table; the consequent child code is equivalent to the punctured code with the codeword matrix in (5). In general, it can be shown that an irregular puncturing table $\boldsymbol{A}$ with periods $(p_0,\ p_1,\ \cdots,\ p_{n-1})$ is equivalent to the conventional puncturing table with a period of the least common multiple of $p_0,\ p_1,\ \cdots,\ p_{n-1}$ (denoted by $lcm(p_0,p_1,\cdots,p_{n-1})$) whose rows comprise copies of the corresponding rows of $\boldsymbol{A}$ as in (6). Therefore, puncturing a parent code with small irregular periods can achieve the same puncturing effect as the conventional scheme with large periods, which also explains why our design may perform better under the condition of $\max(p_0,p_1,\cdots,p_{n-1}) \le p$.

Besides, most of good RCPC codes in the literature are provided with small periods ($p \le 8$) since a direct search of the puncturing tables with large periods usually incurs huge computational complexity far beyond what a practical system can afford. However, a specially designed puncturing table with irregular periods (8,7) can achieve similar puncturing effect as the conventional scheme with a large period of 56 as observed above. Suppose a length-n parent code is to be punctured in the conventional way with a large period p. We can then search the puncturing tables with irregular periods $(p_0,p_1,\cdots,p_{n-1})$ under the constraints that $p_i \le p\ \forall\ 0 \le i < n$ and $lcm(p_0,p_1,\cdots,p_{n-1}) \ge p$ instead, to reduce the search complexity. Irregular puncturing thus provides a feasible alternative for searching good high-rate punctured codes which are originally obtainable by the conventional puncturing with extremely large periods.

III. The Application to UEP

Consider W groups of source data S_l's, each of the required bit error rate (BER) $P_{b,l}$; assume $P_{b,1} \ge P_{b,2} \ge \cdots \ge P_{b,W}$ without loss of generality. To provide UEP for S_l's by irregular puncturing, we first choose a proper parent code together with puncturing tables $\boldsymbol{A}(l)$'s of periods $(p_0,p_1,\cdots,p_{n-1})$ to generate a family of children codes $\hat{\boldsymbol{C}}_l$'s, each of free distance $d_f(\hat{\boldsymbol{C}}_l)$ to satisfy $P_{b,l}$. ($\boldsymbol{A}(l)$'s should also be carefully selected to avoid generating catastrophic encoders for $\hat{\boldsymbol{C}}_l$'s.) $\boldsymbol{A}(l)$ is then switched for puncturing as S_l is fed to the encoder. In this way, S_l could be protected by $\hat{\boldsymbol{C}}_l$ for all l, thus fulfilling the desire for UEP.

To guarantee smooth transition between different rates, we further demand $\boldsymbol{A}(l)$'s to satisfy the following equivalent rate-compatible criterion for irregular puncturing:

$$\begin{array}{l} \text{if}\ \ a_{u,v}(i)=1,\ \text{then}\ a_{u,v}(j)=1, \\ \qquad \forall\ 0 \le u < n,\ 0 \le v < p_u,\ 1 \le i < j \le W \end{array} \tag{7}$$

where $a_{u,v}(i)$ denotes the (u,v)th entry of $\boldsymbol{A}(i)$. Suppose $\boldsymbol{A}(l)$ is switched for puncturing during the interval $[\hat{t}_l,\tilde{t}_l]$; we also require $c_{u,t}$ to be processed according to the value of $a_{u,t \bmod p_u}(l)$, instead of $a_{u,(t-\hat{t}_l) \bmod p_u}(l)$, for all $\hat{t}_l \le t \le \tilde{t}_l$. Under the above restrictions, it implies that all the coded bits of high-rate punctured codes are embedded in the lower rate codes. Consequently, we have $d_f(\hat{\boldsymbol{C}}_i) \le d_f(\hat{\boldsymbol{C}}_{i+1})$ and all codewords across the switching boundary between $\boldsymbol{A}(i)$

and $\boldsymbol{A}(j)$ will have a distance $\min(d_f(\hat{\boldsymbol{C}}_i), d_f(\hat{\boldsymbol{C}}_j))$ at least, $\forall\ 1 \le i < j \le W$. (Please refer to Appendix for the detailed proofs.) Based on the rate-compatible criterion, some RCPC codes with irregular puncturing tables which provide with better performance than conventional punctured codes are listed in Table I. Besides, the proof in Appendix can be generalized to show that UEP performance can still be guaranteed even if puncturing tables with single and irregular periods are alternatively switched for puncturing. RCPC codes searched previously [1]–[5] can hence be combined with our results to provide better performance for UEP.

