Algebraic List-decoding of Reed-Solomon Product Codes

Farzad Parvaresh [*], **Mostafa El-Khamy** [†], **Robert J. McEliece** [‡] **and Alexander Vardy** [§]

[†] [‡] California Institute of Technology
1200 E. California Blvd.,
Pasadena, CA 91125, USA

[*] [§] University of California San Diego,
9500 Gilman Dr.,
La Jolla, CA 92093 USA

{[*]fparvaresh, [§]avardy}@ucsd.edu
{[†]mostafa, [‡]rjm}@systems.caltech.edu

***Abstract*— Product Reed-Solomon codes are widely used in data storage, optical and satellite communication systems. Reed-Solomon product codes can be regarded as evaluation of a bivariate polynomial with constraints on its X and Y-degrees. In this work, we propose polynomial time algorithms to decode Reed-Solomon product codes beyond half the minimum distance. The first algorithm is based on a generalization of the Guruswami-Sudan type decoders. We are able to show that if fraction of number of errors is smaller than $1-\sqrt[6]{4R_p}$, where R_p is the rate of the product code, then the algorithm can efficiently recover the transmitted codeword. The other algorithm is based on the fact that Reed-Solomon product codes can be viewed as subfield-subcode of a generalized Reed-Solomon code. So, the decoding algorithms for Reed-Solomon codes are inherited to decoding of RS product codes. By using this fact, we prove that if fraction of number of errors is smaller than $1-\sqrt[4]{4R_p}$ then the algorithm is able to recover the transmitted codeword.** [1]

I. Introduction

Product codes were introduced by Elias [1], who also proposed decoding them by iteratively decoding the component codes. Conventional bounded distance decoding algorithms can correct up to half the minimum distance of the code. Assume that $\mathcal{R}$ and $\mathcal{C}$ are linear codes with parameters (n_r, k_r, d_r) and (n_c, k_c, d_c). The product code $\mathcal{P} = \mathcal{R} \times \mathcal{P}$ is defined as the set of all two dimensional arrays such that each row of any array in $\mathcal{P}$ is a codeword of $\mathcal{R}$ and each column is a codeword of $\mathcal{C}$. It is well known that $\mathcal{P}$ is an $(n_p, k_p, d_p) = (n_r n_c, k_r k_c, d_r d_c)$ linear code. The rates of $\mathcal{R}$, $\mathcal{C}$ and $\mathcal{P}$ are R_r, R_c and $R_p = R_r R_c$ respectively.

It is well known that the half the distance bound is not always attainable by iteratively decoding the component codes. For example, if the decoding algorithms for the row and column component codes are capable of correcting $(d_r-1)/2$ and $(d_c-1)/2$ errors respectively, and an error rectangular block of $((d_r-1)/2+1) \times ((d_c-1)/2+1)$ occurs, iterative decoding fails although the number of errors is less than or equal to $(d_r d_c - 1)/2$ if $d_r d_c \geqslant d_r + d_c + 3$.

Reed-Solomon (RS) product codes are product codes where the component codes are Reed-Solomon codes. They are widely used in data storage and satellite broadcast systems. A number of soft iterative decoding techniques have been devised for them [2], [3]. Maximum likelihood performance analysis of Reed-Solomon product codes for both hard decision and soft decision decoding show the potential of devising improved polynomial time algorithms for decoding them [4].

Recently, there have been a lot of progress in the area of list decoding of algebraic codes. Algorithms such as Sudan [5], Guruswami-Sudan [6], Parvaresh-Vardy [7], [8], and Guruswami-Rudra [9], show that we can basically decode above the half the minimum distance of the code for some specific codes. In this work, we investigate the generalization of Guruswami-Sudan algorithm for RS product code. We will that see this generalization results in algorithms that can decode more than half the minimum distance for certain rates of a RS product code.

This paper is organized as follows. In Section II, we introduce some notation and show that a Reed-Solomon product can be represented as an evaluation of a bivariate polynomial. In Section III, we investigate two different algorithms for decoding of Reed-Solomon product codes. The first one is based on a generalization of the Guruswami-Sudan algorithm [6]. In the second one, we use the property that the RS product code is a subcode of a q-ary Reed-Muller code (and a subfield-subcode of a generalized RS code over an extension field) so any decoding algorithm for Reed-Muller codes (and RS codes) can be applied for the decoding of RS product codes.

