Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Serial decoding of rateless code over noisy channels*

Ke-di WU[1], Zhao-yang ZHANG[†‡1], Shao-lei CHEN[1], Sheng-tian YANG[2], Pei-liang QIU[1]

([1]*Institute of Information and Communication Engineering, Zhejiang University, Hangzhou 310027, China*)

([2]*Self-employed at Zhengyuan Xiaoqu 10-2-101, Hangzhou 310011, China*)

[†]E-mail: ning_ming@zju.edu.cn

Received Sept. 28, 2010; Revision accepted Dec. 9, 2010; Crosschecked July 29, 2011

Abstract: Rateless code usually generates a potentially infinite number of coded packets at the encoder and collects enough packets at the decoder to ensure reliable recovery of multiple information packets. The conventional rateless decoder usually works in a parallel manner which needs to initiate a new belief propagation (BP) decoding procedure upon each newly received collection of coded packets, thereby resulting in prohibitive decoding complexity in practice. In this paper, we present a novel serial decoding algorithm, i.e., the serial storage belief propagation (SS BP) algorithm, for rateless codes over noisy channels. Specifically, upon receiving a new group of coded packets, the decoder initiates a new attempt to decode all the packets received so far, using the results of the previous attempt as initial input. Moreover, in each iteration of the new attempt, the decoder serially propagates the messages group by group from the most recent one to the earliest one. In this way, the newly updated messages can be propagated faster, expediting the recovery of information packets. In addition, the proposed serial decoding algorithm has significantly lower complexity than the existing parallel decoding algorithms. Simulation results validate its effectiveness in AWGN, Rayleigh, and Rician fading channels.

Key words: Rateless code, Fountain codes, Serial decoding, Noisy channel
doi:10.1631/jzus.C1000340 **Document code:** A **CLC number:** TN911.22

1 Introduction

Rateless code (MacKay, 2005) is a new kind of code originally designed for binary erasure channel (BEC) to approach the channel capacity without the knowledge of channel state information (CSI) at the transmitter. Considering its fascinating property, extensive research efforts have been made on applying it to noisy channels. Raptor code (Shokrollahi, 2006), one of the most popular practical rateless codes, has proved applicable to binary memoryless symmetric channels (BiMSCs) without knowing the CSI a priori (Etesami and Shokrollahi, 2006). For its capacity-approaching potential and resistance to interferences (Palanki and Yedidia, 2004), it has been used in some communication systems, such as relay networks (Castura and Mao, 2007a; 2007b) and cognitive radio systems (Kushwaha *et al.*, 2008).

Instead of encoding the information data into a pre-determined number of packets like the fixed-rate codes, a rateless code usually generates a potentially infinite number of packets. The receiver keeps on attempting to decode the incoming packets using some appropriate channel-specific decoding algorithms, e.g., the original algorithm by Luby (2004) for erasure channel, and the complicated belief-propagation (BP) algorithms for noisy channel (Ma *et al.*, 2006). If the decoding fails, the receiver will wait until additional packets are collected and then restarts. Usually, numerous attempts of decoding are required before the information packets can be correctly recovered. Therefore, for the

‡ Corresponding author

* Project supported by the National Basic Research Program (973) of China (Nos. 2009CB320405 and 2012CB316104), the National High-Tech R & D Program (863) of China (No. 2007AA01Z257), the National Natural Science Foundation of China (No. 60972057), and the National Science & Technology Major Project of China (Nos. 2009ZX03003-004-03 and 2010ZX03003-003-01)

decoding of rateless codes over noisy channels, apart from the complexity introduced by the conventional BP algorithm itself, the numerous attempts and ever-increasing decoding length make the decoding complexity grow even more rapidly, thus becoming a critical issue in rateless code realization.

Fortunately, there are two related features in the decoding of rateless codes, which are generally different from that of conventional fixed-rate codes, such as low-density parity-check (LDPC) code (Gallager, 1963), and which could be exploited in the design of new decoding algorithms. The first is that, as mentioned above, the decoding is continuously attempted and the results of the previous decoding attempt are potentially useful for the next attempt. The second feature concerns the generally indeterministic construction of the received coded packet in rateless codes. On the one hand, usually the degree and the constituents of each coded packet are known to the receiver only when it is received. On the other hand, the newly received packets will change the code-graph structure related to all the connected information packets. Both aspects indicate that the message-passing rule for the current decoding attempt between the new incoming packets and the packets received in previous attempts should be elaborately devised.

To exploit the first feature, Hu *et al.* (2006) proposed the so-called parallel storage (PS) BP algorithm, which uses the results of the previous decoding attempt to initialize the decoder in current attempt, thus significantly expediting the decoding process and reducing its complexity without loss of performance. However, the following message-passing procedure remains the same as conventional BP, which does not take full consideration of the above features of rateless codes. Therefore, in this paper, we take both features into account and propose the so-called serial storage (SS) BP decoding algorithm for rateless codes.

In particular, the decoder collects packets in groups with the size tuned to the required rate granularity. Upon receiving a new group of coded packets, the decoder initiates a new attempt to decode all the packets received so far, using the results of the previous attempt as the initial input. Then the decoder starts to pass the latest received coded packets' messages to their relating information packets, and the updated messages are then serially propagated group by group from the most recent one to the earliest one. In this way, the newly received and updated messages can be propagated faster, and the recovery of the information packets can be further expedited.

In particular, we employ the typical Raptor code for case study, since it behaves "universal-likely" to approach the channel capacity over both BEC and noisy channels and is widely adopted in practical systems, such as the Digital Video Broadcasting (DVB) standards (Cataldi *et al.*, 2009) and 3rd Generation Partnership Project (3GPP) standards (3GPP TS26.346:2007). The analysis and the simulation results show that our proposed algorithm is better than the existing algorithms in performance and complexity.

2 System model and code graph

2.1 System model

We consider a point-to-point communication system over noisy channels (Fig. 1).

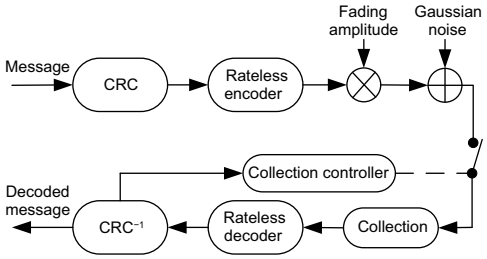

Fig. 1 The system model using rateless codes over wireless fading channel. CRC: cyclic redundancy check; $\mathbf{CRC}^{-1}$: reverse CRC

The transmitter encodes k information packets into a potentially infinite stream of coded packets, and then transmits them until the receiver perfectly decodes the information packets. The receiver begins to decode at a high code rate. If the decoding attempt fails, the receiver starts the next attempt after receiving k' more packets. If the information packets are successfully decoded via n coded packets, we refer to $R \cong k/n$ as the realized code rate. Once the receiver obtains the original information packets successfully, an 'ACK' is transmitted to inform the transmitter to start the transmission of the next information packets. We name the value $R_{\mathrm{as}} = k'/k$ the 'adjustment step', which indicates the step size of code rate.

