Optimal Task Allocation and Coding Design for Secure Coded Edge Computing

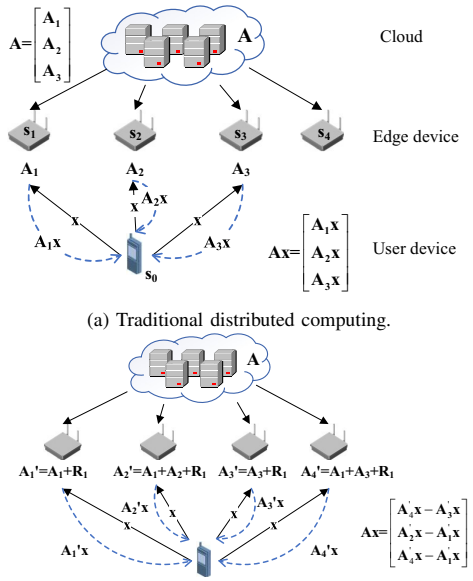
Chunming Cao[†], Jin Wang[†*], Jianping Wang[‡], Kejie Lu[⋆], Jingya Zhou[†], Admela Jukan[§], and Wei Zhao[#]
[†] School of Computer Science and Technology, Soochow University
[†] Collaborative Innovation Center of Novel Software Technology and Industrialization
[‡] Department of Computer Science, City University of Hong Kong
[⋆] Department of Computer Science and Engineering, University of Puerto Rico at Mayagüez
[§] Department of Electrical Engineering, Information Technology, Physics, Technische Universität Braunschweig
[#] American University of Sharjah

***Abstract*—In recent years, *edge computing* has attracted increasing attention for its capability of facilitating delay-sensitive applications. In the implementation of edge computing, however, data confidentiality has been raised as a major concern because edge devices may be untrustable. In this paper, we propose a design of secure and efficient edge computing by linear coding. In general, linear coding can achieve data confidentiality by adding random information to the original data before they are distributed to edge devices. To this end, it is important to carefully design code such that the user can successfully decode the final result while achieving security requirements. Meanwhile, task allocation, which selects a set of edge devices to participate in a computation task, affects not only the total resource consumption, including computation, storage, and communication, but also coding design. In this paper, we study task allocation and coding design, two highly-coupled problems in secure coded edge computing, in a unified framework. In particular, we take matrix multiplication, a fundamental building block of many distributed machine learning algorithms, as the representative computation task, and study optimal task allocation and coding design to minimize resource consumption while achieving information-theoretic security.**

I. Introduction

Edge computing, which allows computation to be done at edge devices near end users, has become a viable solution to support latency-sensitive applications, such as Internet-of-Things (IoT), virtual/augmented/mixed reality (VR/AR/MR), crowdsourcing, machine learning, and big data analytics [1]. In edge computing, the completion time of computation tasks can be reduced since the long round-trip time of moving data between users and backend datacenters is avoided [2]–[6].

In a typical edge computing scenario, a large number of edge devices can be utilized to compute the same task. Therefore, the completion time of a computation task can be further reduced by dividing the whole task into small subtasks and processing them on multiple edge devices simultaneously [7]–[9]. For example, Fig. 1 (a) shows how matrix multiplication can be performed in edge computing. In Fig. 1 (a), $\mathbf{A}$ is a predefined matrix, e.g., a pre-trained deep learning model, and is partitioned into three blocks $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$, each of which is stored in an edge device. In this case, suppose a user has a data vector $\mathbf{x}$ and wants to calculate $\mathbf{y} = \mathbf{A}\mathbf{x}$. It can first send $\mathbf{x}$ to three computing devices and then obtain $\mathbf{y}$ by combining the results from three devices.



(a) Traditional distributed computing.

(b) Secure coded distributed computing.

Fig. 1. Examples of distributed matrix multiplication in edge computing.

Although such traditional distributed computing schemes can be utilized in edge computing, there are still some challenges to be addressed. Firstly, edge devices in edge computing are usually resource limited, *i.e.*, limited storage space, computing capability, and bandwidth. Thus, task allocation, which is to identify a set of suitable edge devices for computing, becomes important. Secondly, edge devices in an edge network may belong to different service providers. They may be

*Corresponding author: Jin Wang, wjin1985@suda.edu.cn.