II. Reed-Solomon Product Codes

To define the Reed-Solomon product codes, we first briefly review the Reed-Solomon codes. Let $d(X) = \sum_{i=0}^{v} d_i X^i$ be a data polynomial over $\mathbb{F}_q[X]$ [2]. Then an (n,k) Reed-Solomon code is generated by evaluating the data polynomial $d(X)$ at n distinct elements of the field forming a set called the *support* set of the code $S = \{\alpha_0, \alpha_1, ..., \alpha_{n-1}\} \subset \mathbb{F}_q$. The generated codeword is $d(S) = (d(\alpha_0), d(\alpha_1), \ldots, d(\alpha_{n-1}))$ [10].

Next, we show how a product of two RS codes can be generated by polynomial evaluation of a bivariate polynomial.

Theorem 1. *Let the data polynomial be represented as*

$$D(X,Y) = \sum_{i=0}^{v_r} \sum_{j=0}^{v_c} d_{i,j} X^i Y^j,$$

[1] Independent work by D. Augot and M. Stepanov has been done on decoding of Reed-Solomon product codes and it is submitted to ISIT06. We are planning to merge the papers in the final submission.

[2] We replace the ubiquitous $k-1$ with v.

where d_{ij}'s are the data symbols. Also denote the support set of the row and column RS codes, $\mathcal{R}$ and $\mathcal{C}$, by $S_r = \{\alpha_0, \alpha_1, ..., \alpha_{n_r-1}\} \subset \mathbb{F}_q$ and $S_c = \{\beta_0, \beta_1, ..., \beta_{n_c-1}\} \subset \mathbb{F}_q$ respectively. Then a codeword $\boldsymbol{p}$ in the RS product code $\mathcal{P} = \mathcal{R} \times \mathcal{C}$ is $\boldsymbol{p} = [\boldsymbol{p}_{i,j}]$ where $\boldsymbol{p}_{i,j} = D(\alpha_i, \beta_j)$ for $i = 0, .., n_r - 1$ and $j = 0, ..., n_c - 1$.

Proof: Since the cardinality of the code generated by bivariate polynomial evaluation described above is $q^{k_r k_c}$, which is equal to cardinality of $\mathcal{R} \times \mathcal{C}$, then it is sufficient to show that the generated code $\mathcal{P}$ is a subcode of the product code $\mathcal{R} \times \mathcal{C}$. Consider a codeword $\boldsymbol{p} \in \mathcal{P}$. The rth row is equal to $\boldsymbol{p}_{r,*} = \{D(\alpha_0, \beta_r), D(\alpha_1, \beta_r), \dots, D(\alpha_{n_r-1}, \beta_r)\}$ where

$$\begin{aligned} D(\alpha_c, \beta_r) &= \sum_{i=0}^{v_r} \sum_{j=0}^{v_c} d_{i,j} (\alpha_c)^i (\beta_r)^j \quad (1) \\ &= \sum_{i=0}^{v_r} \left(\sum_{j=0}^{v_c} d_{i,j} (\beta_r)^j \right) (\alpha_c)^i. \end{aligned}$$

Define $\gamma_{r,s} = \sum_{j=0}^{v_c} d_{s,j} (\beta_r)^j$ and the univariate polynomial $D'_r(X) = \sum_{i=0}^{v_r} \gamma_{r,i} X^i$. It is then easy to see that $\boldsymbol{p}_{r,*}$ can be generated by evaluating the modified data polynomial $D'_r(X)$ at the support set S_r; $\boldsymbol{p}_{r,*} = \{D'(\alpha_0), D'(\alpha_1), ..., D'(\alpha_{n_r-1})\}$. This proves that $\boldsymbol{p}_{r,*} \in \mathcal{R}$.

Similarly, any column c can be generated by evaluating the modified data polynomial $D''(X) = \sum_{j=0}^{v_c} \delta_{c,j} X^j$ at the support set S_c; $\boldsymbol{p}_{*,c} = \{D''(\beta_0), D''(\beta_1), ..., D''(\beta_{n_c-1})\}$, where $\delta_{c,j} = \sum_{i=0}^{v_r} d_{i,j} (\alpha_c)^i$. Thus each column is a codeword in $\mathcal{C}$.