The channel model is given by

$$y_i = h_i \cdot x_i + z_i, \tag{1}$$

where x_i, y_i, and z_i stand for the transmitted symbol, the received symbol, and the additive white Gaussian noise, respectively, h_i stands for the channel gain, and i indicates the time. The CSI is known at the receiver, but unknown at the transmitter. It has been shown that the realized rates of rateless codes on the Gaussian channel with a relatively wide range of signal-to-noise ratio (SNR) are close to the Gaussian channel capacity and that the distribution of h plays little role in the performance (Castura and Mao, 2006). Thus, the channel model reduces to the AWGN channel parameterized by the SNR. Thus, we focus our studies on the AWGN channel.

In a real system, the cyclic redundancy check (CRC) code is used to ensure the correctness of the information packets out of the rateless decoder. Without loss of generality, we assume that a packet contains only one bit, and thus we replace 'input packet' and 'coded packet' with 'input bit' and 'coded bit', respectively.

2.2 Tanner graph of rateless codes

As mentioned in Section 2.1, the designs of Luby transform (LT) codes and Raptor codes are both graph-based. In this study, we do not focus on the encoding process of rateless codes. For more details of the structure and encoding process, please refer to Luby (2004) and Shokrollahi (2006). We use the Tanner graph (Loeliger, 2004), the method usually used in graph-based coding theory, to illustrate their structures and the encoding procedures. The Tanner graph is a bipartite graph composed of two kinds of nodes, variable nodes represented by circles and check nodes represented by squares. The variable node (V-node) denotes an equal operation and the check node (C-node) denotes an exclusive-or operation.

The L-value or log-likelihood ratio (LLR) of a binary random variable $x \in \{0, 1\}$ is defined by

$$L(x) = \log\{\Pr(x = 0)/\Pr(x = 1)\}. \tag{2}$$

Assume that there are two input LLRs, denoted by $L(x)$ and $L(y)$, which are independent of each other. Then, the output LLR of V-node can be calculated by

$$L(z) = L(x) + L(y), \tag{3}$$

and the output LLR of C-node can be calculated by

$$\tanh(L(z)/2) = \tanh(L(x)/2) \cdot \tanh(L(y)/2), \tag{4}$$

where tanh denotes the hyperbolic tangent function.

Fig. 2 shows the Tanner graph of Raptor codes. It contains two parts: LDPC and LT code (LTC). When the receiver collects more coded packets according to the adjustment step, the LTC part correspondingly increases while the LDPC part remains the same. Therefore, different from the fixed-rate codes, the Tanner graph of rateless codes grows incrementally with time as new check nodes become available. Based on the Tanner graph, the BP decoding algorithm can be used to iteratively recover the input information packets.

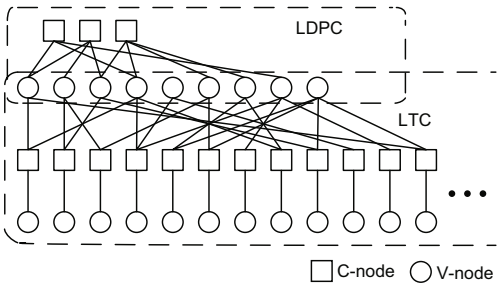

Fig. 2 The Tanner graph of Raptor codes. LDPC: low-density parity-check code; LTC: Luby transform code. C-node: check node; V-node: variable node

In this study, we regard the two parts as a whole. No matter which part the C-nodes belong to, all the C-nodes participate in the iterations of decoding. The V-nodes that belong to both of the LDPC part and LTC part also participate in the iterations of decoding. Other V-nodes, which belong only to the LTC part, do not participate in the iterations and they provide only the initial messages to the corresponding check nodes. Let matrix $\boldsymbol{A}$ denote the connection between the V-nodes and the C-nodes that belong to the LDPC part. Apparently, $\boldsymbol{A}$ is the parity check matrix of the LDPC part. Let matrix $\boldsymbol{B}$ denote the connection between the V-nodes and the C-nodes that participate in the iterations of decoding. According to the adjustment step, we divide the C-nodes and V-nodes that belong only to the LTC part into several groups, and propose the serial decoding algorithm. In the following section, we will show the decoding algorithm in detail.

3 Decoding algorithms over noisy channels

Some symbols are defined before the description. $b_{mn}=1$ denotes that there is an edge between the check node m and the variable node n; $b_{mn}=0$ denotes that there is no edge between the two nodes, $b_{mn}\in\boldsymbol{B}$. The set $N(m)=\{n:b_{mn}=1\}$ denotes the variable nodes that are connecting to the mth check node and the set $M(n)=\{m:b_{mn}=1\}$ denotes the check nodes that are connecting to the nth variable node. Let $\varepsilon_{mn}^{(i)}$ denote the LLR message from check node m to variable node n in the ith iteration, and $g_{nm}^{(i)}$ the LLR message from variable node n to check node m in the ith iteration. $g_n^{(i)}$ denotes the sum of the input messages of variable node n in the ith iteration.

Let $\bar{\boldsymbol{s}}=(s_0,s_1,...,s_{K-1})$ denote the K information packets, $\bar{\boldsymbol{w}}=(w_0,w_1,...,w_{N-1})$ the N coded packets encoded by a high rate LDPC code, and $\bar{\boldsymbol{e}}=(e_0,e_1,...,e_i,...)$ the endless coded packets encoded by the LTC encoder. $s_i,w_i,e_i\in\{0,1\}$. By binary phase shift keying (BPSK) modulation, the transmitted packets are denoted by $\bar{\boldsymbol{x}}$, where $x_i=1-2e_i$. The noise $\bar{\boldsymbol{z}}$ is a Gaussian random variable with mean zero and variance σ^2. Therefore, the received packets can be denoted by $\bar{\boldsymbol{y}}=\bar{\boldsymbol{x}}+\bar{\boldsymbol{z}}$, where $\bar{\boldsymbol{y}}=(y_0,y_1,...,y_i,...)$.

The receiver tries to decode the information packets at a relatively high rate which is called the 'starting code rate'. When the decoding fails, the receiver would collect additional packets and try to decode the information packets again. Setting the starting code rate as R_0, the adjustment step as R_{as}, and the maximum number of iterations as $I_{\max}$, the different decoding algorithms are listed as follows.

3.1 Standard parallel BP decoding algorithm

This algorithm (Loeliger, 2004) has been widely adopted in both LDPC codes and rateless codes. Here, we list it in detail as it is the basis of the following algorithms. We will show only the difference when we depict other algorithms.

1. Beginning: Keep on receiving packets until the received code rate is smaller than R_0.

2. Initialization: The input message from the channel to the mth variable node of the LTC part is $F_m=4y_m/\sigma^2$, where $m=0,1,....$ All the other LLR messages are set to be zero.

