Explicit Low-Bandwidth Evaluation Schemes for Weighted Sums of Reed-Solomon-Coded Symbols

Han Mao Kiah, Wilton Kim, Stanislav Kruglik, San Ling, and Huaxiong Wang

School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore
Emails: hmkiah, wilt0002, stanislav.kruglik, lingsan, hkwang@ntu.edu.sg

Abstract—Motivated by applications in distributed storage, distributed computing, and homomorphic secret sharing, we study communication-efficient schemes for computing linear combinations of coded symbols. Specifically, we design low-bandwidth schemes that evaluate the weighted sum of $\ell$ coded symbols in a codeword $\mathbf{c} \in \mathbb{F}^n$, when we are given access to $d$ of the remaining components in $\mathbf{c}$. Formally, suppose that $\mathbb{F}$ is a field extension of $\mathbb{B}$ of degree $t$. Let $\mathbf{c}$ be a codeword in a Reed-Solomon code of dimension $k$ and our task is to compute the weighted sum of $\ell$ coded symbols. In this paper, for some $s < t$, we provide an explicit scheme that performs this task by downloading $d(t-s)$ sub-symbols in $\mathbb{B}$ from $d$ available nodes, whenever $d \geq \ell|\mathbb{B}| - \ell + k$. In many cases, our scheme outperforms previous schemes in the literature. Furthermore, we provide a characterization of evaluation schemes for general linear codes. Then in the special case of Reed-Solomon codes, we use this characterization to derive a lower bound for the evaluation bandwidth.

A full version [1] of this paper is accessible at: https://arxiv.org/abs/2209.03251

I. INTRODUCTION

As applications in distributed storage and distributed computation become more prevalent, communication bandwidth has become a critical performance metric of codes. In such distributed settings, data is represented as a vector $\mathbf{x} = \mathbb{F}^k$ for some finite field $\mathbb{F}$. To protect against erasures, the data is then encoded into a codeword $\mathbf{c} = (c_1, \ldots, c_n) \in \mathbb{F}^n$ and each codeword symbol $c_i$ is stored in one of $n$ nodes or servers. The beautiful Reed-Solomon (RS) code [2] allows one to recover $\mathbf{x}$ by accessing any available $k$ nodes. In other words, even if $n-k$ nodes fail, we are able to recover the data. Unfortunately, this solution is not ideal in the scenario where only one node fails. In this case, to repair a single node failure, i.e., to recover one codeword symbol, we download $k$ codeword symbols. This is costly as $k$ is usually much greater than one! Typically, the total amount of information downloaded is referred to as the repair bandwidth and the repair bandwidth problem was first introduced in [3]. Since then, there has been a flurry of research to design new codes that reduce this repair bandwidth [4], [5]. Of interest to this paper is the pioneering work of Guruswami and Wootters [6], where the authors revisit the ubiquitous Reed-Solomon codes and demonstrate that the repair bandwidth can be reduced dramatically when more than $k$ nodes are available. Later on, their methods were extended in a variety of scenarios, including multiple erasures and different parameter regimes [6]–[13].

In this paper, we consider the case where any $\ell$ nodes are unavailable. Instead of recovering the contents of these $\ell$ nodes, we recover a weighted sum or linear combination of them. This is motivated by applications in distributed computation. Specifically, in this setting, we have a computation task $\mathcal{T}$ that is split into $\ell$ smaller subtasks $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_\ell$, and we distribute these subtasks to $\ell$ servers to compute locally. However, if there is a straggler among these $\ell$ servers, that is, a server fails to complete its subtask, we are unable to determine $\mathcal{T}$. To mitigate this straggler problem, we employ coding and use $n > \ell$ servers. Specifically, we design a code and $n-\ell$ additional subtasks $\mathcal{T}_{\ell+1}, \mathcal{T}_{\ell+2}, \ldots, \mathcal{T}_n$ such that the following holds for any codeword $\mathbf{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{F}^n$: we have $c_i$ corresponds to the result of subtask $\mathcal{T}_i$ for $1 \leq i \leq n$. As before, we distribute these $n$ codeword symbols to $n$ servers. If the code is chosen to be some code that corrects $n-\ell$ erasures, then we are able to determine $c_1, c_2, \ldots, c_\ell$ as long as we have the results of any $d$ nodes. Typically, for most computation tasks, the result of $\mathcal{T}$ is obtained from a weighted sum of $c_1, c_2, \ldots, c_\ell$. Hence, the result of $\mathcal{T}$ is one symbol in $\mathbb{F}$. As before, we see that it is rather wasteful if we simply download $d$ symbols from the $d$ available nodes. Therefore, our central task is to reduce this amount of downloaded information, which we refer to as evaluation bandwidth.

