Successive Cancellation Permutation Decoding of Reed-Solomon Codes

Peter Trifonov
Saint-Petersburg State Polytechnic University
Email: petert@dcn.icc.spbstu.ru

***Abstract*—A novel soft-decision decoding algorithm for Reed-Solomon codes over $\mathbb{F}_{2^m}$ is considered. The proposed approach is based on their representation as Arikan polar codes with dynamic frozen symbols and application of the sequential successive cancellation decoding algorithm. Furthermore, permutation techniques are utilized in order to reduce the decoding complexity.**

I. Introduction

Efficient soft-decision decoding of Reed-Solomon codes is a long-standing open problem. Numerous techniques were suggested for solving it. Major recent milestones include Koetter-Vardy algebraic soft decision method [1], adaptive belief propagation method [2], and a hybrid approach [3]. However, neither of these techniques can provide near-ML decoding.

In this paper a novel approach towards solving this problem is suggested. The proposed method exploits representation of Reed-Solomon codes as polar codes [4] with dynamic frozen symbols [5]. This enables application of efficient sequential soft-decision decoding techniques developed in the area of polar coding [6]. Furthermore, permutation decoding techniques are utilized in order to reduce the average number of iterations performed by the sequential decoder. This result is similar to the one presented in [7], where permutation techniques were used to significantly reduce the complexity of a list decoding algorithm for Reed-Muller codes.

The paper is organized as follows. Section II introduces the necessary background on Reed-Solomon and polar codes. The proposed decoding algorithm is described in Section III. Numeric results illustrating the performance and complexity of this approach are provided in Section IV.

II. Background

A. Reed-Solomon codes

$(n, k, n-k+1)$ Reed-Solomon code over $\mathbb{F}_{2^m}$ is defined as a set of vectors $(f(X_0), \ldots, f(X_{n-1}))$, where $f(x) = \sum_{i=0}^{k-1} f_i x^i, f_i \in \mathbb{F}_q$, are message polynomials, and X_i are distinct elements of $\mathbb{F}_{2^m}$. In this paper it will be assumed that $n = q = 2^m$, and X_i are arranged in the standard bit order in some basis of $\mathbb{F}_{2^m}$. However, multiple bases will be considered simultaneously, which effectively induce different permutations of X_i.

B. Polar codes

$(n = 2^m, k)$ polar code over $\mathbb{F}_2$ is defined as a linear block code generated by some k rows of matrix $G = B_m \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}^{\otimes m}$, where B_m is the bit-reversal permutation matrix, and $A^{\otimes m}$ denotes m-times Kronecker product of matrix A with itself. That is, the encoding operation is given by $c_0^{n-1} = u_0^{n-1} G$, where $u_i = 0, i \in \mathcal{F}$, $a_i^j = (a_i, \ldots, a_j)$, $\mathcal{F} \subset \{0, \ldots, n-1\}$ is the set of $n-k$ (static) frozen symbols, and u_i are input symbols of the linear transformation given by matrix G (polarizing transformation). It is possible to show that the linear transformation given by G gives rise to n synthetic bit subchannels with transition probabilities $p(y_0^{n-1}, u_0^{i-1} | u_i)$, where y_0^{n-1} is the received noisy sequence. It is possible to show that in the case of binary input output symmetric memoryless channel the capacities of these subchannels converge to 0 and 1 [4], i.e. the channels polarize. However, channel polarization is not the subject of this paper.

Reed-Muller codes represent a special case of polar codes. Recall, that a Reed-Muller code of length 2^m and order r is a set of vectors obtained by evaluating Zhegalkin polynomials of degree at most r in all distinct points of $\mathbb{F}_2^m$. It can be seen that the Reed-Muller code of order r can be also obtained as a polar code by setting $\mathcal{F} = \{i | \operatorname{wt}(i) < m - r\}$, where $\operatorname{wt}(i)$ is the number of 1's in the binary expansion of integer i.