Moreover, in [2], S_l's are suggested to be grouped into super frames before encoding as the conventional multiplexing scheme depicted in Fig. 1 (a). In this scheme, S_l is followed by S_{l+1} to achieve the minimum loss of free distance, i.e., $d_f(\hat{\boldsymbol{C}}_{l+1}) - d_f(\hat{\boldsymbol{C}}_l)$, for all $1 \le l < W$; extra (all-zero) tail bits are inserted at the end of every super frame to avoid the abrupt switching from $\boldsymbol{A}(W)$ to $\boldsymbol{A}(1)$. However, as revealed in Appendix, we can show that the distance between any two codewords will still be lower bounded by $d_f(\hat{\boldsymbol{C}}_l)$ no matter the puncturing tables are switched from $\boldsymbol{A}(l)$ to $\boldsymbol{A}(l+1)$ or from $\boldsymbol{A}(l+1)$ to $\boldsymbol{A}(l)$. Suppose S_l's are multiplexed by the new proposed scheme in Fig. 1 (b), where no tail bits are required but S_l's are multiplexed in a reverse order for alternate super frames. Accordingly, the puncturing tables are restricted to switch either from $\boldsymbol{A}(l)$ to $\boldsymbol{A}(l+1)$ or from $\boldsymbol{A}(l+1)$ to $\boldsymbol{A}(l)$ for all $1 \le l < W$; the same distance loss as the conventional multiplexing scheme could be obtained even without additional overheads. To verify the superiority of our design, we further simulate both of the multiplexing schemes for additive white Gaussian noise channels with binary phase-shift keying modulation. As expected, the new scheme is observed to provide almost the same UEP performance as the conventional one from the BER curves in Fig. 2.

IV. Conclusion

In this paper, convolutional codes are studied for puncturing with irregular periods. Compared with the conventional puncturing, irregular puncturing not only can provide more choices of code rates but also may attain better BER performance under the same puncturing complexity. Since irregular puncturing with small periods can achieve similar puncturing effects as the conventional scenario with large periods, our design also provides a practical alternative for searching conventional puncturing tables with extra large puncturing periods to construct good high-rate punctured convolutional codes. In addition, we devise a new multiplexing scheme for UEP which can minimize the possible performance loss during the transition phase between puncturing tables but requires no extra overheads. Not only for irregular puncturing, the proposed multiplexing scheme is also applicable to all the punctured systems with rate-compatibility.

Appendix

Consider a parent code $\boldsymbol{C}$ which is punctured by W puncturing tables $\boldsymbol{A}(l)$'s satisfying the rate-compatible criterion in (7) to generate a family of rate-compatible children codes $\hat{\boldsymbol{C}}_l$'s, each of free distance $d_f(\hat{\boldsymbol{C}}_l)$, $\forall\ 1 \le l \le W$. In Section III, the following distance properties of $\hat{\boldsymbol{C}}_l$'s for UEP are left unproved:

1) $d_f(\hat{\boldsymbol{C}}_l) \le d_f(\hat{\boldsymbol{C}}_{l+1})$, $\forall\ 1 \le l < W$.
2) All codewords across the switching boundaries between $\boldsymbol{A}(l_1), \boldsymbol{A}(l_2), \cdots, \boldsymbol{A}(l_\phi)$ with $1 \le l_1 < l_2 < \cdots < l_\phi \le W$ will have a distance $\min_{1\le i\le\phi} d_f(\hat{\boldsymbol{C}}_{l_i})$ at least, no matter $\boldsymbol{C}$ is successively punctured by $\boldsymbol{A}(l_1)$, $\boldsymbol{A}(l_2)$, $\cdots$, $\boldsymbol{A}(l_\phi)$ or by $\boldsymbol{A}(l_\phi)$, $\boldsymbol{A}(l_{\phi-1})$, $\cdots$, $\boldsymbol{A}(l_1)$.

We first give Theorem 1 for the case of $W = 2$ and then extend the results to general case.