3. Updating the check nodes: For check node m, $0\le m\le M-1$, where M is the sum of the check nodes of LTC and LDPC parts, the updating rule in the ith iteration is

$$\varepsilon_{mn}^{(i)}=\operatorname{sign}(F_m)\prod_{n'\in N(m),n'\neq n}\alpha_{n'm}^{(i-1)}\cdot\psi^{-1}\Big(\psi(|F_m|)+\sum_{n'\in N(m),n'\neq n}\psi(\beta_{n'm}^{(i-1)})\Big),\tag{5}$$

where $\alpha_{nm}^{(i-1)}=\operatorname{sign}(g_{nm}^{(i-1)})$, $\beta_{nm}^{(i-1)}=|g_{nm}^{(i-1)}|$, $n\in N(m)$ and

$$\psi(x)=-\log\tanh(x/2),\ \operatorname{sign}(x)=\begin{cases}1, & x\ge 0,\\ -1, & x<0.\end{cases}$$

4. Updating the variable nodes: For variable node n, $0\le n\le N-1$, the updating rule in the ith iteration is

$$g_{nm}^{(i)}=\sum_{m'\in M(n),m'\neq m}\varepsilon_{m'n}^{(i)},\tag{6}$$

$$g_n^{(i)}=\sum_{m\in M(n)}\varepsilon_{mn}^{(i)},\tag{7}$$

where $m\in M(n)$.

5. Hard decision:

$$\hat{w}_n^{(i)}=\begin{cases}0, & g_n^{(i)}\ge 0,\\ 1, & g_n^{(i)}<0.\end{cases}$$

If the parity check equation $\boldsymbol{A}\hat{\boldsymbol{w}}^{(i)}=\boldsymbol{0}$ is not satisfied, where $\boldsymbol{A}$ is the parity check matrix of the outer LDPC code, $i=i+1$: if i is smaller than $I_{\max}$, go to step 3; otherwise, go to step 6.

6. The condition of ending iteration: By decoding a very simple LDPC code, the decoded packets can be obtained from the vector $\hat{\boldsymbol{w}}^{(i)}$. The output in iteration i is denoted by $\hat{\boldsymbol{s}}^{(i)}$. If $\hat{\boldsymbol{s}}^{(i)}=\bar{\boldsymbol{s}}$, which means the initial information packets are successfully decoded, go to step 7. Otherwise, $i=i+1$. If $i\le I_{\max}$, go to step 3. When the iteration index $i>I_{\max}$, the decoding process at the current code rate will be ended.

In practice, whether the decoded information packets are correct or not can be determined by CRC. In our simulation, we ignore this and just compare $\bar{\boldsymbol{s}}^{(i)}$ with $\hat{\boldsymbol{s}}$ directly for simplicity.

7. The condition of ending transmission: If the information packets are successfully decoded, end the whole decoding process and inform the transmitter that the transmission is success. Otherwise, receive $K\cdot R_{\mathrm{as}}$ more coded packets to achieve the next lower code rate and go to step 2.

3.2 Parallel storage BP decoding algorithm

The parallel storage (PS) BP decoding algorithm was proposed by Hu *et al.* (2006). The only difference between this algorithm and the standard parallel BP algorithm is the initialization step. After receiving some more coded packets, the Tanner graph extends only to a larger one, and the original part does not change. Thus, the result of the last decoding attempt still contains useful information. Instead of setting all LLR messages to zero, the initialization step is:

Initialization: The input message from the channel to the mth check node of the LTC part is $F_m = 4y_m/\sigma^2$. All the other LLR messages from the last decoding attempt keep unchanged.

The other steps are the same as those of the standard BP algorithm. Therefore, the computation complexity remains the same in one iteration.

3.3 Serial storage BP decoding algorithm

Noticing another feature of the rateless decoder that the receiver serially receives the coded packets, we propose to use a serial decoding algorithm to decode the information packets and improve the convergence speed. We divide the received coded packets into several groups according to the adjustment step R_{as}, which means each group contains $K \cdot R_{\mathrm{as}}$ nodes. To facilitate the expression, let $N_T = K \cdot R_{\mathrm{as}}$. Denoting the number of divided groups by T, the number of received coded packets by $T \cdot N_T$, the way of dividing the packets is described as follows. The firstly received N_T packets are put into group 0, the next received N_T packets into group 1, and so on. The T groups are numbered consecutively from 0 to $T-1$. Therefore, the latest N_T received packets are put into group $T-1$.

When the receiver has received enough coded packets and the received code rate has reached R_0, the first attempt of decoding begins. At this time, $T = 1/(R_0 \cdot R_{\mathrm{as}})$. If this decoding attempt fails, the decoder will collect N_T more packets to start a new decoding attempt, and the number of groups is increased by one. All the LLR messages of the last decoding attempt, including the V-nodes' input messages from C-nodes, the V-nodes' output messages to C-nodes, and the sums of the input messages of V-nodes, are stored as the initial values in the new decoding attempt.

Each decoding attempt contains several decoding iterations. The maximum number of iterations is limited by $I_{\max}$. In each decoding iteration, the LLR messages are updated serially group by group. At first, the messages related with group $T-1$, including the V-nodes' output messages to C-nodes and the V-nodes' input messages from C-nodes, are updated by using the stored messages from the previous decoding attempt and the LLR messages from the channel. Then, the messages related with group $T-2$ are updated, and so on. After the updating of messages related with group 0, the messages of the LDPC part are updated. Then, update all the sums of the variable nodes' input messages. At this time, the decoder performs the step of hard decision, and all the remaining steps, which are the same as in the PS BP algorithm.

To better describe the updating order in each iteration when performing the SS BP algorithm, we draw Fig. 3 to show the process of message passing. Take group t as an example. When updating the messages related with group t (Fig. 3a), all the input messages to C-nodes from the relating V-nodes are first updated. All the output messages to the relating V-nodes are updated (Fig. 3b). Then the messages related with group $t-1$ are updated (Figs. 3c and 3d). Note that the sets of relating V-nodes of different groups are different, and that the messages to be updated of different groups are also different. No message passing along the edge between a V-node and a C-node is updated twice in one iteration. After the updating of group 0 (Fig. 3e), the messages from the V-nodes to the C-nodes of the LDPC part are updated. Then the messages from the C-nodes of the LDPC part to the V-nodes are updated (Fig. 3f). The decoder can then calculate the sums of the variable nodes' input messages, which is the end of the process of message passing during one iteration.

As mentioned above, the difference between SS BP and PS BP algorithms lies in the process of decoding iteration, including the step of updating the input messages from V-nodes to C-nodes of the LTC part and the step of updating the V-nodes. We list the SS BP in detail as follows:

1. Beginning: Set $T = (R_0 \cdot R_{\mathrm{as}})^{-1} - 1$. Keep on receiving packets until the received code rate is smaller than R_0.