It is possible to generalize the construction of polar codes by setting frozen symbols not statically to zero, but to some linear combinations of other symbols (dynamic frozen symbols) [5]. Given any linear code of length $n = 2^m$ with check matrix H, one can represent it as a polar code with dynamic frozen symbols, where the dynamic freezing constraints are given by

$$u_0^{n-1} V^T = 0,$$

where $V = QHG^T$ is a matrix, such that Q is a non-singular $(n-k) \times (n-k)$ matrix, and at most one row of matrix ends[1] in each column of this matrix. Gaussian elimination can be used to obtain such matrix V from HG^T. Then the values of dynamic frozen symbols are given by

$$u_{j_i} = \sum_{s=0}^{j_i - 1} u_s V_{i,s}, 0 \leq i < n-k, \tag{1}$$

where row i of matrix V ends in column j_i, and $\mathcal{F} = \{j_0, \ldots, j_{n-k-1}\}$. Indeed, any vector $c_0^{n-1} = u_0^{n-1} G$, where

[1]Row i ends in column j_i iff $V_{i,j_i} = -1$, and $V_{i,s} = 0, s > j_i$.

u_0^{n-1} satisfies (1), satisfies $c_0^{n-1}H^T = 0$, i.e. is a codeword of the considered code.

C. Sequential decoding of polar codes

Successive cancellation (SC) algorithm is a standard method for decoding polar codes. However, its performance is very far from that of a maximum-likelihood decoder, since the SC algorithm cannot recover from errors which may occur at its early phases. This problem can be avoided by employing list or stack decoding techniques [8], [9].

Codewords of a polar code can be arranged into a tree. Leaves of this tree correspond to codewords $c_0^{n-1} = u_0^{n-1}G$, and intermediate nodes are identified by paths u_0^{i-1}, such that $u_0^{n-1}V^T = 0$. Stack decoding algorithm for binary polar codes performs depth-first search within code tree for a path u_0^{n-1} with the highest probability $p(u_0^{n-1}|y_0^{n-1})$, where y_0^{n-1} is the received vector. That is, paths u_0^i are stored in a stack (priority queue), and at each iteration path u_0^{i-1} with the highest score is extracted for extension. For $i \in \mathcal{F}$ the path is extended to obtain $(u_0, \dots, u_{i-1}, a)$, where a is the value of frozen symbol u_i (see (1)), and for $i \notin \mathcal{F}$ path extension results in two paths $(u_0, \dots, u_{i-1}, 0), (u_0, \dots, u_{i-1}, 1)$. Extended paths are stored in the stack together with their scores. In what follows, i will be referred to as phase number.

The original stack decoding algorithm for polar codes employs $p(u_0^i|y_0^{n-1})$ as score of path u_0^i [9]. It was shown in [6] that the average decoding complexity can be significantly reduced at the cost of negligible performance loss if one defines path score as

$$T(u_0^i, y_0^{n-1}) = R_2(u_0^i, y_0^{n-1})\Omega_2(i), \tag{2}$$

where

$$R_2(u_0^i, y_0^{n-1}) = \max_{u_{i+1}^{n-1} \in \mathbb{F}_2^{n-i-1}} p(u_0^{n-1}|y_0^{n-1}), \tag{3}$$

and

$$\Omega_2(i) = \prod_{j \in \mathcal{F}, j > i} (1 - P_j) \tag{4}$$

is an estimate of the probability that sequence u_0^{n-1}, which achieves maximum in (3), satisfies freezing constraints, i.e. corresponds to a valid codeword. Here P_j is the error probability in the j-th subchannel of the polarizing transformation, provided that symbols u_0^{j-1} are known. In the case of binary polar codes P_j can be computed efficiently using density evolution [10] or Gaussian approximation [11].

It is possible to show that $R_2(u_0^i, y_0^{n-1})$ can be recursively computed as

$$R_2(u_0^{2i}, y_0^{n-1}) = \max_{u_{2i+1} \in \mathbb{F}_2} R_2\left(u_{0,e}^{2i+1} \oplus u_{0,o}^{2i+1}, y_0^{n/2-1}\right) \cdot R_2\left(u_{0,o}^{2i+1}, y_{n/2}^{n-1}\right), \tag{5}$$

$$R_2(u_0^{2i+1}, y_0^{n-1}) = R_2\left(u_{0,e}^{2i+1} \oplus u_{0,o}^{2i+1}, y_0^{n/2-1}\right) \cdot R_2\left(u_{0,o}^{2i+1}, y_{n/2}^{n-1}\right), \tag{6}$$

where $R_2(b, y_j) = p(b|y_j), b \in \mathbb{F}_2$ is the probability of transmission of symbol b over the original channel, given received noisy value y_j.

In order to keep the size of the stack limited, paths with low scores can be purged from the stack. Furthermore, if the decoder returns to phase i more than L times, all paths shorter than $i+1$ are eliminated. Decoding terminates as soon a path of length n appears at the top of the stack, or the stack becomes empty. Hence, the worst case complexity of stack decoding is given by $O(Ln \log n)$.