To reduce the evaluation bandwidth in popular secure distributed matrix multiplication (SDMM) problem, Machado et al. employ trace polynomials (see Section II for a formal definition) [14]. However, as the authors study codes that attain the cut-set bound, the proposed codes are defined over an impractically large field. For example, when $\ell = 4$ and $T = 36$ nodes can collude, then $k = 79$, Machado et al.’s scheme requires a minimal\footnote{Here, we refer to the smallest binary field GF(2$^t$) where the evaluation bandwidth is less than $kt$, the bandwidth used by the classical approach.} binary field $\mathbb{F} = \text{GF}(2^{8085})$ and when $\mathbb{F} = \text{GF}(2^{8085})$, the number of servers $n-k$ is at most 128. In this case, when 89 nodes are available, Machado et al.’s scheme recovers the four code symbols by downloading 63.83 code symbols. In contrast, Scheme 1 (see Section III) from this work can be defined over $\mathbb{F} = \text{GF}(2^8)$ and allows up to $n-k = 252$ servers. In comparison, the evaluation bandwidth of Scheme 1 only downloads 69.5 code symbols whenever 139 nodes are available while the evaluation bandwidth of Scheme 2 only downloads 66.9 code symbols whenever 107 nodes are available. Nevertheless, Scheme 1 and Scheme 2 perform field operations over a significantly smaller field.

Our task can be stated as follows. Consider a codeword $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ of a Reed-Solomon code of length $n$ and dimension $k$. Without loss of generality, we consider the first...
evaluation points

\[ \lambda \lambda \lambda \]

Reed-Solomon code

\[ [2], [17]. \]

Of interest to this paper is the ubiquitous Reed-Solomon codes

polynomials

\[ S = \sum_{i=1}^{\ell} n_i c_i \]

downloading as little information as possible from these \( d \) nodes. For ease of exposition, we assume that these \( d \) nodes are distinct from the nodes containing \( c_1, c_2, \ldots, c_\ell \).

The paper is organized as follows. In Section II we introduce all the definitions we need and list state-of-the-art results. In Section III, we provide explicit schemes to recover \( S = \sum_{i=1}^{\ell} n_i c_i \) from some \( d \) available nodes using \textit{trace polynomials} and \textit{subspace polynomials}. While our method builds upon the frameworks of [6], [9], [10], a key feature of our scheme is that it can be adjusted according to the parameters: \( \ell \) (the number of coded symbols in the sum \( S \)) and \( d \) (the number of available nodes). In Section IV, we provide a lower bound for the evaluation bandwidth for Reed-Solomon codes by modifying the techniques in [15] and [16]. In Section V, we perform numerical analysis on our scheme and compare with those in previous work.

II. PRELIMINARIES

Let \([n]\) denote the set of integers \( \{1, 2, \ldots, n\} \) and let \( \mathbb{F} \) denote a finite field. An \( \mathbb{F} \)-linear code \( [n, k] \) code \( \mathcal{C} \) is an \( \mathbb{F} \)-linear subspace of \textit{dimension} \( k \). Each element of \( \mathcal{C} \) is called a codeword. The orthogonal complement to \( \mathcal{C} \) is called the dual code \( \mathcal{C}^\perp \). Then by definition, for each \( \mathcal{C} = (c_1, \ldots, c_i) \in \mathcal{C} \) and \( c^\perp = (c_1^\perp, \ldots, c_i^\perp) \in \mathcal{C}^\perp \), we have that \( \sum_{i=1}^{\ell} n_i c_i^\perp = 0 \). Of interest to this paper is the ubiquitous Reed-Solomon codes [2], [17].

\textbf{Definition 1.} Let \( \mathbb{F}[x] \) denote the ring of polynomials over \( \mathbb{F} \). The Reed-Solomon code \( \text{RS}(W, k) \) of dimension \( k \) with evaluation points \( W = \left\{ \omega_1, \ldots, \omega_n \right\} \subseteq \mathbb{F} \) and code rate \( R = \frac{k}{n} \) is defined as:

\[
\text{RS}(W, k) = \{(f(\omega_1), \ldots, f(\omega_n)) : f \in \mathbb{F}[x], \deg(f) < k \}. 
\]

On the other hand, the \textit{generalized} Reed-Solomon code \( \text{GRS}(W, k, \lambda) \) of dimension \( k \) with evaluation points \( W \) and multiplier \( \lambda = (\lambda_1, \ldots, \lambda_n) \in \mathbb{F}^n \), where \( \lambda_i \neq 0 \) for all \( i \in [n] \), is defined as:

\[
\text{GRS}(W, k, \lambda) = \{(\lambda_1 f(\omega_1), \ldots, \lambda_n f(\omega_n)) : f \in \mathbb{F}[x], \deg(f) < k \}. 
\]

Clearly, the generalized Reed-Solomon code \( \text{GRS}(W, k, \lambda) \) with multiplier vector \( \lambda = (1, \ldots, 1) \) is a Reed-Solomon code \( \text{RS}(W, k) \) with the same set of evaluation points and dimension. Another important fact is that the dual of \( \text{RS}(W, k) \) is \( \text{GRS}(W, n - k, \lambda) \), where the multiplier vector can be explicitly given as (see, for example [18, Ch. 10.12])

\[
\lambda_i^{-1} = \prod_{j \in [n] \setminus \{i\}} (\omega_i - \omega_j). 
\]