Since each row is a codeword in $\mathcal{R}$ and each column is a codeword in $\mathcal{C}$, then $\mathcal{P}$ is a subcode of $\mathcal{R} \times \mathcal{C}$. ∎

In summary, an RS product code is defined as

$$\begin{aligned} &\mathcal{P}(S_r, S_c, v_r, v_c, q) = \{D(\alpha_i, \beta_j) : D \in \mathbb{F}_q[X, Y], \\ &\quad \alpha_i \in S_r, \beta_j \in S_c, \deg_X D < v_r + 1 \text{ and } \deg_Y D < v_c + 1\} \end{aligned}$$

It is easy to confirm that the minimum distance of $\mathcal{P}$ is indeed $d_r d_c$. From the above proof we have, each row is generated by $D'_r(X) = \sum_{i=0}^{v_r} \gamma_{r,i} X^i$. Since this univariate polynomial has at most v_r zeros, it will evaluate to at least $n_r - v_r$ non-zero values if it is non-zero. This means that at least $n_r - v_r$ columns are nonzero. Each of these columns are evaluated by the polynomial $D''(X)$. Thus each of these nonzero columns have at least $n_c - v_c$ non-zero positions. Thus if $\boldsymbol{p}$ is nonzero the number of the nonzero elements in $\boldsymbol{p}$ is at least $(n_r - v_r)(n_c - v_c)$ which is $d_r d_c$.

Corollary 2. *The number of distinct zeros of the bivariate polynomial $D(X, Y) = \sum_{i=0}^{v_r} \sum_{j=0}^{v_c} d_{i,j} X^i Y^j$ is at most $n_r v_c + n_c v_r - v_c v_r$ if $v_r < n_r$ and $v_c < n_c$.*

The (w_x, w_y) weighted degree of $D(X, Y)$ is given by

$$\begin{aligned} &\mathbf{w}\deg_{w_x, w_y} D(X, Y) \stackrel{\text{def}}{=} \\ &\quad \max\{iw_x + jw_y \,:\, D(X, Y) = \textstyle\sum_{i,j} d_{ij} X^i Y^j, d_{i,j} \neq 0\}. \end{aligned}$$

This definition can also be extended for multivariate polynomials.

Theorem 3. *The number of zeros (counting with multiplicities) of the nonzero bivariate polynomial $D(X, Y)$ evaluated over $S_r \times S_c$, where $|S_r| = n_r$ and $|S_c| = n_c$, is at most $\mathbf{w}\deg_{n_r, n_c} D(X, Y)$.*

Proof: Let $v_c = \deg_Y D(X, Y)$ and $v_r = \deg_X D(X, Y)$. For any $\alpha \in \mathbb{F}_q$, $D(\alpha, Y)$ is either the all zero polynomial or a polynomial in Y with maximum degree v_c. Define $\mathcal{G} \triangleq \{\gamma : (X - \gamma) | D(X, Y)\}$. Assuming that for each $\gamma_i \in \mathcal{G}$, m_i is the largest integer that $(X - \gamma_i)^{m_i}$ divides $Q(X, Y)$ then we can rewrite $D(X, Y)$ as follows

$$D(X, Y) = \left(\prod_{i=1}^{\ell} (X - \gamma_i)^{m_i} \right) D'(X, Y)$$

where $D'(\alpha, Y)$ is a non zero polynomial for any $\alpha \in S_r$ and $\deg_Y D'(X, Y) = v_c$.

For any $\alpha \notin \mathcal{G}$, $D(\alpha, Y)$ is nonzero so it has at most v_c many zeros. For any $\alpha = \gamma_i \in \mathcal{G}$, let assume that $D'(\gamma_i, Y)$ is zero at $\{\beta_1, \beta_2, \dots, \beta_u\}$ with multiplicity $\{r_1, r_2, \dots, r_u\}$, respectively. Then the number of zeros of $D(\gamma_i, Y)$ counting with multiplicity over $\mathcal{S}_r \times \mathcal{S}_c$ is

$$\sum_{j=1}^{u} (m_i + r_j) + (n_c - u) m_i \leqslant u m_i + v_c + (n_c - u) m_i$$