III. Successive cancellation decoding of Reed-Solomon codes

A. Reed-Solomon codes as polar codes with dynamic frozen symbols

Given a $(2^m, k, 2^m - k + 1)$ Reed-Solomon code, one can represent it as a polar code with dynamic frozen symbols as described in Section II-B. Let us investigate in more details the structure of the freezing constraints.

Let $X = \sum_{i=0}^{m-1} x_i a_i, x_i \in \mathbb{F}_2$, be an expansion of $X \in \mathbb{F}_{2^m}$ in some basis $\mathbf{a} = (a_0, \dots, a_{m-1}), a_i \in \mathbb{F}_{2^m}$, and $j = \sum_{s=0}^{m-1} j_s 2^s$ be a binary expansion of integer j. Then $X^j = \prod_{s=0}^{m-1} \left(\sum_{i=0}^{m-1} x_i a_i^{2^s}\right)^{j_s}$. Expanding this expression, one obtains some polynomial in variables $x_0, \dots, x_{m-1}$ of degree $\mathrm{wt}(j) = \sum_{s=0}^{m-1} j_s$, which can be represented as $x^j = \sum_{t \in Q_j} w_{jt}^{(\mathbf{a})} \prod_{i=0}^{m-1} x_i^{t_i}$, where Q_j is the set of multidegrees $t = (t_0, \dots, t_{m-1})$ of non-zero terms, and $w_{jt}^{(\mathbf{a})} \in \mathbb{F}_{2^m}$ are some coefficients. Hence, one obtains

$$f(x) = \sum_{j=0}^{k-1} f_j \sum_{t \in Q_j} w_{jt}^{(\mathbf{a})} \prod_{i=0}^{m-1} x_i^{t_i}. \tag{7}$$

This expression represents a generalized Zhegalkin polynomials $f : \mathbb{F}_2^m \to \mathbb{F}_{2^m}$. Hence, any codeword of the Reed-Solomon code can be obtained as

$$c_0^{n-1} = f_0^{k-1} W^{(\mathbf{a})} G, \tag{8}$$

where $W^{(\mathbf{a})} = (w_{jt}^{(\mathbf{a})})$, i.e. $W^{(\mathbf{a})}G$ is the generator matrix of the Reed-Solomon code. Observe that the t-th row of G is a sequence of values of monomials $\prod_{i=0}^{m-1} x_i^{1-t_i}$ in various points $(x_0, \dots, x_{m-1}) \in \mathbb{F}_2^m$, where $t = \sum_{i=0}^{m-1} t_{m-1-i} 2^i$. It can be seen that the dynamic freezing constraints for Reed-Solomon code are given by matrix $V = V^{(\mathbf{a})}$, such that $W^{(\mathbf{a})}(V^{(\mathbf{a})})^T = 0$.

It can be seen that different bases $\mathbf{a}$ correspond to different permutations of codeword symbols. However, for any basis $\mathbf{a}$ the subchannels of polarizing transformation with indices

$$i : \mathrm{wt}(i) < \max_{0 \le j < k} (m - \mathrm{wt}(j))$$

are frozen. Furthermore, it is possible to show that the set of (dynamic) frozen symbols does not depend on basis $\mathbf{a}$.

Example 1. Consider $(8, 4, 4)$ Reed-Solomon code over $\mathbb{F}_{2^3}$. Its check matrix is given by

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & \alpha & 1+\alpha & \alpha^2 & 1+\alpha^2 & \alpha+\alpha^2 & 1+\alpha+\alpha^2 \\ 0 & 1 & \alpha^2 & 1+\alpha^2 & \alpha+\alpha^2 & 1+\alpha+\alpha^2 & \alpha & 1+\alpha \\ 0 & 1 & 1+\alpha & \alpha^2 & 1+\alpha^2 & \alpha+\alpha^2 & 1+\alpha+\alpha^2 & \alpha \end{pmatrix},$$

where α is a primitive root of x^3+x+1. By applying elementary operations to rows of $H(B_3F^{\otimes 3})^T$, one obtains

$$V=\begin{pmatrix}0 & \alpha & 0 & 1+\alpha & 0 & \alpha+\alpha^2 & 1 & 0\\ 0 & 1+\alpha & 0 & 0 & 1 & 0 & 0 & 0\\ 0 & \alpha+\alpha^2 & 1 & 0 & 0 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{pmatrix}$$

Hence, u_0 is a statically frozen symbol, and u_2, u_4, u_6 are dynamic frozen symbols.