2. Initialization: Set $T = T + 1$. Number the T groups consecutively. The input message from the

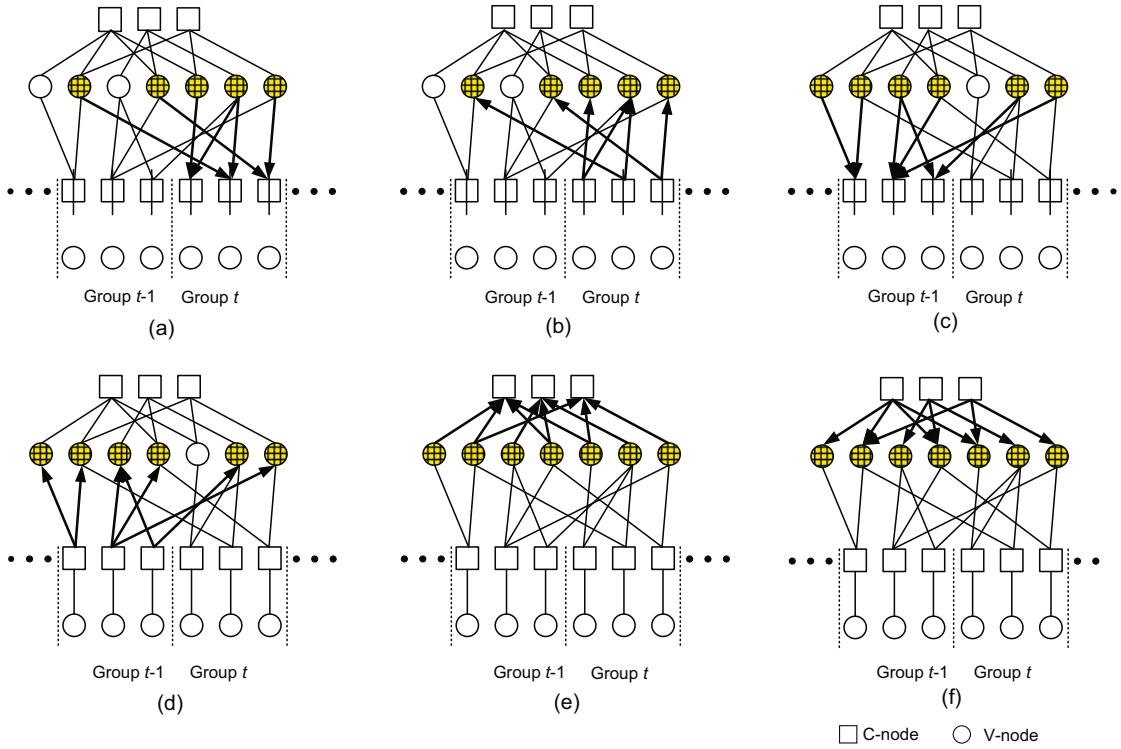

Fig. 3 The graphs of message passing when performing the serial storage belief propagation (SS BP) algorithm. (a) For group t, all input messages to C-nodes from the relating V-nodes are updated; (b) For group t, all output messages to the relating V-nodes are updated; (c) For group $t-1$, all input messages to C-nodes from the relating V-nodes are updated; (d) For group $t-1$, all output messages to the relating V-nodes are updated; (e) After group 0 is updated, the messages from the V-nodes to the C-nodes of the LDPC part are updated; (f) The messages from the C-nodes of the LDPC part to V-nodes are updated. The colored circles with crosshatch denote the relating V-nodes and the bold lines with arrows denote the messages to be updated. C-node: check node; V-node: variable node

channel to the mth check node of the LTC part is $F_m = 4y_m/\sigma^2$. All the other LLR messages from the last decoding attempt keep unchanged.

3. Updating the messages of the LTC part:

For $(t = T-1; t \geq 0; t--)$, where t indexes the groups

For $(m = tN_T; m < (t+1)N_T; m++)$

{For every $n \in N(m)$, update the input messages from V-nodes to C-nodes in group t according to Eq. (8):

$$g_{nm}^{(i)} = \sum_{\substack{m' \in M(n), m' \neq m \\ m' \leq tN_T}} \varepsilon_{m'n}^{(i-1)} + \sum_{\substack{m' \in M(n) \\ m' > tN_T}} \varepsilon_{m'n}^{(i)}; \quad (8)$$

update the output messages of C-node m according to Eq. (5); }

4. Updating the messages of the LDPC part:

Update the input messages from V-nodes to C-nodes of the LDPC part according to Eq. (6);

Update the output messages from C-nodes of the LDPC part to V-nodes according to Eq. (5).

5. Updating the sum of the input messages of V-nodes according to Eq. (7).

From the step of 'hard decision', the remaining steps are the same as in the PS BP algorithm. The following section will show the effectiveness of serial decoding compared with parallel decoding.

4 Effectiveness of serial decoding and complexity analysis

4.1 Effectiveness of serial decoding

In this subsection, we show the reason why serial decoding is more efficient than parallel decoding. We use Gaussian approximation density evolution (DE) theory (Chung *et al.*, 2001; Richardson and Urbanke,

2001; Richardson *et al.*, 2001) to analyze the rate of convergence during the iterations, and show the superiority of serial decoding.

According to Gaussian approximation theory, every LLR message passing along the edge between a V-node and a C-node can be regarded as a Gaussian distributed variable whose variance is twice as large as its mean. Therefore, we need only to study the evolution of the mean during the iterations. According to the theory of DE, the error probability of decoding is no concern of the transmitted codes when the symmetric condition is satisfied. Therefore, assuming that we transmit an 'all one' code word, the decoding has no error if the mean of the message passing from the C-node to the V-node tends to infinity as the number of iterations tends to infinity. The rate of convergence of the mean denotes the effectiveness of decoding.

Here we concentrate on the analysis of decoding of the LTC part of Raptor codes, because serial decoding and parallel decoding perform the same when decoding the LDPC part. The LTC part can be described by its output degree distribution of the check nodes. The degree of a check node is defined as the number of edges connecting with the variable nodes. Let $\Omega_1, \Omega_2, ..., \Omega_{d_{\max}}$ be the distribution of degrees $1, 2, ..., d_{\max}$ so that Ω_d denotes the probability of choosing the value d under this distribution. The polynomial $\Omega(x) = \sum_{d_c=1}^{d_{\max}} \Omega_{d_c} \cdot x^{d_c}$, which is associated with the corresponding edge degree distribution $\omega(x) = \sum_{d_c=1}^{d_{\max}} \omega_i \cdot x^{d_c-1} = \Omega'(x)/\Omega'(1)$, denotes the structure of the coded check nodes. $\Omega'(x)$ denotes the derivative operation on function $\Omega(x)$. The input variable nodes are chosen uniformly at random, so that their node degree distribution is binomial and can be approximated by a Poisson distribution with parameter α. Both the node and edge degree distributions of the input variable nodes can be expressed by the polynomial $I(x) = \mathrm{e}^{\alpha(x-1)}$. Both distributions are of mean α.