Theorem 1: † *Consider an (n, k) parent code $\boldsymbol{C}$. Given the puncturing periods $(p_0, p_1, \cdots, p_{n-1})$, let $\boldsymbol{A}(1) = (a_{u,v}(1))$ and $\boldsymbol{A}(2) = (a_{u,v}(2))$ be two puncturing tables satisfying the rate-compatible criterion, i.e., if $a_{u,v}(1) = 1$, then $a_{u,v}(2) = 1$ $\forall\ u, v$. Denote by $\boldsymbol{c} = (c_{i,t}\ \forall\ i, t)$ and $\tilde{\boldsymbol{c}} = (\tilde{c}_{i,t}\ \forall\ i, t)$ two codewords of $\boldsymbol{C}$. Let $d_{\boldsymbol{A}(i)}(\boldsymbol{c}, \tilde{\boldsymbol{c}})$ be the Hamming distance between $\boldsymbol{c}$ and $\tilde{\boldsymbol{c}}$ after puncturing by $\boldsymbol{A}(i)$ for i=1,2. We have*

$$\min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \le \min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}). \tag{A-1}$$

Suppose we first puncture $\boldsymbol{C}$ by $\boldsymbol{A}(1)$ and switch the puncturing table to $\boldsymbol{A}(2)$ later. Let $d_{\boldsymbol{A}(1)|\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}})$ be the distance between $\boldsymbol{c}$ and $\tilde{\boldsymbol{c}}$ after puncturing. Then it implies that

$$\min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \le \min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(1)|\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}). \tag{A-2}$$

Moreover, even the order of puncturing is reversed, we also have

$$\min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \le \min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(2)|\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}). \tag{A-3}$$

Proof:

Let $\boldsymbol{e} = (e_{i,t}\ \forall\ i, t)$ be the difference between $\boldsymbol{c}$ and $\tilde{\boldsymbol{c}}$, where $e_{i,t} = c_{i,t} - \tilde{c}_{i,t}$ for all i and t. Suppose $\boldsymbol{C}$ is punctured by either $\boldsymbol{A}(1)$ or $\boldsymbol{A}(2)$; we have

$$d_{\boldsymbol{A}(l)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) = \sum_t \sum_i a_{i,t \bmod p_i}(l) \cdot 1(e_{i,t}),\ \forall\ l = 1, 2 \tag{A-4}$$

where $1(x)$ is defined as the function with $1(x) = 1$ if $x \neq 0$ and $1(x) = 0$ if $x = 0$. Owing to the rate-compatible restriction: $a_{u,v}(2) - a_{u,v}(1) \ge 0\ \forall\ u, v$, it implies that

$$d_{\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) - d_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) = \sum_t \sum_i (a_{i,t \bmod p_i}(2) - a_{i,t \bmod p_i}(1)) \cdot 1(e_{i,t}) \ge 0$$

for all possible $\boldsymbol{c}$ and $\tilde{\boldsymbol{c}}$, and hence

$$\min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \le \min_{\boldsymbol{c}\neq\tilde{\boldsymbol{c}}\in\boldsymbol{C}} d_{\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}).$$

Next, suppose $\boldsymbol{C}$ is first punctured by $\boldsymbol{A}(1)$ and the puncturing table is switched to $\boldsymbol{A}(2)$ at time t_0. $d_{\boldsymbol{A}(1)|\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}})$ can then be expressed as

$$\sum_{t<t_0} \sum_i a_{i,t \bmod p_i}(1) \cdot 1(e_{i,t}) + \sum_{t\ge t_0} \sum_i a_{i,t \bmod p_i}(2) \cdot 1(e_{i,t}). \tag{A-5}$$

†With a straight-forward extension, (A-1), (A-2), (A-3) can be shown to still hold no matter whether $A(1)$ and $A(2)$ are with single or irregular puncturing periods.