B. Sequential successive cancellation decoding

The SC decoding algorithm and its variations can be easily extended to the case of Arikan polar codes (with dynamic frozen symbols) over $\mathbb{F}_{2^m}$ [12]. Let us consider the case of sequential decoding algorithm [6], discussed in Section II-B. The idea of this approach remains valid in the case of codes over $\mathbb{F}_{2^m}$. Namely, (2) becomes

$$T(u_0^i, y_0^{n-1}) = R_{2^m}(u_0^i, y_0^{n-1})\Omega_{2^m}(i), \tag{9}$$

where $R_{2^m}(u_0^i, y_0^{n-1})$ are computed in the same way as $R_2(u_0^i, y_0^{n-1})$, except that maximization in (5) is performed over $\mathbb{F}_{2^m}$. In the case of transmission of a binary image of the code over a memoryless channel, i.e. if $p(c_j|y_j) = \prod_{s=0}^{m-1} p(c_j[s]|y_j[s])$, these expressions can be further simplified to

$$R_{2^m}(u_0^i, y_0^{n-1}) = \prod_{s=0}^{m-1} R_2\left(u_0^i[s], y_0^{n-1}[s]\right), \tag{10}$$

where $a_i^j[s] = (a_i[s], \dots, a_j[s])$, $u_i = \sum_{s=0}^{m-1} u_i[s]a_s$, $y_j = (y_j[0], \dots, y_j[m-1])$, and $y_j[s]$ is the received noisy symbol corresponding to the s-th bit of c_j. Function $\Omega_{2^m}(i)$ is defined similarly to the case of a binary code, i.e.

$$\Omega_{2^m}(i) = \prod_{j\in\mathcal{F}, j>i} (1-P'_j), \tag{11}$$

where P'_j is the error probability in the j-th subchannel of the polarizing transformation in the case of a memoryless output-symmetric channel with 2^m-ary input, provided that symbols u_0^{j-1} are known to the decoder. Error probabilities P'_j can be evaluated using gaussian approximation techniques [6]. In the case of transmission of the binary image of the code over a memoryless channel one obtains $1-P'_j = (1-P_j)^m$, where P_j is the error probability in the j-th subchannel of the polarizing transformation in the case of binary-input channel, which can be computed using density evolution [10] or Gaussian approximation [11].

C. Permutation decoding

Observe that given some noisy vector y_0^{n-1}, one can apply to it the above described sequential decoding algorithm with dynamic constraint matrices $V^{(\mathbf{a})}$ obtained using arbitrary basis $\mathbf{a}$. If list size L and stack capacity Θ are finite and sufficiently small, the decoder may fail (i.e. kill the correct path u_0^{n-1} and produce a very low probability codeword) for one basis, and produce a correct solution of the decoding problem for another basis[2]. Hence, one can try to perform decoding a number of times using different bases, and select the codeword with the maximum probability. However, this approach results in rather inefficient implementation, since most of these decoding attempts produce identical results, i.e. a decoding failure or the ML codeword.

Therefore, we propose to perform cooperative decoding using a number of different permutations (bases). Let $\phi_0 < n$ be a parameter called decision boundary. Let $W'(\mathbf{a})$ be submatrix of $W(\mathbf{a})$ consisting of first ϕ_0 columns. Let $\widehat{W}'(\mathbf{a})$ and $\widehat{W}(\mathbf{a})$ be submatrices of $W'(\mathbf{a})$ and $W(\mathbf{a})$, respectively, consisting of columns with indices in $\{0, \dots, n-1\} \setminus \mathcal{F}$. Let $\widehat{u}(\mathbf{a}) = f_0^{k-1}\widehat{W}(\mathbf{a})$ and $\widehat{u}'(\mathbf{a}) = f_0^{k-1}\widehat{W}'(\mathbf{a})$, so that $\widehat{u}(\mathbf{a}) = (\widehat{u}'(\mathbf{a})|\widehat{u}''(\mathbf{a}))$ for some vector $\widehat{u}''(\mathbf{a})$. Let $\mathbf{a}^{(0)}, \dots, \mathbf{a}^{(t-1)}$ be some distinct bases of $\mathbb{F}_{2^m}$. Let

$$Z = \left(\widehat{W}'(\mathbf{a}^{(1)})| \dots |\widehat{W}'(\mathbf{a}^{(t-1)})|\widehat{W}(\mathbf{a}^{(0)})\right)$$