First, we analyze the parallel decoding algorithm. As mentioned in Section 3, assuming all the input messages of V-node $\varepsilon_{mn}^{(i)}$ are independent, the rule of updating the output message of the V-node in the parallel decoding algorithm is

$$g_{nm}^{(i)} = \sum_{\substack{m' \in M(n), \\ m' \neq m}} \varepsilon_{m'n}^{(i)}. \tag{9}$$

Let $m_{\mathrm{c}}^{(i)}$ denote the mean value of the LLR message from the C-node to the V-node in the ith iteration, $m_{\mathrm{v}}^{(i)}$ the mean value of the LLR message from the V-node to the C-node. From the Gaussian approximation theory, ε_{mn} obeys the Gaussian distribution $N(m_{\mathrm{c}}, 2m_{\mathrm{c}})$, and g_{nm} obeys the Gaussian distribution $N(m_{\mathrm{v}}, 2m_{\mathrm{v}})$. The rate of convergence of $m_{\mathrm{c}}^{(i)}$ denotes the effectiveness of decoding. From Eq. (9), the rule of density evolution on the V-node with degree d_{v} is

$$m_{\mathrm{v},d_{\mathrm{v}}}^{(i)} = (d_{\mathrm{v}} - 1) m_{\mathrm{c}}^{(i)}, \tag{10}$$

where $d_{\mathrm{v}} = |M(n)|$. Because the edge degree distribution of the variable nodes can be expressed by the polynomial $I(x) = \mathrm{e}^{\alpha(x-1)}$ and the mean of the degree is α, the rule of the density evolution on the V-node of the LTC part is

$$m_{\mathrm{v}}^{(i)} = (\alpha - 1) m_{\mathrm{c}}^{(i)}. \tag{11}$$

Also, denote the degree number of the check node by d_{c}, $d_{\mathrm{c}} = |N(m)|$. Assuming the input messages $g_{nm}^{(i-1)}$ are independent, from Eq. (5), the rule of updating the output message of the C-node can be rewritten as

$$\tanh\left(\frac{\varepsilon_{mn}^{(i)}}{2}\right) = \tanh\left(\frac{F_n}{2}\right) \prod_{\substack{n' \in N(m), \\ n' \neq n}} \tanh\left(\frac{g_{n'm}^{(i-1)}}{2}\right). \tag{12}$$

Therefore,

$$E\left(\tanh\left(\frac{\varepsilon_{mn}^{(i)}}{2}\right)\right) = E\left(\tanh\left(\frac{F_n}{2}\right)\right) \cdot \prod_{i=1}^{d_{\mathrm{c}}-1} E\left(\tanh\left(\frac{g_{nm}^{(i-1)}}{2}\right)\right). \tag{13}$$

By the theory of Gaussian approximation,

$$E\left(\tanh\left(\frac{g_{nm}^{(i-1)}}{2}\right)\right) = \frac{1}{\sqrt{4\pi m_{\mathrm{v}}^{(i-1)}}} \cdot \int_R \tanh(u/2) \exp\left(-\frac{(u - m_{\mathrm{v}}^{(i-1)})^2}{4m_{\mathrm{v}}^{(i-1)}}\right) \mathrm{d}u. \tag{14}$$

Define $\phi(x)$ as

$$\phi(x) = \begin{cases} 1 - \dfrac{1}{\sqrt{4\pi x}} \displaystyle\int_R \tanh(u/2) \mathrm{e}^{-\frac{(u-x)^2}{4x}} \mathrm{d}u, & x > 0, \\ 1, & x = 0, \end{cases}$$

where $\phi(0) = 1$, $\phi(\infty) = 0$, and function $\phi(x)$ is continuous, convex, and monotonically decreasing.

Therefore, Eq. (14) can be rewritten as

$$m_{\mathrm{c}}^{(i)}=\phi^{-1}\left\{1-[1-\phi(m_0)][1-\phi(m_{\mathrm{v}}^{(i-1)})]^{d_{\mathrm{c}}-1}\right\}, \quad (15)$$

where m_0 denotes the mean value of the initial message F_n. Eq. (15) is the rule of density evolution on the C-node with degree d_{c}. At the beginning of the iteration, $i=1$, $m_{\mathrm{v}}^{(0)}=0$ and $m_{\mathrm{c}}^{(1)}=0$ when $d_{\mathrm{c}}\geq 2$. Only if $d_{\mathrm{c}}=1$, can the decoding iteration get start and $m_{\mathrm{c}}^{(1)}=m_0$. Therefore, the check nodes with degree one are used to initialize the process of decoding. Denoting the check nodes' edge degree distribution of the LTC part by $\omega(x)=\sum_{d_{\mathrm{c}}=1}^{d_{\max}}\omega_{d_{\mathrm{c}}}\cdot x^{d_{\mathrm{c}}-1}$, the rule of density evolution on the C-node of the LTC part is

$$m_{\mathrm{c}}^{(i)}=\sum_{d_{\mathrm{c}}=1}^{d_{\max}}\omega_{d_{\mathrm{c}}}\varphi^{-1}\{1-[1-\varphi(m_0)][1-\varphi(m_{\mathrm{v}}^{(i-1)})]^{d_{\mathrm{c}}-1}\}. \quad (16)$$

Then, associating with Eq. (11), we obtain the whole Gaussian approximation DE of the LTC part when performing parallel decoding:

$$\begin{aligned}m_{\mathrm{c}}^{(i)}=&\sum_{d_{\mathrm{c}}=1}^{d_{\max}}\omega_{d_{\mathrm{c}}}\varphi^{-1}\{1-[1-\varphi(m_0)]\\&\cdot[1-\varphi((\alpha-1)m_{\mathrm{c}}^{(i-1)})]^{d_{\mathrm{c}}-1}\}.\end{aligned} \quad (17)$$

Second, we analyze the serial decoding algorithm. When dividing the check nodes of the LTC part into T groups, each group contains N_T check nodes. The variable t indexes the groups, $0\leq t\leq T-1$. For $tN_T\leq m\leq(t+1)N_T-1$, to every $n\in N(m)$, the rule of updating the output message of the V-node when performing the serial decoding algorithm is

$$g_{nm}^{(i)}=\sum_{\substack{m'\in M(n),m'\neq m\\ m'\leq tN_T}}\varepsilon_{m'n}^{(i-1)}+\sum_{\substack{m'\in M(n)\\ m'>tN_T}}\varepsilon_{m'n}^{(i)}. \quad (18)$$

Therefore, the rule of density evolution on the variable node is

$$\begin{aligned}m_{\mathrm{v},t}^{(i)}=&\sum_{\substack{m'\in M(n),m'\neq m\\ m'\leq tN_T}}m_{\mathrm{c}}^{(i-1)}+\sum_{\substack{m'\in M(n)\\ m'>tN_T}}m_{\mathrm{c}}^{(i)}\\=&\frac{t+1}{T}(\alpha-1)m_{\mathrm{c}}^{(i-1)}+\frac{T-1-t}{T}(\alpha-1)m_{\mathrm{c}}^{(i)},\end{aligned} \quad (19)$$

$$m_{\mathrm{v}}^{(i)}=\frac{1}{T}\sum_{t=0}^{T-1}m_{\mathrm{v},t}^{(i)}. \quad (20)$$