The term $(n_c - u) m_i$ is the contribution of the points that $D'(\gamma_i, \beta)$ is not zero. Also notice that $\sum_j r_j \leqslant v_c$. So, in total for all $\alpha \in \mathcal{G}$ we have

$$\sum_{i=1}^{\ell} (v_c + n_c m_i) \leqslant \ell v_c + n_c v_r$$

many zeros. Here we have used the facts that $\sum_i m_i \leqslant v_r$. Thus, total number of the zeros is upper bounded by $(n_r - \ell) v_c + \ell v_c + n_c v_r$ which is equal to $n_r v_c + v_c n_r$ and it is equal to $\mathbf{w}\deg_{n_r, n_c} D(X, Y)$. ∎

III. Error Correction Algorithms

We know that half the distance bound for the RS product code RS is given by

$$\begin{aligned} \frac{1/2\, d_p}{n_p} &\approx \frac{(1 - R_c)(1 - R_r)}{2} \\ &= \frac{1 - (R_c + R_r - R_c R_r)}{2} \\ &\leqslant \frac{1}{2} - \sqrt{R_c + R_r - R_c R_r} \\ &\leqslant \frac{1}{2} - \sqrt[4]{4R_p} \sqrt{1 - \frac{\sqrt{R_p}}{2}}, \quad (2) \end{aligned}$$

where the inequalities follow from the arithmetic and geometric mean inequality. We use this later for comparing the results of different decoding algorithms.

A. Generalizing the Guruswami-Sudan Algorithm

Using the observation in Theorem 1, we devise an algorithm for decoding Reed-Solomon product codes by generalizing the Guruswami-Sudan [6] algorithm. Assume that the Reed-Solomon product code $\mathcal{P} = \mathcal{R} \times \mathcal{C}$ is defined as in Theorem 1. The received word is $\boldsymbol{y} = [y_{i,j}]$, for $i = 1, 2, \ldots n_r$ and $j = 1, 2, \ldots n_c$, given that the codeword $\boldsymbol{p} \in \mathcal{P}$ is transmitted. The Hamming distance between $\boldsymbol{y}$ and $\boldsymbol{p}$ will be denoted by $\delta(\boldsymbol{y}, \boldsymbol{p})$.

In order to decode, we first find a trivariate interpolation polynomial $Q(X, Y, Z) \in \mathbb{F}_q[X, Y, Z]$ that passes through all the $(\alpha_i, \beta_j, y_{i,j})$ with multiplicity m. The interpolation polynomial can be found efficiently using the generalized form of the algorithm given in [11] or [7]. Assume that

$$H(X, Y) \triangleq Q(X, Y, D(X, Y)).$$

Lemma 4. *Let $\tau_m = \delta(\boldsymbol{y}, \boldsymbol{p})$. If $m(n_r n_c - \tau_m) > \mathbf{w}\deg_{n_c,n_r} H(X, Y)$, then $(Z - D(X, Y))$ divides $Q(X, Y, Z)$.*

Proof: For any $y_{i,j} = p_{i,j}$, we know $H(\alpha_i, \beta_j)$ is zero with multiplicity m, so $H(X, Y)$ has at least $m(n_r n_c - \tau_m)$ many zeros on $S_r \times S_c$. From Theorem 3, if the number of zeros of $H(X, Y)$ becomes larger than $\mathbf{w}\deg_{n_c,n_r} H$, then $H(X, Y)$ is equivalent to zero. ∎

There are many efficient algorithms that can be used for finding factors of the form $(Z - f(X, Y))$ out of $Q(X, Y, Z)$ [12]–[14].

Lemma 5. *The (n_c, n_r)-weighted degree of $H(X, Y)$ is less than or equal to the $(n_c, n_r, n_c v_r + n_r v_c)$ weighted degree of $Q(X, Y, Z)$.*

Proof: Assume that $X^i Y^j Z^\ell$ is a monomial of $Q(X, Y, Z)$. When Z is substituted by $D(X, Y)$, for this monomial we have

$$\begin{aligned} &\mathbf{w}\deg_{n_c,n_r} X^i Y^j (D(X, Y))^\ell \leqslant \\ &\quad \mathbf{w}\deg_{n_c,n_r} X^i Y^j (X^{v_r} Y^{v_c})^\ell \\ &\quad \leqslant n_c i + n_r j + (n_c v_r + n_r v_c)\ell \\ &\quad = \mathbf{w}\deg_{n_c,n_r,n_c v_r + n_r v_c} X^i Y^j Z^\ell. \end{aligned}$$