be a $k \times (k + (t-1)\phi_1)$ matrix, where ϕ_1 is the column dimension of $\widehat{W}'(\mathbf{a}^{(s)}), 1 \le s < t$. Gaussian elimination can be used to construct a cross-check matrix $Y : ZY^T = 0$, such that $Y^T = (\Phi^T|\Psi^T)$, matrix Φ has all its non-zero elements in columns $0, 1, \dots, t\phi_1 - 1$, while rows of $\rho \times (\phi_1(t-1)+k)$ matrix Ψ end in distinct positions $h_i \ge t\phi_1$, and $Q_{i,h_i} = 1$, where ρ is some non-negative integer. . Let $\mathcal{H} = \{h_0, \dots, h_{\rho-1}\}$. Then any valid combination of vectors $\widehat{u}(\mathbf{a}^{(i)})$ satisfies

$$(\widehat{u}'(\mathbf{a}^{(1)})| \dots |\widehat{u}'(\mathbf{a}^{(t-1)})|\widehat{u}(\mathbf{a}^{(0)}))\Phi^T = 0 \tag{12}$$

$$(\widehat{u}'(\mathbf{a}^{(1)})| \dots |\widehat{u}'(\mathbf{a}^{(t-1)})|\widehat{u}(\mathbf{a}^{(0)}))\Psi^T = 0 \tag{13}$$

Equation (12) can be used to identify combinations of paths $\widehat{u}'(\mathbf{a}^{(i)}) = u_0^{\phi_0-1}$, which can possibly correspond to the same codeword. Having identified such combinations, one can use (13) to predict the values of $u_i, i \ge \phi_0$, for path $\widehat{u}(\mathbf{a}^{(0)})$, i.e. freeze some symbols which are not originally frozen. This reduces the probability of the sequential decoder diverging from the correct path. The value of ρ, i.e. the number of elements of $u''(\mathbf{a}^{(0)})$ which can be predicted by combining paths $u'(\mathbf{a}^{(s)}), 0 \le s < t$, depends on ϕ_0.

Example 2. Consider $(16, 8, 9)$ Reed-Solomon code over $\mathbb{F}_{2^4}$. Let α be a primitive root of x^4+x+1. Let $\mathbf{a}^{(0)} = (1, \alpha, \alpha^2, \alpha^3)$, and $\mathbf{a}^{(1)} = (\alpha^3, \alpha^2, \alpha, 1)$. The first basis corresponds to ordering $X_0 = 0, X_1 = 1, X_2 = \alpha, X_3 = \alpha+1, \dots$. The second basis corresponds to ordering $X_0 = 0, X_1 = \alpha^3, X_2 = \alpha^2, X_3 = \alpha^3+\alpha^2, \dots$. In both cases symbols $u_{2i}, 0 \le i < 8$, are frozen.

It can be verified that

$$\widehat{W}(\mathbf{a}^{(0)}) = \begin{pmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 1 & 0 & \alpha & \alpha^2 & \alpha^{12}\\ 0 & 0 & 0 & 1 & 0 & \alpha^2 & \alpha^4 & \alpha^9\\ 0 & \alpha^5 & \alpha^{10} & \alpha^2 & \alpha^8 & \alpha^5 & 0 & \alpha^6\\ 0 & 0 & 0 & 1 & 0 & \alpha^4 & \alpha^8 & \alpha^3\\ 0 & 1 & 1 & \alpha^5 & \alpha^5 & \alpha^{10} & \alpha^{10} & 1\\ 0 & \alpha^{10} & \alpha^5 & \alpha^4 & \alpha & \alpha^{10} & 0 & \alpha^{12}\\ 1 & \alpha^4 & \alpha^8 & \alpha^{11} & \alpha^{11} & \alpha^3 & \alpha^{14} & \alpha^9\end{pmatrix}$$

[2]Observe that each basis defines a permutation of y_0^{n-1}.