Because the rule of updating the check node is the same as that in parallel decoding, we obtain the whole Gaussian approximation DE of the LTC part when performing the serial decoding algorithm:

$$\begin{aligned}m_{\mathrm{c},t}^{(i)}=&\sum_{d_{\mathrm{c}}=1}^{d_{\max}}\omega_{d_{\mathrm{c}}}\cdot\phi^{-1}\{1-[1-\phi(m_0)]\\&\cdot[1-\phi(\frac{t+1}{T}(\alpha-1)m_{\mathrm{c},t+1}^{(i-1)}\\&+\frac{T-1-t}{T}(\alpha-1)m_{\mathrm{c},t+1}^{(i)})]^{d_{\mathrm{c}}-1}\},\end{aligned} \quad (21)$$

$$m_{\mathrm{c}}^{(i)}=\frac{1}{T}\sum_{t=0}^{T-1}m_{\mathrm{c},t}^{(i)}. \quad (22)$$

Define function $f(s)$ as follows:

$$f(s)=\sum_{d_{\mathrm{c}}=1}^{d_{\max}}\omega_{d_{\mathrm{c}}}\cdot\phi^{-1}\left\{1-[1-\phi(m_0)][1-\phi(s)]^{d_{\mathrm{c}}-1}\right\}. \quad (23)$$

When transmitting an 'all one' code word, m_0 is greater than zero, $1-\phi(m_0)>0$. Because function $\phi(x)$ is continuous and monotonically decreasing, function $f(s)$ is continuous and monotonically increasing. Obviously, $m_{\mathrm{c},T-1}^{(i)}<m_{\mathrm{c},T-2}^{(i)}<\cdots<m_{\mathrm{c},1}^{(i)}<m_{\mathrm{c},0}^{(i)}$. Compared with Eqs. (17), (21), and (22), the $m_{\mathrm{c}}^{(i)}$ of parallel decoding equals only the $m_{\mathrm{c},T-1}^{(i)}$ of serial decoding. Therefore, the value of $m_{\mathrm{c}}^{(i)}$ when performing serial decoding is greater than the value when performing parallel decoding. This means $m_{\mathrm{c}}^{(i)}$ increases and tends to infinity faster when performing serial decoding. This is the reason why serial decoding is more efficient than parallel decoding.

We also verify the DE analysis results of the Raptor code for serial decoding and parallel decoding by simulation. Fig. 4 shows the improvement of the serial decoding algorithm compared with the parallel decoding algorithm. Here, the inner LT code is generated using the distribution

$$\begin{aligned}\mu(x)=&0.007969x+0.493570x^2+0.166220x^3\\&+0.072646x^4+0.082558x^5+0.056058x^8\\&+0.037229x^9+0.055590x^{19}+0.025023x^{65}\\&+0.003135x^{66},\end{aligned} \quad (24)$$

which has been widely used (Palanki and Yedidia, 2004; Shokrollahi, 2006). The rate of the outer

LDPC codes is 0.95. The SNR of the binary input (BI) AWGN channel is -2.83 dB. The mean of the output LLR message of the V-node varies as the number of iterations increases. According to the mean, we can calculate the error probability of decoding. Obviously, according to the result, the serial decoding algorithm needs fewer iterations than the parallel decoding algorithm when the error probability tends to zero. Also, we notice that the Raptor code can be successfully decoded at rates of both 2.3^{-1} and 2.4^{-1}, but the numbers of iterations needed are different.

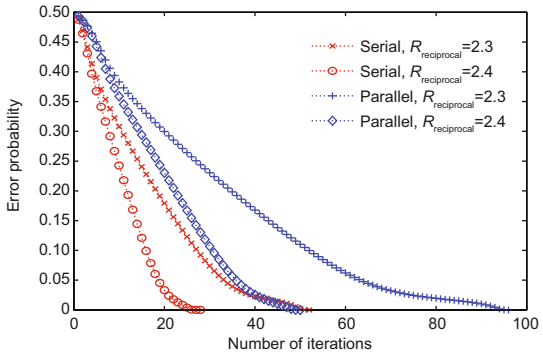

Fig. 4 The relationship between the number of iterations and the error probability of the serial decoding algorithm and the parallel decoding algorithm

At last, note that the DE method is based on the code length of infinity and the cycle-free Tanner graph. It is not possible to realize the conditions in real systems. In practical systems, the decoder may need more iterations to recover the information packets, or need more received packets to achieve successful decoding, which means the realized code rate needs to be slower.

4.2 Complexity comparison

In this subsection, we compare the complexity of the PS BP algorithm and the SS BP algorithm. The complexities of updating both the C-node and the V-node should be studied. Specifically, the complexity of updating the C-node is more important. From the description of the decoding algorithms in Section 3, the rules of updating the check nodes are the same when performing PS BP and SS BP algorithms, and all the output messages of C-nodes are updated once in each iteration. Therefore, in terms of updating the check nodes in one iteration, the complexities of the two algorithms are the same.

According to the updating rule of the V-node, we notice that the main operation is adding the LLR messages from the corresponding C-nodes. We define the addition as '$+\varepsilon$' operation. Set the degree of a variable node as d_{v}. When performing the PS BP algorithm, according to Eq. (6), it takes $(d_{\mathrm{v}}-1)$ '$+\varepsilon$' operations to update one output LLR message of the V-node. When performing the SS BP algorithm, according to Eq. (8), it also takes $(d_{\mathrm{v}}-1)$ '$+\varepsilon$' operations to update one output LLR message of the V-node in the step of 'updating the input messages of the C-node in group t', and only the messages along the edges connecting to the C-nodes in group t are updated. When the group index t varies from $T-1$ to 0 in each iteration, each LLR message from the V-node to the C-node is updated only once. Therefore, the complexities of updating the LLR message from the V-node to the C-node are the same when performing the two algorithms. As the rules of updating the sum of the input messages of V-nodes are the same, the complexities of updating the variable nodes in each iteration are also the same.

As the other steps act the same, the complexities of the two algorithms are the same. Therefore, the number of decoding iterations decides the whole complexity of the decoding algorithm. The smaller the number of iterations, the better the complexity performance.

Of course, in practice, by utilizing the sum value of all the input LLR messages of the V-node, the updating of the V-node can be implemented in a simple way. When performing the PS BP algorithm, the sum value of all the input LLR messages of the V-node can be calculated by

$$Q_n^{(i)}=\sum_{\substack{m'\in M(n),\\ |M(n)|=d_{\mathrm{v}}}}\varepsilon_{m'n}^{(i)}. \tag{25}$$

Subtracting the corresponding input LLR message, each output LLR message to the C-node can be obtained by

$$g_{nm}^{(i)}=Q_n^{(i)}-\varepsilon_{mn}^{(i)}. \tag{26}$$

We name it '$-\varepsilon$' operation. Therefore, by utilizing the sum value $Q_n^{(i)}$, the PS BP algorithm needs only to do d_{v} '$+\varepsilon$' operations and d_{v} '$-\varepsilon$' operations in one iteration when updating the V-nodes.