Therefore, the lemma is true for a general polynomial. ∎

Lemma 6. *There exist a nonzero trivariate polynomial $Q(X, Y, Z) \in \mathbb{F}_q[X, Y, Z]$ such that $Q(X, Y, Z)$ passes through all the $(\alpha_i, \beta_j, y_{i,j})$ for $i = 1, 2, \ldots, n_r$, $j = 1, 2, \ldots, n_c$ with multiplicity m and $\mathbf{w}\deg_{n_c,n_r,n_c v_r + n_r v_c} Q(X, Y, Z) \leqslant d_m$ where*

$$d_m = \left\lceil m(n_r n_c) \sqrt[3]{(R_r + R_c)\left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)} \right\rceil. \quad (3)$$

Proof: Following [7], [15], there exists a nonzero polynomial of weighted degree at most Δ that passes through all

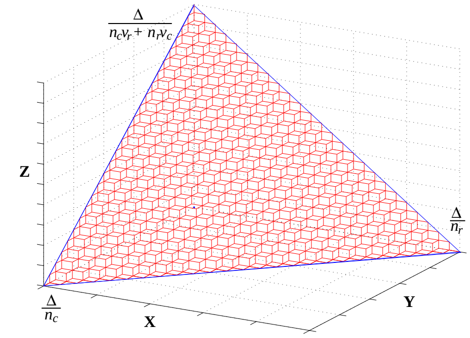

Fig. 1. The number of monomials of maximum weighted degree Δ is lower bounded by the volume of this pyramid in $\mathbb{R}^3$.

the points $(\alpha_i, \beta_j, y_{i,j})$ with multiplicity m if

$$N(\Delta) > n_r n_c \frac{m(m + 1)(m + 2)}{6}$$

where $N(\Delta)$ is the number of trivariate monomials with weighted degree at most Δ. $N(\Delta)$ can be lower bounded by the volume of the pyramid in $\mathbb{R}^3$, shown in Fig. 1. Thus,

$$N(\Delta) > \frac{1}{6} \frac{\Delta^3}{n_r n_c (n_c v_r + n_r v_c)}$$

This implies the following condition

$$\mathbf{w}\deg_{n_c,n_r,n_c v_r + n_r v_c} Q(X, Y, Z) \leqslant \left\lceil m(n_r n_c) \sqrt[3]{\left(\frac{v_r}{n_r} + \frac{v_c}{n_c}\right)\left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)} \right\rceil, \quad (4)$$

and the theorem follows. ∎

From Lemmas 4, 5 and 6, one can show the following theorem.

Theorem 7. *Assume we transmit a codeword $\boldsymbol{p} \in P(S_r, S_c, v_r, v_c, q)$ with row and column component code rates R_r and R_c respectively. Let $\boldsymbol{y} = [y_{i,j}]$ be the received word. If m is the interpolation multiplicity, then $\boldsymbol{p}$ can be efficiently list decoded from $\boldsymbol{y}$ if the Hamming distance between $\boldsymbol{y}$ and $\boldsymbol{p}$, $\tau_m = \delta(\boldsymbol{y}, \boldsymbol{c})$, is bounded by*

$$\tau_m \leqslant \left\lfloor n_c n_r \left(1 - \sqrt[3]{(R_c + R_r)\left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)}\right) - \frac{1}{m} \right\rfloor (5)$$

Corollary 8. *For an interpolation multiplicity m, the error correction radius τ_m is upper bounded by*

$$\tau_m \leqslant \left\lfloor n_p \left(1 - \sqrt[6]{4R_p} \sqrt[3]{\left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)}\right) - \frac{1}{m} \right\rfloor \quad (6)$$

where R_p and n_p are the rate and length of the product code, respectively. The upper bound on the decoding radius is maximized when R_r is equal to R_c.

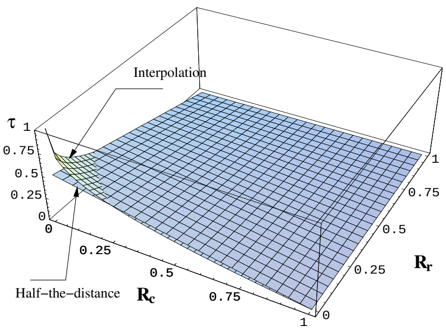

Fig. 2. $1 - \sqrt[3]{R_c + R_r}$ and the half-the-distance bound.