By (A-4), (A-5), and the rate-compatible criterion, we have

$$d^2_{\boldsymbol{A}(1)|\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) - d^2_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) = \sum_{t \geq t_0} \sum_i \left(a_{i,t \bmod p_i}(2) - a_{i,t \bmod p_i}(1)\right) \cdot 1(e_{i,t}) \geq 0$$

and thus

$$\min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \leq \min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(1)|\boldsymbol{A}(2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}).$$

Finally, if we reverse the order of puncturing, i.e., $\boldsymbol{A}(2)$ is initially employed and $\boldsymbol{A}(1)$ is adopted for puncturing after t_0, $d_{\boldsymbol{A}(2)|\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}})$ can be expressed as

$$\sum_{t<t_0} \sum_i a_{i,t \bmod p_i}(2) \cdot 1(e_{i,t}) + \sum_{t \geq t_0} \sum_i a_{i,t \bmod p_i}(1) \cdot 1(e_{i,t})$$

which infers that

$$d_{\boldsymbol{A}(2)|\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) - d_{A(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) = \sum_{t<t_0} \sum_i \left(a_{i,t \bmod p_i}(2) - a_{i,t \bmod p_i}(1)\right) \cdot 1(e_{i,t}) \geq 0$$

and therefore

$$\min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \leq \min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(2)|\boldsymbol{A}(1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}).$$

∎

By (A-1) and the definition of

$$d_{\min}(\hat{\boldsymbol{C}}_l) = \min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d^2_{\boldsymbol{A}(l)}(\boldsymbol{c}, \tilde{\boldsymbol{c}})$$

it implies that $d_f(\hat{\boldsymbol{C}}_l) \leq d_f(\hat{\boldsymbol{C}}_{l+1})$, since $\boldsymbol{A}(l)$ and $\boldsymbol{A}(l+1)$ both satisfy the rate-compatible restriction for all l. In addition, suppose $\boldsymbol{C}$ is successively punctured by $\boldsymbol{A}(l_1)$, $\boldsymbol{A}(l_2)$, $\cdots$, $\boldsymbol{A}(l_\phi)$ with the rate-compatibility and $1 \leq l_1 < l_2 < \cdots < l_\phi \leq W$. For two distinct codewords $\boldsymbol{c}$ and $\tilde{\boldsymbol{c}}$ across the boundaries between $\boldsymbol{A}(l_i)$'s, by (A-2), we have

$$\begin{aligned} & d_{\boldsymbol{A}(l_1)|\boldsymbol{A}(l_2)|\cdots|\boldsymbol{A}(l_\phi)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \\ & \geq \min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(l_1)|\boldsymbol{A}(l_2)|\cdots|\boldsymbol{A}(l_\phi)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \\ & \geq \min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(l_1)|\boldsymbol{A}(l_2)|\cdots|\boldsymbol{A}(l_{\phi-1})}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \\ & \geq \cdots \geq \min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(l_1)|\boldsymbol{A}(l_2)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \\ & \geq \min_{\boldsymbol{c} \neq \tilde{\boldsymbol{c}} \in \boldsymbol{C}} d_{\boldsymbol{A}(l_1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) = \min_{1 \leq i \leq \phi} d_f(\hat{\boldsymbol{C}}_{l_i}) \end{aligned}$$

which infers that

$$d_{\boldsymbol{A}(l_1)|\boldsymbol{A}(l_2)|\cdots|\boldsymbol{A}(l_\phi)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \geq \min_{1 \leq i \leq \phi} d_f(\hat{\boldsymbol{C}}_{l_i}).$$

Similarly, by (A-3), we can also obtain

$$d_{\boldsymbol{A}(l_\phi)|\boldsymbol{A}(l_{\phi-1})|\cdots|\boldsymbol{A}(l_1)}(\boldsymbol{c}, \tilde{\boldsymbol{c}}) \geq \min_{1 \leq i \leq \phi} d_f(\hat{\boldsymbol{C}}_{l_i})$$

thereby completing the proof.

REFERENCES

[1] J. B. Cain, G. C. Clark, and J. M. Geist,"Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 97-100, Jan. 1979.

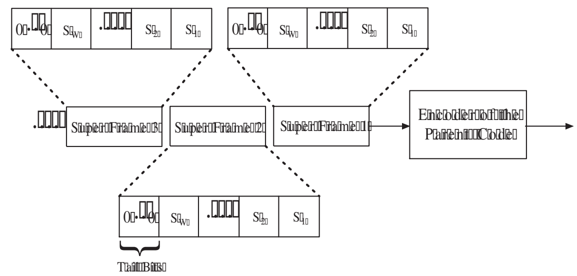
[2] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. 36, pp. 389-400, Apr. 1988.