When performing the SS BP algorithm, we can also update the output messages of V-nodes by using the sum value. Assume that the degree of the V-node is d_{v}, $d_{\mathrm{v}}=|M(n)|$, and that the d_{v} C-nodes

connecting to the V-node belong to different groups. The sum value of all the input LLR messages of the V-node can be calculated by

$$Q_n^{(i)} = \sum_{\substack{m' \in M(n) \\ m' \le tN_T}} \varepsilon_{m'n}^{(i-1)} + \sum_{\substack{m' \in M(n) \\ m' > tN_T}} \varepsilon_{m'n}^{(i)}. \quad (27)$$

From Eq. (27) and Eq. (8), the message from V-node n to C-node m in group t can be calculated by

$$g_{nm}^{(i)} = Q_n^{(i)} - \varepsilon_{mn}^{(i-1)}. \quad (28)$$

Obviously, only one '$-\varepsilon$' operation is needed to update one message from the V-node to the C-node. As the degree of the V-node is d_v, d_v '$-\varepsilon$' operations are needed to update all the input messages to C-nodes.

Moreover, according to Eq. (27), we notice that $Q_n^{(i)}$ will be updated when t varies. Therefore, the operations of updating the sum value $Q_n^{(i)}$ should be done. The updating of $Q_n^{(i)}$ is caused by the updating of the message from the C-node to the V-node, which is denoted by $\varepsilon_{mn}^{(i)}$. Thus, the updating rule is

$$Q_{n(\mathrm{new})}^{(i)} = Q_{n(\mathrm{old})}^{(i)} - \varepsilon_{mn}^{(i-1)} + \varepsilon_{mn}^{(i)}, \quad (29)$$

where $Q_{n(\mathrm{new})}^{(i)}$ denotes the newly updated value of $Q_n^{(i)}$ and $Q_{n(\mathrm{old})}^{(i)}$ denotes the original value of $Q_n^{(i)}$. From Eq. (28), we notice that the '$-\varepsilon$' operation has been done when updating the input message to C-nodes. Therefore, only one '$+\varepsilon$' operation is needed to obtain the newly updated value of $Q_n^{(i)}$. As the degree of the V-node is d_v, $Q_n^{(i)}$ will be updated d_v times in one iteration. Therefore, d_v '$+\varepsilon$' operations are needed when updating the sum value $Q_n^{(i)}$ in one iteration.

In summary, by using the sum value, the SS BP algorithm needs to do d_v '$+\varepsilon$' operations and d_v '$-\varepsilon$' operations in one iteration when updating the V-node, which is the same as in the PS BP algorithm. Therefore, the number of decoding iterations also decides the whole complexity of the decoding algorithm. Obviously, serial decoding is better, as seen from the above analysis.

5 Simulation results

We use the Monte Carlo simulation method to show the performance of our proposed serial storage decoding algorithm. We do the simulations over different kinds of noisy channels including the AWGN channel, Rayleigh fading channel, and Rice fading channel.

5.1 Performances over the AWGN channel

As an example, we study the scenario when the SNR of the BIAWGN channel is -2.83 dB, which means the channel's capacity is 0.5. The Raptor code has 9500 information bits and uses an outer LDPC code of rate 0.95 to obtain 10 000 coded bits. Here we use the progressive edge growth (PEG) algorithm (Hu and Eleftheriou, 2005) to generate the LDPC part to maximize the girth length. These bits are then encoded by an inner LT code encoder whose degree distribution is as given in Eq. (24).

Fig. 5 shows the decoding performances of the three algorithms we have mentioned in Section 3. The adjustment step is 0.1. We limit the maximum number of decoding iterations to observe the bit error rates (BERs) at different code rates. At code rate 2.1^{-1}, all of the BERs are higher than 10^{-2} and the receivers need to receive some more coded packets. At code rate 2.3^{-1}, the BER is lower than 10^{-4} when performing SS BP with 40 iterations, SS BP with 20 iterations, PS BP with 40 iterations, and standard parallel BP with 150 iterations. But the BER is much higher than 10^{-3} when performing PS BP with 20 iterations and standard parallel BP with both 20 and 40 iterations.

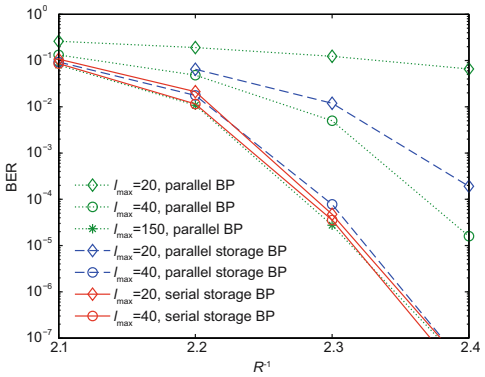

Fig. 5 Bit error rates for different decoding algorithms with different limits of iteration numbers over the AWGN channel

The performances of SS BP with both 20 and 40 iterations, PS BP with 40 iterations, and standard parallel BP with 150 iterations are nearly the same. All of them successfully decode the information bits at rate 2.4^{-1}. Obviously, the SS BP algorithm performs best according to a smaller number of iterations and lower BER. As analyzed in the above section, the differences of complexities among these

algorithms depend on their numbers of iterations. Therefore, the complexity of the SS BP algorithm is also the smallest when achieving the same BER performance.

We further investigate the contrast between SS BP and PS BP algorithms. We set the decoder to start decoding at rate 2.3^{-1}, and store the messages passing along the edges at different numbers of iterations. Then, the stored messages are used to help the decoding at rate 2.4^{-1}. Fig. 6 shows the different average numbers of iterations between SS BP and PS BP that are needed to reduce the BER to below 10^{-7}. Obviously, SS BP decoding needs fewer iterations than PS BP decoding.

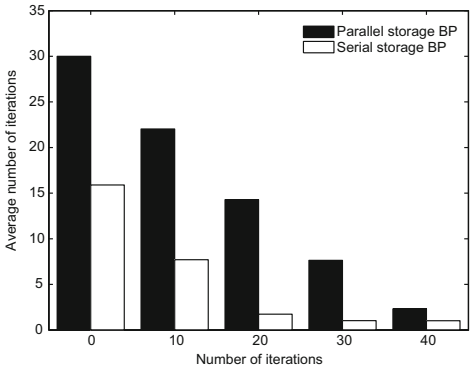

Fig. 6 The average iteration numbers for two decoding algorithms with different storage limits. The x axis denotes the number of iterations at rate 2.3^{-1}, and the y axis denotes the average number of iterations needed to descend the BER to below 10^{-7}. Iteration 0 means there is no storage at rate 2.3^{-1}

5.2 Performances over other noisy channels

In this subsection, we do simulations over the Rayleigh fading channel and the Rician fading channel, which are two widely used wireless channel models. Simulations show that the SS BP decoding algorithm is also efficient.