This work was supported in part by the National Natural Science Foundation of China (No.61672370, 61572310, 61502328), NSFC-Guangdong Joint Fund (No. U1501254), Science Technology and Innovation Committee of Shenzhen Municipality (No. JCYJ20170818095109386), Natural Science Foundation of the Higher Education Institutions of Jiangsu Province (No. 16KJB520040), Tang Scholar of Soochow University, Shanghai Key Laboratory of Intelligent Information Processing, Fudan University (No. IIPL-2016-008), the Priority Academic Program Development of Jiangsu Higher Education Institutions and National Science Foundation (No. CNS-1730325).

2575-8411/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDCS.2019.00111

untrusted. Thus, data confidentiality must be provided. To address both challenges, *coded distributed computing* (CDC) has been exploited in different distributed computation scenarios, *e.g.*, matrix multiplication [3], [4], [8]–[13], which is a critical and indispensable building block of many distributed machine learning algorithms [3], [4], [8]–[10], [14].

For matrix multiplication, most existing studies for CDC focus on the tradeoff between the latency and computing resources [3]–[6]. Few efforts have been devoted to the security aspects by fully utilizing linear coding. For instance, in [8], [9], the authors proposed the secure matrix multiplication scheme by exploiting staircase codes, with the objective of minimizing the computation latency. In [10], the authors investigated how to keep computing data confidential to edge devices. In these studies, the authors utilized the random information and the redundant computation resource to provide *information-theoretic security* (ITS) without considering the communication, computation, and storage cost. In this paper, we address the design of secure CDC for edge computing, with the objective to *minimize the total resource usage*, which has not yet been investigated in the literature.

Specifically, we consider a matrix multiplication model in which matrix $\mathbf{A}$ is pre-defined in the cloud and coded blocks of $\mathbf{A}$ are disseminated to edge devices in advance [8]–[10], [15], [16], as shown in Fig. 1. Moreover, we aim to achieve the confidentiality of $\mathbf{A}$ such that the coded block assigned to each computing device cannot be used to compute any linear combination of rows in $\mathbf{A}$, which is the ITS requirement. On the other hand, we assume that vector $\mathbf{x}$ is also a coded version of the original data, which cannot be used by any edge device to reveal the original data. In this paper, since the security of all data is rather comprehensive, we will only focus on how to achieve the confidentiality of $\mathbf{A}$. In the literature, homomorphic encryption can be exploited to compute directly on the encrypted data, but it requires high computation overhead and implementation complexity [17]–[19]. Specifically, using the latest HElib library developed in 2018, the authors in [19] demonstrated that the running time of multiplying a 628×628 dimensional matrix by a 628×1 dimensional vector in homomorphic encryption mode is 2.2 second, which is more than 2×10^3 times slower than directly multiplication on two unencrypted matrix. Therefore, homomorphic encryption may not be efficient for edge computing. In this paper, we consider the secure coded edge computing by fully utilizing the properties of the linear coding itself, which has lower computation complexity.

To achieve the ITS of $\mathbf{A}$, the cloud shall generate some random blocks and linearly combine them with the blocks in $\mathbf{A}$, as shown in Fig. 1 (b). Certainly, adding random blocks will lead to more resource usage. Therefore, in this paper, we formulate an optimization problem to minimize the total resource usage in *Secure Coded Edge Computing* (SCEC) with ITS guarantee. In particular, we will jointly study the optimal task allocation and coding scheme design for SCEC. Our objectives include: 1) completing the computation task, 2) satisfying the security requirements, and 3) minimizing the total cost of storage, computation, and communication. To the best of the authors' knowledge, no previous work has been conducted to address such a *Minimum Cost SCEC* (MCSCEC) problem for matrix multiplication by *jointly studying* the *task allocation* and *coding scheme* design. The main contributions of the paper are summarized as follows:

- We adopt linear coding to achieve *secure edge computing* by exploiting the available resources of massive edge devices in edge networks. To this end, we formally define the *Minimum Cost Secure Coded Edge Computing* (MCSCEC) problem.
- We conduct solid theoretical analysis to show the impacts of the number of random vectors used in SCEC, the number of coded vectors stored on each edge device, and the number of edge devices selected to perform computation tasks while fulfilling the aforementioned objectives. We also prove the lower bound of the MCSCEC problem, which enables us to further design the optimal task allocation schemes.
- We develop two efficient optimal algorithms to firstly obtain a set of selected edge devices, i.e., task allocation, and then design coded computing scheme, i.e., coding design, to achieve the first two objectives of the MCSCEC problem. Moreover, we also prove that the cost achieved by the proposed scheme is the minimum. In both task allocation and coding design, based on the aforementioned theoretical analyses, we present novel designs to significantly reduce both the computational complexity and decoding complexity.
- We conduct extensive simulation experiments in Sec. V with five parameters to demonstrate the effectiveness of the proposed task allocation and code design schemes. For example, the total cost obtained by the proposed MCSCEC scheme is less than 0.5% higher than the lower bound. MCSCEC can save more than 26% in resource consumption, compared to the baseline solutions, even when the size of data matrix is very large (10^4 rows).