$$\widehat{W}(\mathbf{a}^{(1)}) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \alpha^3 & 0 & \alpha^2 & \alpha & \alpha^{12} \\ 0 & 0 & 0 & \alpha^6 & 0 & \alpha^4 & \alpha^2 & \alpha^9 \\ 0 & \alpha^{11} & \alpha^{13} & \alpha^{13} & \alpha^8 & 0 & \alpha^5 & \alpha^6 \\ 0 & 0 & 0 & \alpha^{12} & 0 & \alpha^8 & \alpha^4 & \alpha^3 \\ 0 & \alpha^{10} & \alpha^5 & \alpha^{10} & \alpha^5 & \alpha^{10} & \alpha^{10} & 1 \\ 0 & \alpha^7 & \alpha^{11} & \alpha^{11} & \alpha & 0 & \alpha^{10} & \alpha^{12} \\ \alpha^7 & \alpha^4 & 1 & \alpha^{13} & \alpha^{11} & \alpha^{14} & \alpha^3 & \alpha^9 \end{pmatrix}$$

Let $\phi_0 = 4$. Then one obtains

$$\Phi = \begin{array}{c} \begin{matrix} 1 & 3 & 5 & 7 & 1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 \end{matrix} \\ \left(\begin{array}{cccc||cccc|cccc} \alpha^8 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \alpha^3 & \alpha^8 & \alpha & 0 & 0 & \alpha^8 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right) \end{array},$$

and

$$\Psi = \begin{array}{c} \begin{matrix} 1 & 3 & 5 & 7 & 1 & 3 & 5 & 7 & 9 & 11 & 13 & 15 \end{matrix} \\ \left(\begin{array}{cccc||cccc|cccc} \alpha^6 & 0 & \alpha^7 & 0 & 0 & \alpha^{14} & 0 & 0 & 1 & 0 & 0 & 0 \\ \alpha^6 & \alpha^6 & \alpha^8 & \alpha^7 & 0 & \alpha^{10} & 0 & \alpha^{14} & 0 & \alpha^8 & 1 & 0 \end{array}\right) \end{array}.$$

Here column indices denote the corresponding input symbols u_i of the polarizing transformation.

That is, having correctly identified symbols u_1, u_3, u_5, u_7 for received sequence permutations given by bases $\mathbf{a}^{(0)}$ and $\mathbf{a}^{(1)}$ (observe that the first permutation is the identity one, and the second permutation is the bit reversal permutation), one can predict symbols u_9 and u_{13} corresponding to $\mathbf{a}^{(0)}$.

The ability to combine paths $u'(\mathbf{a}^{(s)})$ to obtain a longer one enables the following generalization of the above presented sequential decoding algorithm. Here the stack stores quadruples (T, u_0^i, s, U), where T is the score of path u_0^i, s identifies the corresponding basis $\mathbf{a}^{(s)}$, and U is either $\emptyset$, or a vector of dimension ρ, which is used to predict values of non-frozen symbols. Path scores are given by (9). Let Λ_s be the set of paths $u'(\mathbf{a}^{(s)})$ of length ϕ_0 obtained by the decoder for basis $\mathbf{a}^{(s)}$.

1) Permute the received sequence y_0^{n-1} according to bases $\mathbf{a}^{(s)}, 0 \le s < t$. Put into the stack triples $(1, \epsilon, s, \emptyset), 0 \le s < t$. Let $\Lambda_s = \emptyset$. Let $\psi_0^{n-1} = 0$.
2) Extract from the stack triple (T, u_0^{i-1}, s, U) with the largest score T. Let $\psi_i = \psi_i + 1$.
3) If $i = \phi_0 \wedge U = \emptyset$, then $\Lambda^{(s)} \leftarrow \Lambda^{(s)} \cup \{(u_0^{i-1}, T)\}$. For all tuples $[(u^{(j_1)}, T_{j_1}), \ldots, (u^{(j_{t-1})}, T_{j_{t-1}}), (u^{(j_0)}, T_{j_0})] \in \Lambda^{(1)} \times \cdots \times \Lambda^{(t-1)} \times \Lambda^{(0)}$, such that $u^{(j_s)} = u_0^{i-1}$ and $\mu\tilde{\Phi}^T = 0$, compute $U = \mu\tilde{\Psi}^T$. Here $\tilde{A}$ denotes submarix of A consisting of first $\phi_1 t$ columns, $\mu = (\tau(u^{(j_1)}), \ldots, \tau(u^{(j_{t-1})}), \tau(u^{(j_0)}))$, and $\tau(u_0^{i-1})$ is a subvector of u_0^{i-1} given by indices not in $\mathcal{F}$. Put into the stack quadruples $(\min_s T_{j_s}, u^{(j_0)}, 0, U)$.
4) If $i \in \mathcal{F}$, compute u_i from (1), and put (T', u_0^i, s, U) into the stack, where T' is the score of u_0^i.
5) If $i \notin \mathcal{F} \wedge (U = \emptyset \vee i \notin \mathcal{H})$, put into the stack paths $(T', u_0^i, s, U), u_i \in \mathbb{F}_{2^m}$, where T' are the corresponding scores.
6) If $i \notin \mathcal{F} \wedge U \neq \emptyset \wedge i \in \mathcal{H}$, then compute