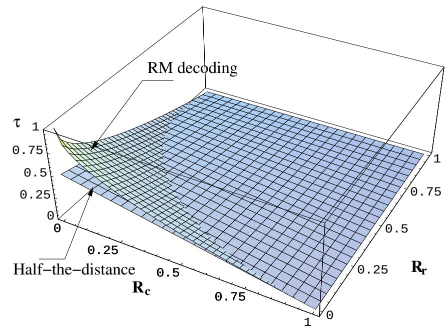

Fig. 3. $1 - \sqrt{R_c + R_r}$ and the half-the-distance bound

Proof: From the arithmetic and geometric mean inequality, $R_r + R_c \geqslant 2\sqrt{R_r R_c}$ with equality if $R_r = R_c = \sqrt{R_p}$. ∎

It thus follows that the relative *asymptotic decoding radius* of the proposed algorithm is

$$\begin{aligned} \frac{\tau}{n_p} = \lim_{m\to\infty} \frac{\tau_m}{n_p} &< 1 - \sqrt[3]{R_c + R_r} \\ &\leqslant 1 - \sqrt[6]{4R_p} \end{aligned} \tag{7}$$

Remark. When m is large, the interpolation algorithm is correcting any pattern of errors of cardinality greater than that of half the minimum distance decoder when $R_c + R_r \leqslant 0.22$. cf. Fig 4.

The following theorem shows that the number of candidates on the decoding list of our proposed algorithms does not increase with the code length, n_p, or the alphabet size, q.

Theorem 9. *For interpolating with a fixed multiplicity m, and for any received word $\boldsymbol{y} \in \mathbb{F}_q^{n_p}$, the candidate list size is upper bounded by*

$$L_m < \left\lceil m \sqrt[3]{\frac{1}{4R_p}\left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)} \right\rceil + 1. \tag{8}$$

Proof: The total number of candidate words on the list, counting plausible and implausible words, is upper bounded by the number of factors of $Q(X, Y, Z)$ which are of the form $Z - D(X, Y)$. This is upper bounded by the Z-degree of the polynomial $Q(X, Y, Z)$. From Fig. 1 and (4), we can see this can be upper bounded by

$$\begin{aligned} L_m &< \frac{\Delta}{n_c v_r + n_r v_c} \\ &\leqslant m \sqrt[3]{\left(\frac{n_r n_c}{n_c v_r + n_r v_c}\right)^2 \left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)} \\ &\approx m \sqrt[3]{\left(\frac{1}{R_c + R_r}\right)^2 \left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)} \\ &\leqslant m \sqrt[3]{\frac{1}{4R_p}\left(1 + \frac{1}{m}\right)\left(1 + \frac{2}{m}\right)}, \end{aligned}$$

where the last inequality follows from $\frac{1}{2}(R_c + R_r) \geqslant \sqrt{R_p}$ with equality if R_c is equal to R_p. ∎

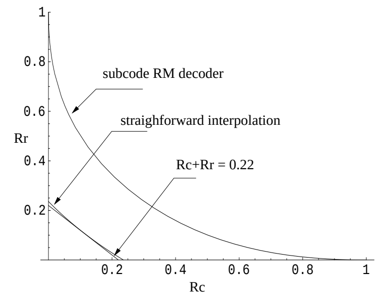

Fig. 4. Rate region that the decoders are better than the half-the-distance bound

It is worth noting that the list size of the Guruswami-Sudan algorithm is bounded by [16] (A smaller list size is preferred.)

$$L_m^{GS} \approx \left(m + \frac{1}{2}\right)\sqrt{\frac{1}{R}}. \tag{9}$$

We will now give a formulation of our generalized GS algorithm:

Algorithm 1: *Decoding of Product Reed-Solomon Codes.* Let $\boldsymbol{y} \in \mathbb{F}_q^{n_p}$ be the received word when the codeword $\boldsymbol{p} \in \mathcal{P}(S_r, S_c, v_r, v_c, q)$ is transmitted.