[3] L. H. C. Lee, "New rate-compatible punctured convolutional codes for Viterbi decoding," *IEEE Trans. Commun.*, vol. 42, pp. 3073-3079. Dec. 1994.

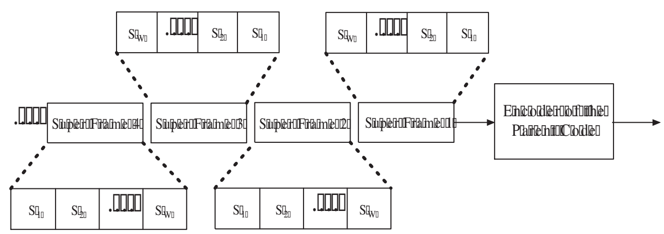
[4] D. Haccoun and G. Begin, "High-rate punctured convolutional codes for Viterbi and sequential decoding," *IEEE Trans. Commun.*, vol. 37, pp. 1113-1125, Nov. 1989.

[5] Y. Bian, A. Popplewell, and J. J. O'Reilly, "New very high rate punctured convolutional codes," *Electron. Lett.*, vol. 30, pp. 1119-1120, July 1994.

[6] R. J. McEliece, "The algebraic theory of convolutional codes," in *Handbook of Coding Theory*, V. S. Pless and W. C. Huffman eds. Amsterdam, The Netherlands: Elsevier, 1998, pp. 1065–1138.



(a) The conventional multiplexing scheme.



(b) The new proposed multiplexing scheme.

Fig. 1. Multiplexing schemes of RCPC codes for UEP.

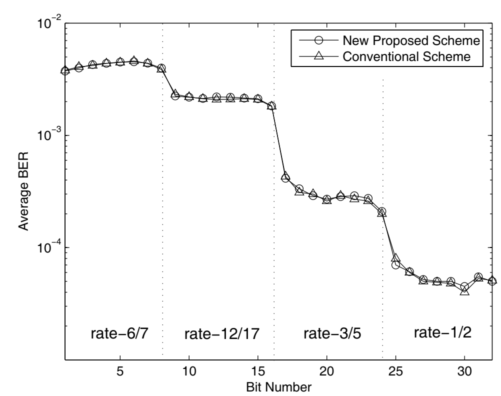


Fig. 2. Average BER of source bits in a super frame for different multiplexing schemes at signal-to-noise ratio 4.5 dB, where four groups of data S_1, S_2, S_3, S_4 (each containing 8 bits per super frame) are protected by the code family of parent code [23, 35] with $(p_0, p_1) = (3, 4)$ in Table I.

TABLE I

GOOD RATE-COMPATIBLE PUNCTURED CODES WITH IRREGULAR PUNCTURING PERIODS ($G(D)$, $d_f(\boldsymbol{C})$, r_p: GENERATOR MATRIX (IN OCTAL), FREE DISTANCE, AND CODE RATE OF THE PARENT CODE, (p_0, p_1): PUNCTURING PERIODS, $\boldsymbol{A}$: PUNCTURING TABLE, $d_f(\hat{\boldsymbol{C}})$, r_c: FREE DISTANCE AND CODE RATE OF THE CHILD CODE)

$G(D) = [5\ \ 7],\ d_f(\boldsymbol{C}) = 5,\ r_p = 1/2$			
(p_0, p_1)	$\boldsymbol{A}$	$d_f(\hat{\boldsymbol{C}})$	r_c
$(3,4)$	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 1 & 1 & 1 \end{pmatrix}$	4	$\frac{3}{5}$
	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 1 & 1 & 0 \end{pmatrix}$	3	$\frac{12}{17}$
	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 0 & 1 & 0 \end{pmatrix}$	2	$\frac{6}{7}$

(a) Parent code of code rate 1/2 and memory 2

$G(D) = [15\ \ 17],\ d_f(\boldsymbol{C}) = 6,\ r_p = 1/2$			
(p_0, p_1)	$\boldsymbol{A}$	$d_f(\hat{\boldsymbol{C}})$	r_c
$(3,4)$	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 1 & 1 & 1 \end{pmatrix}$	4	$\frac{3}{5}$
	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 1 & 1 & 0 \end{pmatrix}$	3	$\frac{12}{17}$
	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 0 & 1 & 0 \end{pmatrix}$	2	$\frac{6}{7}$
$(7,6)$	$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & \end{pmatrix}$	5	$\frac{6}{11}$
	$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & \end{pmatrix}$	4	$\frac{42}{71}$
	$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & \end{pmatrix}$	3	$\frac{42}{65}$
	$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & \end{pmatrix}$	2	$\frac{42}{59}$
	$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & \end{pmatrix}$	2	$\frac{42}{52}$