$$u_i = U_p + \sum_{\substack{j=\phi_0 \\ j \notin \mathcal{F}}}^{i-1} u_j \Psi_{p,\sigma(j)},$$

where $h_p = i$, and $\sigma(j)$ is the index[3] of column in Q corresponding to i. Put (T', u_0^i, s, U) into the stack, where T' is the score of u_0^i.
7) If $\psi_i \ge L$, then remove from the stack all entries corresponding to paths shorter than i.

Observe that for $t = 1$ this algorithm reduces to the sequential decoding algorithm discussed in Section III-B.

It can be seen that this algorithm executes steps 4–6 at most Ln times. Therefore, its worst-case complexity is given by $O(Ln \log n)$ unit operations, where unit operation is given by (5)–(6), where maximization is performed over $\mathbb{F}_{2^m}$ instead of $\mathbb{F}_2$. In the case of transmission of the binary image of the code unit operation can be simplified as discussed in Section III-B.

IV. Performance of the decoding algorithm

Figure 1 presents simulation results illustrating the performance of the proposed decoding algorithm for the case of transmission of the binary image of $(32, 15, 18)$ Reed-Solomon code over AWGN channel. For comparison, performance of the Koetter-Vardy algebraic soft-decision decoding method is also shown. For the proposed method, the decision boundary was set to $\phi_0 = 10$, and bases $(\alpha^i, 0 \le i < 5)$ and $(\alpha^{4-i}, 0 \le i < 5)$ were used. This permitted prediction of $\rho = 2$ non-frozen symbols u_i. Figure 2 presents the average number of iterations performed by the proposed decoding algorithm.

It can be seen that even without permutation techniques, the proposed sequential decoding algorithm does outperform Koetter-Vardy method. However, for considered values of L, error floor appears at high SNR in the case of $t = 1$. Setting $t = 2$ provides significant performance improvement and eliminates the error floor. At the same time this results in significant reduction of the number of iterations performed by the decoder.

Observe that increasing L enables one to improve the performance of the decoding algorithm. Increasing t also results in better decoder performance.

V. Conclusions

In this paper a novel decoding method for Reed-Solomon codes was presented. It exploits sequential successive cancellation decoding techniques together with permutation decoding to obtain significant performance improvement compared to Koetter-Vardy approach. Simulation results indicate that permutation techniques enable not only performance improvement, but also reduction of the average decoding complexity.

[3] In the context of Example 1, $\sigma(9) = 8, \sigma(11) = 9, \sigma(13) = 10$.