For convenience, when it is clear from context, \( f(x) \) denotes a polynomial of degree at most \( k - 1 \) corresponding to a codeword in \( \text{RS}(W, k) \), while \( r(x) \) denotes a polynomial of degree at most \( n-k-1 \) corresponding to a dual codeword from \( \text{GRS}(W, n - k, \lambda) \). Our low-bandwidth schemes rely on the existence of low-degree polynomials \( r(x) \) and the following relation

\[
\sum_{i=1}^{n} \lambda_i f(\omega_i) r(\omega_i) = 0. 
\]

In what follows, we refer to polynomials \( r(x) \) as \textit{parity-check polynomials} for \( \mathcal{C} \). Of interest, we employ \textit{trace polynomials} and \textit{subspace polynomials}. Formally, we describe our evaluation framework in the next subsection.

A. Trace Recovery Framework

Suppose that \( \mathbb{F} \) is a field extension of \( \mathbb{B} \) of degree \( t > 1 \). Throughout this paper, we refer to the elements of \( \mathbb{F} \) as \textit{symbols} and the elements of \( \mathbb{B} \) as \textit{sub-symbols}. We also denote the nonzero elements of \( \mathbb{F} \) as \( \mathbb{F}_* \). As mentioned in the introduction, a remarkable property of Reed-Solomon codes is the following: for any Reed-Solomon code of dimension \( k \), we can reconstruct the entire word \( v \) by accessing any \( k \) symbols in \( c \). This is also known as the \textit{maximum distance separable} (MDS) property.

Fix \( \ell \geq 1 \) and some \( \ell \)-tuple \( \kappa \in \mathbb{F}_t^\ell \). In this work, we are interested in recovering the weighted sum \( S = \sum_{i=1}^{\ell} n_i \kappa_i \) by accessing some \( d \geq k \) available nodes. The classical approach downloads \( k \) symbols from any \( k \)-subset of the \( d \) available nodes. Over \( \mathbb{B} \), the number of sub-symbols downloaded is \( k t \) and the question we want to address is: can we download less than \( k t \) sub-symbols, especially when \( d \) is much greater than \( k \)? Let us formally define a weighted-sum evaluation scheme for general \( \mathbb{F} \)-linear code.

\textbf{Definition 2.} Let \( \mathbb{F} \) be a degree-\( t \) field extension of \( \mathbb{B} \). Let \( \ell \) and \( d \) be integers and \( \mathcal{C} \) be an \( \mathbb{F} \)-linear code of length \( \ell + d \). We fix a set of coefficients \( \kappa \in \mathbb{F}_t^\ell \). A \( (\kappa, \mathcal{C}) \)-weighted-sum evaluation scheme or shortly \( (\kappa, \mathcal{C}) \)-evaluation scheme is defined to be a collection of \( d \) functions

\[
g_j : \mathbb{F} \to \mathbb{B}^{\ell} \quad \text{for } j \in [d] \]

such that for all codewords \( e \in \mathcal{C} \), the weighted sum \( \sum_{i=1}^{\ell} n_i \kappa_i c_i \) can be computed using values in the set \( \{g_j(c_{j+1}) : j \in [d]\} \). The evaluation bandwidth \( b \) is given by \( \sum_{j=1}^{d} b_j \). If the functions \( g_j(j \in [d]) \) along with the function that determines the weighted sum \( \sum_{i=1}^{\ell} n_i \kappa_i c_i \) are all \( \mathbb{B} \)-linear, we say that the \( (\kappa, \mathcal{C}) \)-evaluation scheme is \( \mathbb{B} \)-linear.

In this work, we limit our attention to \( \mathbb{B} \)-linear evaluation scheme only. To this end, we consider the \textit{trace function} \( \text{Tr} : \mathbb{F} \to \mathbb{B} \) defined as follows: \( \text{Tr}(x) = \sum_{i=0}^{t-1} x | \mathbb{B}^i | \) for \( x \in \mathbb{F} \). It is straightforward to see that the trace function is \( \mathbb{B} \)-linear. Furthermore, \( \text{Tr}(x) \) is a polynomial in \( x \) with degree \( | \mathbb{B} |^{t-1} \). Hence, in this paper, we also refer to \( \text{Tr}(x) \) as a \textit{trace polynomial}. Thus far, we have a function that maps symbols in \( \mathbb{F} \) to sub-symbols in \( \mathbb{B} \). Next, we describe a procedure to recover a symbol in \( \mathbb{F} \) using sub-symbols in \( \mathbb{B} \). If we view \( \mathbb{F} \) as a \( \mathbb{B} \)-linear subspace of dimension \( t \), we can define a \( \mathbb{B} \)-basis \( \{u_1, \ldots, u_t\} \) for \( \mathbb{F} \). Furthermore, there exists a \textit{trace-dual basis} \( \{\tilde{u}_1, \ldots, \tilde{u}_t\} \) for \( \mathbb{F} \) such that \( \text{Tr}(u_i \tilde{u}_j) = 1 \) if \( i = j \), and \( \text{Tr}(u_i \tilde{u}_j) = 0 \), otherwise. The following result plays a crucial role in our evaluation framework.