- Interpolate a trivariate polynomial $Q(X, Y, Z)$ such that:
  - $Q(X, Y, Z)$ passes through the points $(\alpha_i, \beta_j, y_{i,j})$ with multiplicity m.
  - The $(n_c, n_r, n_c v_r + n_r v_c)$ weighted degree of $Q(X, Y, Z)$ is less than d_m (Lemma 6).
- Factorize $Q(X, Y, Z)$ into irreducible factors. If $(Z - D(X, Y)) | Q(X, Y, Z)$, then $\boldsymbol{c} = [D(\alpha_i, \beta_j)]$, where $\alpha_i \in \mathcal{S}_c$ and $\beta_j \in \mathcal{S}_r$ is added to the list of candidates if
  - $\deg_X D(X, Y) \leqslant v_r$ and $\deg_Y D(X, Y) \leqslant v_c$
  - $\delta(\boldsymbol{c}, \boldsymbol{p}) \leqslant \tau_m$ (Theorem 7).

B. Subcode of a Reed-Muller code

Let the set of polynomials $\mathcal{P}_{RS}$ be the set of bivariate polynomials with X-degree smaller than or equal to v_r and Y-degree smaller than or equal to v_c. Evaluation of polynomials in $\mathcal{P}_{RS}$ on the elements of $\mathbb{F}_q^2$ gives the RS product code. Now assume that $\mathcal{P}_{RM}$ is the set of bivariate polynomials with total degree smaller than or equal to $v_r + v_c$. Evaluation of polynomials of $\mathcal{P}_{RM}$ over $\mathbb{F}_q^2$ gives a Reed-Muller code,

$RM_q(v_c+v_r,2)$. It is simple to see that $\mathcal{P}_{RS} \subseteq \mathcal{P}_{RM}$ or the RS product code is the subset of the Reed-Muller code. Therefore, any algorithm for decoding of the RM code can be used for decoding of RS product code. From [17], [18] we know that the $RM_q(v_c+v_r,2)$ is a subfield-subcode of a generalized Reed-Solomon code over $\mathbb{F}_{q^2}$. Thus, by decoding the generalized Reed-Solomon code using the Guruswami-Sudan algorithm [6] basically we can decode the RS product code.

Theorem 10. *[17] Assume that d is the minimum distance of q-ary Reed-Muller code $RM_q(v_c+v_r,2)$ of length n, then we can efficiently decode the Reed-Muller code if number of errors is smaller than*

$$t < n\left(1-\sqrt{1-\frac{d}{n}}\right). \tag{10}$$

Corollary 11. *Assume that the RS product code is defined over $\mathbb{F}_q$. If $n_c = n_r = q$ and $R_c + R_r < 1$ then the decoding radius of the algorithm is equal to*

$$\tau < q^2\left(1-\sqrt{R_c+R_r}\right). \tag{11}$$

Proof: When $R_c+R_r<1$ then the minimum distance of $RM_q(v_c+v_r,2)$ is equal to $d=(q-v_c-v_r)q$ and its length is q^2. Then (11) follow form (10). ∎

The RS product code $\mathcal{P}(S_r,S_c,v_r,v_c,q)$ with $|S_r|=n_r$ and $|S_c|=n_c$ is a subcode of a (punctured) GRS of length n_rn_c and minimum distance $d=(q-v_c-v_r)q$. This implies that it can be decoded using bivariate interpolation and factorization such that the asymptotic relative error capability is given by

$$\frac{\tau}{n_p} \leqslant \left(1-\sqrt{1-\frac{q(q-v_c-v_r)}{n_rn_c}}\right) \tag{12}$$

$$\approx \left(1-\sqrt{\frac{q}{n_r}R_c+\frac{q}{n_c}R_r-\frac{q^2}{n_cn_r}+1}\right). \tag{13}$$

In general one can say, that using bivariate interpolation, the asymptotic relative decoding radius is bounded by

$$\frac{\tau}{n_p} \leqslant 1-\sqrt[4]{4R_p}. \tag{14}$$

Recall that half the minimum distance of the product code is upper bounded by $\frac{1/2 d_p}{n_p} \leqslant 1/2 - \sqrt{R_c+R_r-R_cR_r}$. This implies that an algorithm with an asymptotic relative decoding radius $1-\sqrt{R_c+R_r-R_rR_c}$ will always decode beyond half the minimum distance of the code for any pattern of errors and rates R_r and R_c (cf. Fig. 5). One can see that such an algorithm exists if it is true that the RS product code $\mathcal{P}(S_r,S_c,v_r,v_c,q)$ is a subfield-subcode of the a GRS code over $\mathbb{F}_{q^2}$ with the same minimum distance of the product code, $(n_r-v_r)(n_c-v_c)$, length n_r,n_c and dimension $n_rv_c+n_cv_r-v_rv_c+1$. Existence of such a GRS and efficiently finding it remains open.