First, we study the scenario of the Rayleigh fading channel. The fading coefficient h is in submission to Rayleigh distribution:

$$f(h)=\begin{cases}\dfrac{h}{a^2}\exp\left(-\dfrac{h^2}{2a^2}\right), & h\geq 0,\\ 0, & h<0,\end{cases}$$

where $a^2=1$. The variance of the Gaussian noise is $\sigma^2=1$. We use the Raptor code which is the same as the one used in the simulation over the AWGN channel. Fig. 7 shows the decoding performances of the three algorithms. The adjustment step is 0.1. Obviously, the SS BP algorithm performs the best by using fewer iterations and achieving lower BER.

Second, we study the scenario of the Rician fading channel. The fading coefficient h is in submission to Rice distribution:

$$f(h)=\begin{cases}\dfrac{h}{a^2}\exp\left(-\dfrac{s^2+h^2}{2a^2}\right)\cdot I_0\left(\dfrac{s\cdot h}{a^2}\right), & h\geq 0,\\ 0, & h<0,\end{cases}$$

where $a^2=1$, $s^2=0.5$, and the function $I_0(x)$ is the first-class zero-order modified Bessel function. Also, the variance of the Gaussian noise is $\sigma^2=1$. Fig. 8 shows the decoding performances of the three algorithms. Obviously, the SS BP algorithm is also more efficient than other algorithms.

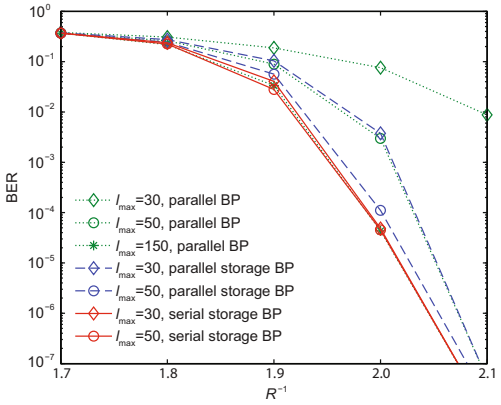

Fig. 7 Bit error rates for different decoding algorithms with different limits of iteration numbers over the Rayleigh fading channel

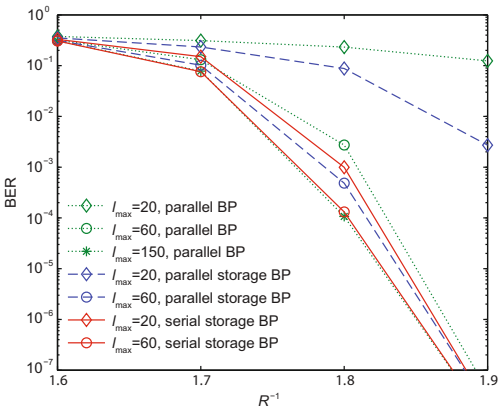

Fig. 8 Bit error rates for different decoding algorithms with different limits of iteration numbers over the Rician fading channel

6 Conclusions

In this paper, we propose a serial storage BP decoding algorithm of rateless codes over noisy channels. The proposed algorithm is designed based on the feature of the rateless decoder. Compared with the existing decoding methods, our proposed algorithm has significant advantages in computational complexity without loss of performance. Simulation results demonstrate that the SS BP algorithm is well suited for the decoding over not only the AWGN channels, but also the Rayleigh and Rician fading channels. The reduction of the rateless codes' decoding complexity will make rateless codes more practical and competitive over noisy channels.

References

3GPP TS26.346:2007. Technical Specification Group Services and System Aspects, Multimedia Broadcast/Multicast Services (MBMS) Protocols and Codes (Release 6).

Castura, J., Mao, Y., 2006. Rateless coding over fading channels. *IEEE Commun. Lett.*, **10**(1):46-48. [doi:10.1109/LCOMM.2006.1576565]

Castura, J., Mao, Y., 2007a. Rateless coding and relay networks. *IEEE Signal Process. Mag.*, **24**(5):27-35. [doi:10.1109/MSP.2007.904814]

Castura, J., Mao, Y., 2007b. Rateless coding for wireless relay channels. *IEEE Trans. Wirel. Commun.*, **6**(5):1638-1642. [doi:10.1109/TWC.2007.360364]

Cataldi, P., Gerla, M., Zampognaro, F., 2009. Rateless Codes for File Transfer over DVB-S. First Int. Conf. on SPACOMM, p.7-12. [doi:10.1109/SPACOMM.2009.20]

Chung, S.Y., Richardson, T.J., Urbanke, R.L., 2001. Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation. *IEEE Trans. Inf. Theory*, **47**(2):657-670. [doi:10.1109/18.910580]

Etesami, O., Shokrollahi, A., 2006. Raptor codes on binary memoryless symmetric channels. *IEEE Trans. Inf. Theory*, **52**(5):2033-2051. [doi:10.1109/TIT.2006.872855]

Gallager, R.G., 1963. Low-Density Parity-Check Codes. MIT Press, Cambridge, MA.

Hu, K., Castura, J., Mao, Y., 2006. WLC44-3: Reduced-Complexity Decoding of Raptor Codes over Fading Channels. IEEE Global Telecommunications Conf., p.1-5. [doi:10.1109/GLOCOM.2006.873]

Hu, X., Eleftheriou, E., 2005. Regular and irregular progressive edge growth Tanner graphs. *IEEE Trans. Inf. Theory*, **51**(1):386-398. [doi:10.1109/TIT.2004.839541]

Kushwaha, H., Xing, Y., Chandramouli, R., 2008. Reliable multimedia transmission over cognitive radio networks using fountain codes. *Proc. IEEE*, **96**(1):155-165. [doi:10.1109/JPROC.2007.909917]

Loeliger, H.A., 2004. An introduction to factor graphs. *IEEE Signal Process. Mag.*, **21**(1):28-41. [doi:10.1109/MSP.2004.1267047]

Luby, M., 2004. LT Codes. 43rd Annual IEEE Symp. on Foundations of Computer Science, p.271-280. [doi:10.1109/SFCS.2002.1181950]

Ma, Y., Yuan, D., Zhang, H., 2006. Fountain Codes and Applications to Reliable Wireless Broadcast System. IEEE Information Theory Workshop, p.66-70. [doi:10.1109/ITW2.2006.323758]

MacKay, D.J.C., 2005. Fountain codes. *IEE Proc. Commun.*, **152**(6):1062-1068. [doi:10.1049/ip-com:20050237]

Palanki, R., Yedidia, J.S., 2004. Rateless Codes on Noisy Channels. IEEE Int. Symp. on Information Theory, p.37. [doi:10.1109/ISIT.2004.1365075]

Richardson, T.J., Urbanke, R.L., 2001. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Trans. Inf. Theory*, **47**(2):599-618. [doi:10.1109/18.910577]

Richardson, T.J., Shokrollahi, M.A., Urbanke, R.L., 2001. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Trans. Inf. Theory*, **47**(2):619-637. [doi:10.1109/18.910578]

Shokrollahi, A., 2006. Raptor codes. *IEEE Trans. Inf. Theory*, **52**(6):2551-2567. [doi:10.1109/TIT.2006.874390]