The rest of the paper is organized as follows. Sec. II introduces system model for the MCSCEC problem. We then give theoretical analysis of the MCSCEC problem in Sec. III. In Sec. IV we design efficient optimal schemes including two task allocation algorithms and a secure code design. Considerable simulations are conducted in Sec. V. Finally, we conclude the paper in Sec. VI.

II. Problem Modeling

In this section, we first introduce the SCEC model and then present the attack model considered in this paper. At last, we give the formal definition of the MCSCEC problem and provide an overview of the framework solving the problem.

A. System Model

In this paper, we study an edge computing system $\mathbf{S} = \{s_0, s_1, \cdots, s_k\}$, in which s_0 denotes a user device and $s_j, \forall j \in \{1, \cdots, k\}$, $k \geq 2$, represents the j-th edge device. s_0 needs to perform computations on a confidential data set

represented by an $m \times l$ matrix $\mathbf{A}$. Let $\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_m$ be m row vectors of $\mathbf{A}$, each with dimension $1 \times l$. In our study, without loss of generality, we focus on the multiplication of data matrix $\mathbf{A}$ with one input vector $\mathbf{x}$. The schemes proposed in this paper can also be applied to more general cases that require multiplication of two matrices and/or multiplication of a data matrix with different input vectors.

To compute $\mathbf{Ax}$ and achieve the information-theoretic security (ITS) requirement, $\mathbf{A}$ needs to be coded, divided into blocks, and stored at edge devices. This pre-process can be done by a cloud, e.g., a parameter server which has trained a deep learning model. To code $\mathbf{A}$, r random vectors with dimension $1 \times l$, represented as $\{\mathbf{R}_1, \mathbf{R}_2, \cdots, \mathbf{R}_r\}$, need to be generated and encoded with the row vectors of $\mathbf{A}$ into $m + r$ coded vectors[1]. We note that r is a variable to be determined, which has great impacts on not only the total resource usage but also the existence of the secure linear coding scheme for the edge computing. Let $\mathbf{T} = \left[\mathbf{A}_1^\top, \cdots, \mathbf{A}_m^\top, \mathbf{R}_1^\top \cdots, \mathbf{R}_r^\top\right]^\top$ and the $(m + r) \times (m + r)$ dimensional encoding coefficient matrix $\mathbf{B} = \left[\mathbf{B}_1^\top, \cdots, \mathbf{B}_k^\top\right]^\top$, in which $\mathbf{B}_j$ is the encoding coefficient matrix of the coded vectors to be stored on s_j and $^\top$ denotes matrix transposition. Finally, coded vectors, i.e., the row vectors of $\mathbf{B}_j\mathbf{T}$, are distributed and stored on each edge device s_j, $\forall j \in \{1, \cdots, k\}$. Let $V(\mathbf{B}_j)$ denote the number of rows in $\mathbf{B}_j$. The number of coded vectors stored on s_j is $V(\mathbf{B}_j)$. We note that the encoding coefficient matrix is an empty matrix for an edge device which is not selected, i.e., none coded vector is stored on it. $\sum_{j=1}^{k} V(\mathbf{B}_j) = m + r$.

To compute $\mathbf{Ax}$, s_0 firstly sends the input vector $\mathbf{x}$ to the selected edge devices. Each edge device s_j then multiplies the coded vectors, i.e., the row vectors of $\mathbf{B}_j\mathbf{T}$, by $\mathbf{x}$ and sends the intermediate results $\mathbf{B}_j\mathbf{Tx}$ with length $V(\mathbf{B}_j)$ back to s_0. Then, s_0 can decode $\mathbf{Ax}$ after receiving these intermediate results. Specifically, since the user device s_0 receives all the $\mathbf{B}_j\mathbf{Tx}$, $\forall j \in \{1, \cdots, k\}$, it can obtain $\mathbf{BTx}$. If the encoding matrix $\mathbf{B}$ is a full rank matrix, the user device can obtain $\mathbf{Tx}$ by Gaussian elimination, in which $\mathbf{Ax}$ is composed by the first m values of $\mathbf{Tx}$. In Sec. IV-B, we give a secure linear coding design with much lower decoding complexity, in which the user device only needs to perform m subtractions on the received $m + r$ intermediate results, i.e., values, to obtain the final result $\mathbf{y}$=$\mathbf{Ax}$. To grantee that the user can decode the final result $\mathbf{y}$, we give the following availability condition.