1974
Proposition 1. [19, Ch. 2] Let \( \{u_1, \ldots, u_t\} \) be a \( \mathbb{B} \)-basis of \( \mathbb{F} \). Then there exists a trace-dual basis \( \{\tilde{u}_1, \ldots, \tilde{u}_t\} \) and we can write each element \( x \in \mathbb{F} \) as \( x = \sum_{i=1}^t \text{Tr}(u_i x) \tilde{u}_i \).

Hence, following [6], our strategy to evaluate \( S \) is to compute \( t \) independent traces \( \text{Tr}(u_i S) \) \((i \in [t])\) by utilizing as little symbols in \( \mathbb{B} \) as possible.

B. Related Work

Our problem of interest bears similarities with certain previous work. Namely, one can obtain the evaluation of a weighted sum by employing a scheme that either repairs multiple erasures or evaluates a weighted sum of all data symbols. Hence, in this subsection, we review state-of-the-art schemes for these scenarios and reformulate their results in terms of \((\kappa, \mathcal{C})\)-evaluation schemes.

Low-bandwidth repair of multiple erasures. Low-bandwidth repair for Reed-Solomon codes was initiated by the pioneering paper of Guruswami and Wootters [6]. In this work, Guruswami and Wootters introduced the trace repair framework and designed an optimal repair scheme for the case of single erasure. Specifically, they used trace polynomials as parity-check polynomials and formed equations to determine the traces to download. Later, in place of trace polynomials, Dau and Milenkovic studied the use of subspace polynomials as parity-check polynomials and designed another class of optimal repair schemes [7]. This setup was then generalized to the case of two and three erasures in centralized and decentralized repair models in papers [9], [10], [15]. The trade-off between the sub-packetization and bandwidth for multiple erasures was investigated in [16], [20]. Herein, we focus on the case of small sub-packetization level (of the order \( \log(n) \)).

We also do not impose restrictions on the set of evaluation points and can choose them to be any set of specific size. This is noticeably different from related secret-sharing schemes in which each secret is represented by a single field symbol (see, for example, [21] and the references therein). Specifically, in [21], the evaluation points, which correspond to secrets, are required to satisfy certain algebraic properties.

Now, suppose that we have a low-bandwidth scheme that repairs \( \ell \) erasures by downloading information from any \( d \) available nodes. Then for \( \kappa \in \mathbb{F}_s^* \), we can first use the repair scheme to recover the coded symbols \( c_1, c_2, \ldots, c_\ell \) and then compute the weighted sum \( S = \sum_{i \in [\ell]} \kappa_i c_i \). Therefore, we obtain an \((\kappa, \mathcal{C})\)-evaluation scheme with the same bandwidth. In the following theorem, we summarize the state-of-the-art results for low-bandwidth repair of multiple erasures.

Theorem 2 ([15, Theorems 2 and 3]). Let \( \mathcal{B} \) and \( \mathcal{A} \) be two disjoint subsets of distinct points in \( \mathbb{F} \) with \( |\mathcal{B}| = \ell \) and \( |\mathcal{A}| = d \) and let \( \kappa \in \mathbb{F}_s^* \). For \( \ell \leq k \leq \ell + d \), let \( \mathcal{C} \) be the Reed-Solomon code \( \text{RS}(\mathcal{B} \cup \mathcal{A}, k) \).

Suppose that \( \mathbb{F} \) is a field extension of \( \mathcal{B} \) with degree \( t \).

(i) If \( s \leq t \) and \( d \geq |\mathcal{B}|*(2\ell - 1) - 2\ell + k + 1 \), there is an \((\kappa, \mathcal{C})\)-evaluation scheme with bandwidth \( d(t - s) \) subsymbols.

(ii) If \( t > \binom{\ell}{2} + \log_{|\mathcal{B}|}(1 + \ell + \binom{\ell}{2}(|\mathcal{B}| - 1)(|\mathcal{B}| - 1)) \) and \( d \geq |\mathcal{B}|^{t-1} - \ell + k \), there is an \((\kappa, \mathcal{C})\)-evaluation scheme with bandwidth \( d\ell - (|\mathcal{B}| - 1)\binom{\ell}{2} \) subsymbols.

Low-bandwidth function evaluation of Reed-Solomon encoded data. Recently, Lenz et al. investigated the problem of designing codes for function evaluation of information symbols transmitted over a noisy channel [22]. In the paper, the authors analyzed the trade-off between channel noise and coding rates for general coding channels. Later, Shutty and Wootters studied such a problem for Reed-Solomon codes and investigated the efficiency on the receiver side [11]. More formally, they introduced the framework for low-bandwidth recovery of weighted sum of Reed-Solomon encoded data while some of the coded symbols may be erased.