(b) Parent code of code rate 1/2 and memory 3

$G(D) = [23\ \ 35],\ d_f(\boldsymbol{C}) = 7,\ r_p = 1/2$			
(p_0, p_1)	$\boldsymbol{A}$	$d_f(\hat{\boldsymbol{C}})$	r_c
$(3,4)$	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 1 & 1 & 1 \end{pmatrix}$	5	$\frac{3}{5}$
	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 1 & 1 & 0 \end{pmatrix}$	4	$\frac{12}{17}$
	$\begin{pmatrix} 1 & 1 & 0 & \\ 1 & 0 & 1 & 0 \end{pmatrix}$	3	$\frac{6}{7}$
$(5,4)$	$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	6	$\frac{5}{9}$
	$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	5	$\frac{5}{8}$
	$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	4	$\frac{5}{7}$
	$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & \end{pmatrix}$	3	$\frac{20}{23}$
$(6,7)$	$\begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$	6	$\frac{6}{11}$
	$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$	5	$\frac{3}{5}$
	$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$	4	$\frac{21}{32}$
	$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$	3	$\frac{21}{29}$
	$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$	3	$\frac{14}{17}$
	$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$	2	$\frac{14}{15}$

(c) Parent code of code rate 1/2 and memory 4

$G(D) = [53\ \ 75],\ d_f(\boldsymbol{C}) = 8,\ r_p = 1/2$			
(p_0, p_1)	$\boldsymbol{A}$	$d_f(\hat{\boldsymbol{C}})$	r_c
$(4,5)$	$\begin{pmatrix} 1 & 1 & 1 & 0 & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$	6	$\frac{4}{7}$
	$\begin{pmatrix} 1 & 0 & 1 & 0 & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$	6	$\frac{2}{3}$
	$\begin{pmatrix} 1 & 0 & 0 & 0 & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$	4	$\frac{4}{5}$
	$\begin{pmatrix} 1 & 0 & 0 & 0 & \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}$	3	$\frac{20}{21}$
$(5,4)$	$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	7	$\frac{5}{9}$
	$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	6	$\frac{5}{8}$
	$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	5	$\frac{5}{7}$
	$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & \end{pmatrix}$	3	$\frac{20}{23}$
$(7,6)$	$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & \end{pmatrix}$	6	$\frac{6}{11}$
	$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & \end{pmatrix}$	6	$\frac{3}{5}$
	$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & \end{pmatrix}$	6	$\frac{2}{3}$
	$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & \end{pmatrix}$	4	$\frac{3}{4}$
	$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & \end{pmatrix}$	3	$\frac{14}{17}$
	$\begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & \end{pmatrix}$	2	$\frac{14}{15}$

(d) Parent code of code rate 1/2 and memory 5

$G(D) = [133\ \ 171],\ d_f(\boldsymbol{C}) = 10,\ r_p = 1/2$			
(p_0, p_1)	$\boldsymbol{A}$	$d_f(\hat{\boldsymbol{C}})$	r_c
$(5,4)$	$\begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	8	$\frac{5}{9}$
	$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	6	$\frac{5}{8}$
	$\begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & \end{pmatrix}$	6	$\frac{5}{7}$
	$\begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & \end{pmatrix}$	4	$\frac{20}{23}$
$(7,6)$	$\begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & \end{pmatrix}$	8	$\frac{7}{13}$
	$\begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & \end{pmatrix}$	7	$\frac{7}{12}$
	$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & \end{pmatrix}$	6	$\frac{7}{11}$
	$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & \end{pmatrix}$	5	$\frac{7}{10}$
	$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & \end{pmatrix}$	4	$\frac{7}{9}$
	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & \end{pmatrix}$	3	$\frac{7}{8}$

(e) Parent code of code rate 1/2 and memory 6