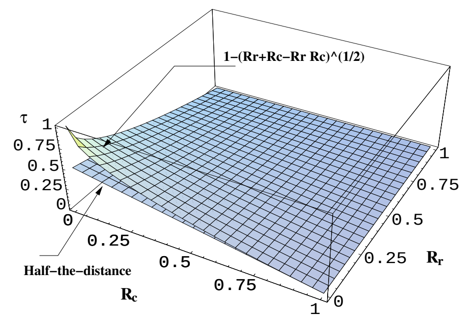

Fig. 5. $1-\sqrt{R_c+R_r-R_rR_c}$ and the half-the-distance bound

Acknowledgment

This research was supported by NSF grant no. CCF-0514881.

References

[1] P. Elias, “Error-free coding,” *IRE Trans. Inform. Theory*, vol. IT-4, pp. 29–37, Sept 1954.

[2] R. Pyndiah, A. Glavieux, A. Picart, and S. Jacq, “Near optimum decoding of product codes,” in *Proc. of IEEE GLOBECOM Conf.*, 1994.

[3] C. Argon, S. McLaughlin, and T. Souvignier, “Iterative application of the Chase algorithm on Reed-Solomon product codes,” June 2001.

[4] M. El-Khamy, “The average weight enumerator and the maximum likelihood performance of product codes,” in *International Conference on Wireless Networks, Communications and Mobile Computing, WirelessCom Information Theory Symposium*, June 2005.

[5] M. Sudan, “Decoding of Reed-Solomon codes beyond the error-corrrection bound,” *J. Complexity*, vol. 13, pp. 180–193, 1997.

[6] V. Guruswami and M. Sudan, “Improved decoding of Reed-Solomon codes and algebraic geometry codes,” *IEEE Trans. Inform. Theory*, vol. 45, no. 6, pp. 1757–1767, Sept. 1999.

[7] F. Parvaresh and A. Vardy, “Multivariate interpolation decoding beyond the guruswami-sudan radius,” in *Proc. 42 nd Annual Allerton Conference on Communication, Control and Computing, Urbana, IL., October 2004.*

[8] F. Parvaresh and A. Vardy, “Correcting errors beyond the Guruswami-Sudan radius in polynomial time,” *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 285-294, October 2005.

[9] V. Guruswami and A. Rudra, “Explicit capacity-achieving list-decodable codes,” in *Electronic Colloquium on Computational Complexity (ECCC) Tech Report TR05-133. November 2005.*

[10] F. J. MacWilliams and N. J. Sloane, *The Theory of Error Correcting Codes*. Amsterdam: North Holland, 1977.

[11] R. Kötter, “On algebraic decoding of algebraic-geo- metric and cyclic codes.” *Ph.D. Thesis, University of Linköping, Sweden*, 1996.

[12] E. Kaltofen, “Polynomial factorization: a success story,”, *ISSAC: Proceedings of the 2003 international symposium on Symbolic and algebraic computation*, pp. 3–4, Philadelphia, PA., 2003.

[13] X.-W. Wu and P. H. Siegel, “Efficient root-finding algorithm with application to list decoding of algebraic-geometric codes,” *IEEE Trans. Inform. Theory*, vol. 47, no. 6, pp. 2579–2587, September 2001.

[14] X.-W. Wu, “An algorithm for finding the roots of the polynomials over order domains,” in *Proc. of IEEE International Symposium on Information Theory, Lausane, Switezerland, June 2002*, p. 202.

[15] D. Coppersmith and M. Sudan, “Reconstructing curves in three (and higher) dimensional space from noisy data,” in *STOC’03, June 911, 2003, San Diego, California, USA.*

[16] R. J. McEliece, “The Guruswami-Sudan decoding algorithm for Reed-Solomon codes,” IPN Progress Report, Tech. Rep. 42–153, May 15 2003.

[17] R. Pellikaan and X.-W. Wu, “List decoding of q-ary Reed-Muller codes,” *IEEE Trans. Inform. Theory*, pp. 679 – 682, April 2004.

[18] T. Kasami, S. Lin, and W. Peterson, “New generalizations of the reed-muller codes–i: Primitive codes,” *IEEE Trans. Inform. Theory*, pp. 189 – 199, Mar 1968.