Now, suppose that we have such a scheme that can tolerate \( \ell \) erasures by employing \( d \) available nodes. Clearly, for \( \kappa \in \mathbb{F}_s^* \) the weighted sum \( S = \sum_{i \in [\ell]} \kappa_i c_i \) of coded symbols \( c_1, c_2, \ldots, c_\ell \) can be transformed to the weighted sum of information symbols. Therefore, we obtain an \((\kappa, \mathcal{C})\)-evaluation scheme with the same bandwidth. The following theorem formalizes such a result. We defer the explanation on why [11, Theorem 19] translates to the form of Theorem 3 to the extended version [1].

Theorem 3 ([11, Theorem 19]). Let \( \mathcal{B} \) and \( \mathcal{A} \) be two disjoint subsets of distinct points in \( \mathbb{F} \) with \( |\mathcal{B}| = \ell \) and \( |\mathcal{A}| = d \) and let \( \kappa \in \mathbb{F}_s^* \). For \( \ell \leq k \leq \ell + d \), let \( \mathcal{C} \) be the Reed-Solomon code \( \text{RS}(\mathcal{B} \cup \mathcal{A}, k) \). Suppose that \( \mathbb{F} \) is a field extension of \( \mathcal{B} \) with degree \( t \). For any positive number \( s \), if \( |\mathcal{B}|^{t-1}(s+1)+k+1 \leq d \leq |\mathcal{B}|^{t} - \ell \), then there is a \((\kappa, \mathcal{C})\)-evaluation scheme with bandwidth \( d|\mathcal{B}|/s \) subsymbols.

Remark 1. Our notion of low-bandwidth function evaluation of Reed-Solomon encoded data is closely related to information-theoretically secure single-client compact Homomorphic Secret-Sharing (HSS) scheme whose parties jointly compute the sum of the secrets from short messages formed in Section III, while scheme based on subspace polynomials is described in the extended version [1] due to space limitations. Properties of these schemes are summarized in the following theorem.

Theorem 4. Let \( \mathcal{B} \) and \( \mathcal{A} \) be two disjoint subsets of distinct points in \( \mathbb{F} \) with \( |\mathcal{B}| = \ell \) and \( |\mathcal{A}| = d \) and let \( \kappa \in \mathbb{F}_s^* \). For \( \ell \leq k \leq \ell + d \), let \( \mathcal{C} \) be the Reed-Solomon code \( \text{RS}(\mathcal{B} \cup \mathcal{A}, k) \).

(i) Suppose \( \mathbb{F} \) is a field extension of \( \mathcal{B} \) with degree \( t \). If \( d \geq \ell|\mathcal{B}|^{t-1} - \ell + k \), then Scheme 1 is a \((\kappa, \mathcal{C})\)-weighted sum evaluation scheme with bandwidth \( d \) subsymbols.
(ii) Suppose further \( W \) is a \( B \)-linear subspace of \( F \) with dimension \( s \). If \( d \geq \ell |B| - \ell + k \), then Scheme 2 is a \((k, C)\)-weighted sum evaluation scheme with bandwidth \( d(t-s) \) subsymbols.

Our next result is a lower bound on the evaluation bandwidth of weighted-sum schemes involving Reed-Solomon codes.

**Theorem 5.** Let \( B \) and \( A \) be two disjoint subsets of distinct points in \( F \) with \( |B| = \ell \) and \( |A| = d \). For \( \ell \leq k \leq \ell + d \), let \( C \) be the Reed-Solomon code \( RS(B \cup A, k) \). Then for any \( \kappa \in F^\ell \), we have that the evaluation bandwidth is at least \( b_{\min} \) subsymbols where

\[
b_{\min} = n_0 \left[ \log_2 |B| + (d - n_0) \left[ \log_2 |F| \right] \right], \tag{4}
\]

and

\[
n_0 = \left\lfloor \frac{L - d |B| - \left\lfloor \log_2 |\frac{B}{F} - \log_2 |F| \right\rfloor}{\left| B - \log_2 |\frac{B}{F}) \right|} \right\rfloor , \tag{5}
\]

\[
L = \frac{1}{|F|} \left( (|F| - 1)(\ell + d - k - 1) + d \right). \tag{6}
\]

It is clear that any \( \ell \leq k \) code symbols of the Reed-Solomon code are linearly independent as vectors over \( B \), hence we cannot recover \( \ell \) erased code symbols with bandwidth smaller than those for evaluation of their weighted sum. As a result, we can formulate the following corollary.

**Corollary 6.** Let \( B \) and \( A \) be two disjoint subsets of distinct points in \( F \) with \( |B| = \ell \) and \( |A| = d \). For \( \ell \leq k \leq \ell + d \), let \( C \) be the Reed-Solomon code \( RS(B \cup A, k) \). Then for erased symbols \( c_1, c_2, \ldots, c_\ell \) we have that the recovery bandwidth is at least \( b_{\min} \) subsymbols, as defined by (4), (5) and (6).

**Remark 2.** It is clear that for \( \ell = 1 \), results of Theorems 4 and 5 coincide with results for one coded symbol recovery from [6], [7].