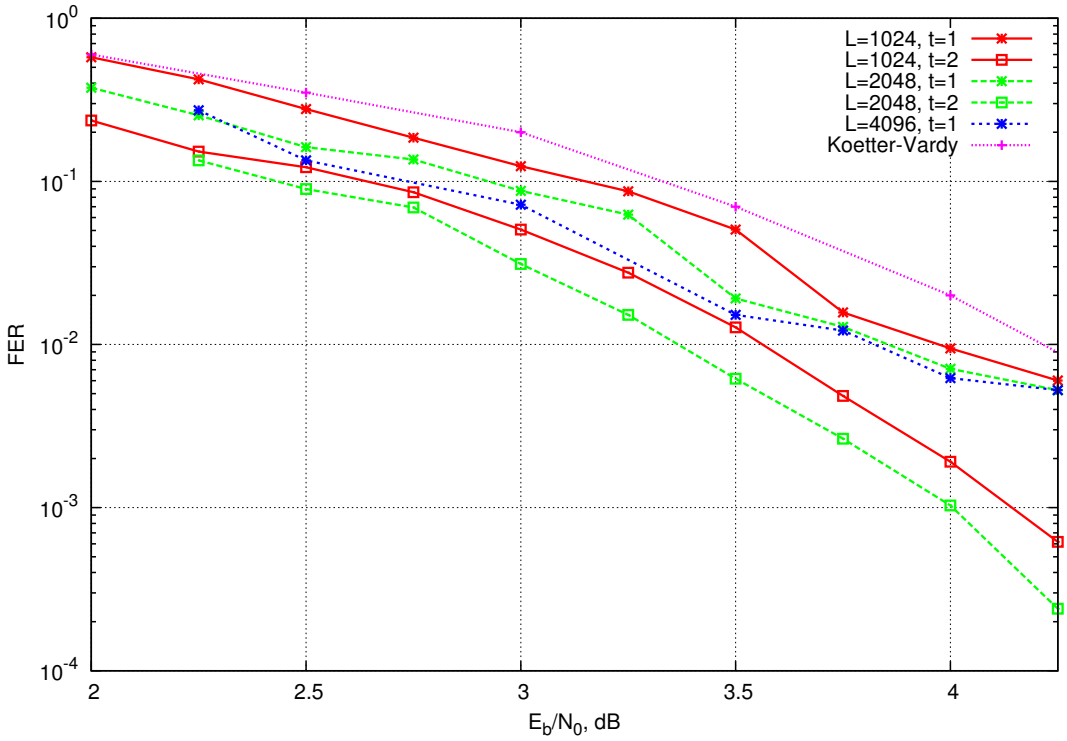

Fig. 1. Performance of the proposed decoding algorithm

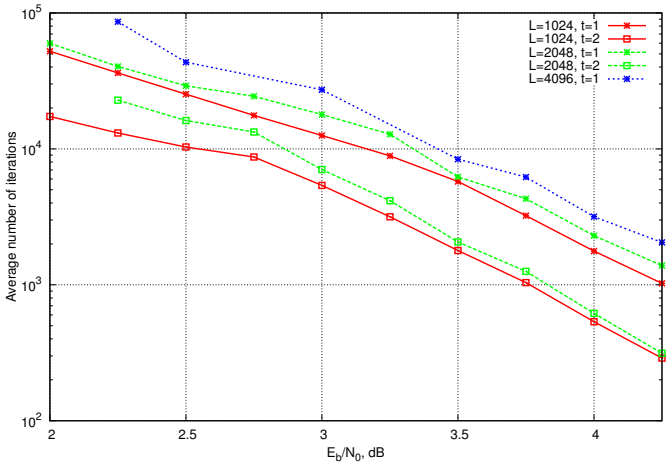

Fig. 2. Average decoding complexity

ACKNOWLEDGEMENTS

This work was partially supported by the grant of the President of Russia MK-5407.2013.9.

REFERENCES

[1] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," *IEEE Transactions on Information Theory*, vol. 49, no. 11, pp. 2809–2825, November 2003.

[2] J. Bellorado, A. Kavcic, M. Marrow, and L. Ping, "Low-complexity soft-decoding algorithms for Reed-Solomon codes–part II: Soft-input soft-output iterative decoding," *IEEE Transactions On Information Theory*, vol. 56, no. 3, March 2010.

[3] M. El-Khamy and R. McEliece, "Iterative algebraic soft-decision list decoding of reed-solomon codes," *IEEE Journal On Selected Areas In Communications*, vol. 24, no. 3, March 2006.

[4] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions On Information Theory*, vol. 55, no. 7, pp. 3051–3073, July 2009.

[5] P. Trifonov and V. Miloslavskaya, "Polar codes with dynamic frozen symbols and their decoding by directed search," in *Proceedings of IEEE Information Theory Workshop*, September 2013, pp. 1 – 5.

[6] V. Miloslavskaya and P. Trifonov, "Sequential decoding of polar codes," *IEEE Communications Letters*, vol. 18, no. 7, pp. 1127–1130, 2014.

[7] I. Dumer and K. Shabunov, "Soft-decision decoding of ReedMuller codes: Recursive lists," *IEEE Transactions On Information Theory*, vol. 52, no. 3, March 2006.

[8] I. Tal and A. Vardy, "List decoding of polar codes," in *Proceedings of IEEE International Symposium on Information Theory*, 2011.

[9] K. Chen, K. Niu, and J. Lin, "Improved successive cancellation decoding of polar codes," *IEEE Transactions On Communications*, vol. 61, no. 8, August 2013.

[10] I. Tal and A. Vardy, "How to construct polar codes," *IEEE Transactions On Information Theory*, vol. 59, no. 10, October 2013.

[11] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Transactions on Communications*, vol. 60, no. 11, pp. 3221 – 3227, November 2012.

[12] V. Miloslavskaya and P. Trifonov, "Sequential decoding of Reed-Solomon codes," in *Proceedings of International Symposium on Information Theory and Applications*, 2014.