### III. LOW-BANDWIDTH EVALUATION SCHEMES

In this section, we provide low-bandwidth schemes that evaluate the weighted sum of \( \ell \) Reed-Solomon code symbols. Throughout this section, we set \( B \triangleq \{\beta_1, \ldots, \beta_\ell\} \) and \( A \triangleq \{\alpha_1, \ldots, \alpha_d\} \) to be two disjoint subsets of distinct points in \( F \). Hence, we have that \( \ell + d \leq |F| \). We further choose \( f(x) \) to be a polynomial over \( \mathbb{F} \) with degree less than \( k \). Suppose we have \( d \geq k \) nodes and for \( j \in [d] \) we store in node \( j \) the value \( f(\alpha_j) \). Next, we fix an \( \ell \)-tuple of coefficients \( \kappa \in F^\ell \), and our task is to compute \( S = \sum_{i=1}^{\ell} \kappa_i f(\beta_i) \) by downloading as little symbols from the other \( d \) nodes. Specifically, we provide an \((\kappa, C)\)-weighted-sum evaluation scheme where \( C \) is the Reed-Solomon code \( RS(B \cup A, k) \). To this end, we consider a base field \( \mathbb{F} \) such that \( F \) is a field extension of degree \( t \) over \( \mathbb{F} \). We let \( \{u_1, \ldots, u_t\} \) be a \( B \)-basis of \( F \) and \( \{\tilde{u}_1, \ldots, \tilde{u}_t\} \) be the corresponding dual basis. Our first scheme (Scheme 1) uses trace polynomials and it demonstrates our general framework. Scheme 2 uses subspace polynomials in lieu of trace polynomials, however, due to space limitation, we defer the discussion of Scheme 2 to the extended version [1].

**Proof of Theorem 4(i).** First, it is straightforward verify that only one sub-symbol in \( B \) is downloaded from each node. Hence, the evaluation bandwidth is \( d \).

Next, we show that Scheme 1 is correct. Crucially, we prove that (9) holds for all \( i \in [t] \). Now, since \( (f(\omega))_{\omega \in B^{t \cup A}} \) belongs
to RS(\(B \cup A, k\)), we consider its dual code GRS(\(B \cup A, n - k, \lambda\)) where \(\lambda\) is defined by (1). Next, we use trace polynomials to form parity check polynomials. For \(i \in [\ell]\), let

\[
  r_i(x) = \frac{g(x)\text{Tr} \left( u_i \prod_{j \in [\ell]} (x - \beta_j) \right)}{\prod_{j \in [\ell]} (x - \beta_j)},
\]

where \(g(x)\) and \(\lambda\) are defined by (8) and (1), respectively.

Observe that \(g(\beta_j) = \kappa_j/\lambda_j\) for all \(j \in [\ell]\). Hence, \(r_i(\beta_j) = \kappa_j u_i\) for all \(j \in [\ell]\). Furthermore, the polynomial \(r_i(x)\) has degree \(\ell - 1 + \ell |B|^{t-1} - \ell = \ell |B|^{t-1} - 1\). As \(d \geq \ell |B|^{t-1} - \ell + k\), the polynomial \(r_i(x)\) is indeed a parity-check polynomial for \(f(x)\). So, it follows from (2) that for \(i \in [\ell]\),

\[
  u_i \kappa_1 f(\beta_1) + \ldots + u_i \kappa_\ell f(\beta_\ell) = - \sum_{j \in [d]} r_i(\alpha_j) \lambda_{\ell+j} f(\alpha_j) \tag{11}
\]

Applying the trace function to both sides of the equation and utilizing its linearity, we obtain

\[
  \text{Tr}(u_iS) = \text{Tr} \left( u_i \sum_{j \in [\ell]} \kappa_j f(\beta_j) \right) = - \sum_{j \in [d]} \text{Tr}(r_i(\alpha_j) \lambda_{\ell+j} f(\alpha_j)) = - \sum_{j=1}^d \sigma_{i,j} \tau_j.
\]

\[\square\]

IV. LOWER BOUND ON EVALUATION BANDWIDTH

Henceforth, \(F\) is an extension field of \(B\) with degree \(t\). Hence, each element in \(F\) can be considered as a column vector of length \(t\) over \(B\). In particular, we represent a vector in \(F^d\) as a \((t \times d)\)-matrix \(M\) over \(B\). Then we use \(\text{rank}_B(M)\) to denote the rank of \(M\) over \(B\). We also adopt the following matrix-vector notation. For a matrix \(M\), we use \(M[i, j]\) to denote the entry in the \(i\)-th row and \(j\)-th column. Moreover, we use \(M[i, :]\) and \(M[ :, j]\) to refer to the \(i\)-th row and \(j\)-th column of \(M\), respectively.

**Definition 3.** Let \(C \subseteq F^{\ell+d}\) be an \(F\)-linear code. Let us fix \(\kappa \in F^t\). Then an \((\ell + d) \times t\) matrix \(M\) whose entries belong to \(F\) is called a \((\kappa, C)\)-evaluation matrix with bandwidth \(\beta\) if the following conditions hold:

1. Columns of \(M\) are codewords in the dual code \(C^\perp\).
2. For \(i \in [\ell]\), the following holds:
   \[
   \left( \prod_{k \neq i} \kappa_k \right) M[1, :] = \ldots = \left( \prod_{k \neq \ell} \kappa_k \right) M[\ell, :].
   \]
   For convenience, we set \(M_0 = \left( \prod_{k \neq i} \kappa_k \right) M[1, :]\).
3. We have that \(\sum_{j \in [d]} \text{rank}_B M[\ell + j, :] = \beta\).

Using this notion of evaluation matrix, we characterize when a code \(C\) admits an evaluation scheme with certain bandwidth.

**Theorem 7.** Let \(C \subseteq F^{\ell+d}\) be an \(F\)-linear code and \(\kappa \in F^t\). Then \(C\) admits a \((\kappa, C)\)-weighted-sum evaluation scheme with bandwidth \(\beta\) if and only if there exists a \((\kappa, C)\)-evaluation matrix with bandwidth \(\beta\) subsums that satisfies the conditions in Definition 3.

Due to space limitation, we defer the proof of Theorem 7 to the extended version [1]. Characterizing the evaluation matrix of Reed-Solomon code in terms of parity-check polynomials of its dual code, we provide a lower bound for the evaluation bandwidth of a \((\kappa, C)\)-evaluation scheme formulated in Theorem 5. Due to space limitation, we also defer its proof to the extended version [1]. It is clear that any \(\ell \leq k\) code symbols of the Reed-Solomon code are linearly independent as vectors over \(B\), hence we cannot recover \(\ell\) erased code symbols with bandwidth smaller than those for evaluation of their weighted sum. As a result, we can formulate the corollary 6.

V. NUMERICAL RESULTS

In this section, we compare the evaluation bandwidth of our schemes with those in previous work and lower bounds. Motivated by practical applications, we consider the following scenario. We vary the number of available nodes \(d\) while the size \(\ell\) of the weighted sum is fixed. We set \(\ell = 2\) and \(k = 79\).

Consider \(F = GF(2^8)\) and \(n = |F| - \ell = 254\) nodes. Hence, the classical evaluation scheme can compute any weighted sum \(S\) of two coded symbols using any \(k = 79\) coded symbols, or equivalently, downloading 632 bits from any 79 available nodes. Suppose that \(d = 141\) nodes are available. Then Schemes 1 and 2 are able to evaluate \(S\) with 436 and 423 bits, respectively. In contrast, the schemes in Theorem 2(i) [15] and Theorem 2(ii) [15] have evaluation bandwidth 496 and 558 bits, respectively, while the schemes in Theorem 3 [11] is unable to improve the classical evaluation bandwidth. Suppose further that all \(d = 254\) nodes are available. Then the evaluation bandwidth of both Schemes 1 and 2 is 410 bits. In contrast, the schemes in Theorem 2(i, ii) [15] have evaluation bandwidth 496 and 409 bits, respectively, while the schemes in Theorem 3 [11] cannot improve the classical evaluation bandwidth. We do note that such an improvement of scheme in Theorem 2(ii) [15] over our Schemes 1 and 2 can be obtained only for the fields of small size and high number of available nodes \(d\). In Figure 1, we complete the picture and provide the evaluation bandwidth for various schemes and lower bounds for \(79 \leq d \leq 254\). We do note that during plotting the graph hereinafter we take minimum overall bandwidth for a smaller or equal number of available nodes. We also emphasize the significant gap between constructions and lower bounds that grows with the increase of \(d\). Another observation is that our scheme is beneficial against competing approaches in case of large values of \(k\) and small values of \(\ell\).

Acknowledgements.

This research / project is supported by the National Research Foundation, Singapore under its Strategic Capability Research Centres Funding Initiative, and Singapore Ministry of Education Academic Research Fund Tier 2 Grants MOE2019-T2-2083 and MOE-T2EP20121-0007. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.
REFERENCES

[1] H. M. Kiah, W. Kim, S. Kruglik, S. Ling, and H. Wang, "Explicit low-bandwidth evaluation schemes for weighted sums of reed-solomon-coded symbols," 2023. arXiv: 2209.03251.

[2] I. S. Reed and G. Solomon, “Polynomial codes over certain finite fields,” Journal of the Society for Industrial and Applied Mathematics, vol. 8, no. 2, pp. 300–304, 1960.

[3] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” IEEE Transactions on Information Theory, vol. 56, no. 9, pp. 4539–4551, 2010.

[4] S. B. Balaji, M. N. Krishnan, M. Vajha, V. Ramkumar, B. Sasidharan, and P. V. Kumar, “Erasure coding for distributed storage: An overview,” Science China Information Sciences, vol. 61, no. 10, p. 100301, 2018.

[5] S. Liu and F. Oggier, “An overview of coding for distributed storage systems,” in Network Coding and Subspace Designs. Cham: Springer International Publishing, 2018, pp. 363–383.

[6] V. Guruswami and M. Wootters, “Repairing Reed-Solomon codes,” IEEE Transactions on Information Theory, vol. 63, no. 9, pp. 5684–5698, 2017.

[7] H. Dau and O. Milenkovic, “Optimal repair schemes for some families of full-length Reed-Solomon codes,” in 2017 IEEE International Symposium on Information Theory (ISIT), 2017, pp. 346–350.

[8] I. Duursma and H. Dau, “Low bandwidth repair of the RS(10,4) Reed-Solomon code,” in 2017 Information Theory and Applications Workshop (ITA), 2017, pp. 1–10.

[9] H. Dau, I. M. Duursma, H. M. Kiah, and O. Milenkovic, “Repairing Reed-Solomon codes with multiple erasures,” IEEE Transactions on Information Theory, vol. 64, no. 10, pp. 6567–6582, 2018.

[10] S. H. Dau, T. X. Dinh, H. M. Kiah, T. T. Luong, and O. Milenkovic, “Repairing Reed-Solomon codes via subspace polynomials,” IEEE Transactions on Information Theory, vol. 67, no. 10, pp. 6395–6407, 2021.

[11] N. Shutty and M. Wootters, “Low-Bandwidth Recovery of Linear Functions of Reed-Solomon-Encoded Data,” in 13th Innovations in Theoretical Computer Science Conference (ITCS 2022), ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 215, 2022, pp. 117:1–117:19.

[12] R. Con and I. Tamo, “Nonlinear repair of Reed-Solomon codes,” IEEE Transactions on Information Theory, vol. 68, no. 8, pp. 5165–5177, 2022.

[13] A. Berman, S. Buzaglo, A. Dor, Y. Shany, and I. Tamo, “Repairing reed-solomon codes evaluated on subspaces,” IEEE Transactions on Information Theory, vol. 68, no. 10, pp. 6505–6515, 2022.

[14] R. A. Machado, R. G. L. D’Oliveira, S. El Rouayheb, and D. Heinlein, “Field trace polynomial codes for secure distributed matrix multiplication,” in 2021 XVII International Symposium “Problems of Redundancy in Information and Control Systems” (REDUNDANCY), 2021, pp. 188–193.

[15] J. Mardia, B. Bartan, and M. Wootters, “Repairing multiple failures for scalar mds codes,” IEEE Transactions on Information Theory, vol. 65, no. 5, pp. 2661–2672, 2019.

[16] W. Li, Z. Wang, and H. Jafarkhani, “On the sub-packetization size and the repair bandwidth of Reed-Solomon codes,” IEEE Transactions on Information Theory, vol. 65, no. 9, pp. 5484–5502, 2019.

[17] T. X. Dinh, L. Y. Nhi Nguyen, L. J. Mohan, S. Boztas, T. T. Luong, and S. H. Dau, “Practical considerations in repairing reed-solomon codes,” in 2022 IEEE International Symposium on Information Theory (ISIT), 2022, pp. 2607–2612.

[18] F. J. MacWilliams and N. J. A. Sloane, The Theory of Error Correcting Codes. North-Holland Pub. Co., 1977, p. 762.

[19] R. Lidl and H. Niederreiter, Finite Fields, 2nd, ser. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1996.

[20] A. Chowdury and A. Vardy, “Improved schemes for asymptotically optimal repair of mds codes,” IEEE Transactions on Information Theory, vol. 67, no. 8, pp. 5051–5068, 2021.

[21] J. Ding, C. Lin, H. Wang, and C. Xing, “Communication efficient secret sharing with small share size,” IEEE Transactions on Information Theory, vol. 68, no. 1, pp. 659–669, 2022.

[22] A. Lenz, R. Bitar, A. Wachtter-Zeh, and E. Yaakobi, “Function-correcting codes,” in 2021 IEEE International Symposium on Information Theory (ISIT), 2021, pp. 1290–1295.

[23] E. Boyle, N. Gilboa, and Y. Ishai, “Breaking the circuit size barrier for secure computation under dhd,” in Advances in Cryptology – CRYPTO 2016, 2016, pp. 509–539.

[24] E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, and M. Orrù, “Homomorphic Secret Sharing: Optimizations and Applications,” in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017, pp. 2105–2122.

[25] B. Sasidharan, and P. V. Kumar, “Erasure coding for distributed storage systems,” in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017, pp. 2105–2122.

[26] E. Boyle, N. Gilboa, Y. Ishai, H. Lin, and S. Tessaro, “Foundations of Homomorphic Secret Sharing,” in 9th Innovations in Theoretical Computer Science Conference (ITCS 2018), ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 94, 2018, pp. 21:1–21:21.

[27] L. Roy and J. Singh, “Large message homomorphic secret sharing from dcr and applications,” in Advances in Cryptology – CRYPTO 2021, 2021, pp. 687–717.

[28] I. Fossli, Y. Ishai, V. I. Kolobov, and M. Wootters, “On the Download Rate of Homomorphic Secret Sharing,” in 13th Innovations in Theoretical Computer Science Conference (ITCS 2022), ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 215, 2022, pp. 71:1–71:22.