Definition 1. ***(Availability Condition)*** *$\phi(\boldsymbol{S}, k, m, r)$ is an $(m+r)$ dimensional Linear Code for Edge Computing (LCEC) if and only if the encoding coefficient matrix $\boldsymbol{B}$ is full rank.*

In this paper, to minimize the storage, computation and communication cost, we assume that all the edge devices are available in SCEC, i.e., all the intermediate results $\{\mathbf{B}_1\mathbf{Tx}, \cdots, \mathbf{B}_k\mathbf{Tx}\}$ will be correctly computed and transmitted to the user device in a timely manner.

[1]In this paper, we use redundant vectors to assure security only. In a similar way, redundant vectors can also be used to provide processing delay guarantee.

TABLE I
NOTATIONS

Notations	Meaning
$\mathbf{S}$	the set of edge devices and a user device, $\mathbf{S} = \{s_0, s_1 \cdots, s_k\}$.
$\mathbf{A}$	the $m \times l$ dimensional data matrix.
$\mathbf{R}_p$	the p-th random vector.
$\mathbf{B}$	the encoding coefficient matrix.
$\mathbf{B}_j$	the encoding coefficient matrix for edge device s_j.
k	the number of edge devices. $K = \{1, \cdots, k\}$
m	the number of row vectors in the data matrix $\mathbf{A}$.
c_j	the unit cost of edge device s_j.
r	the number of random vectors to be encoded with the data vectors.
$V(\cdot)$	the number of row vectors in a matrix.
$Rank(\cdot)$	the rank of a vector set or matrix.
$\mathbf{A}^\top$	the transposition of matrix $\mathbf{A}$.
$\{\cdot\}_b^*$	the matrix composed by the set of row vectors with indexes from a to b in a matrix.
$(\cdot)_{p,q}$	the element in the p-th row and q-th column of a matrix.
$L(\cdot)$	the linear span space of row vectors of a matrix.

For each edge device s_j, let the unit cost of storage be c_j^s. Let the unit cost of addition and multiplication be c_j^a and c_j^m respectively, where $c_j^a \leq c_j^m$. Let the unit cost of communication from s_j to s_0 be c_j^d. Firstly, for storage, s_j needs to store $l \times 1$ dimensional input vector $\mathbf{x}$, $V(\mathbf{B}_j)$ coded vectors, i.e., $1 \times l$ dimensional row vectors of $\mathbf{B}_j\mathbf{T}$ and $V(\mathbf{B}_j)$ intermediate results (values) $\mathbf{B}_j\mathbf{Tx}$. Therefore, the storage cost is up to $(l + V(\mathbf{B}_j)l + V(\mathbf{B}_j))c_j^s$. Secondly, to compute the multiplication between a $V(\mathbf{B}_j)) \times l$ coded data matrix $\mathbf{B}_j\mathbf{T}$ and the $l \times 1$ input vector $\mathbf{x}$, the total computation cost is $V(\mathbf{B}_j)(lc_j^m + (l-1)c_j^a)$. Thirdly, after completion of the computing task, s_j shall send $V(\mathbf{B}_j)$ intermediate results (values) $\mathbf{B}_j\mathbf{Tx}$ to s_0, which will lead to up to $V(\mathbf{B}_j)c_j^d$ communication cost. Therefore, the total cost on s_j is:

$$\begin{aligned}&\sum_{j=1}^{k}(l + (l+1)V(\mathbf{B}_j))c_j^s + V(\mathbf{B}_j)(lc_j^m + (l-1)c_j^a) + V(\mathbf{B}_j)c_j^d \\ &= \sum_{j=1}^{k}(((l+1)c_j^s + lc_j^m + (l-1)c_j^a + c_j^d)V(\mathbf{B}_j) + lc_j^s)\end{aligned} \tag{1}$$

Let $c_j = (l+1)c_j^s + lc_j^m + (l-1)c_j^a + c_j^d$ be the unit cost of each edge device s_j. It reflects the involved storage, computation, and communication cost for s_j to handle one row vector. Since l and c_j^s are given values, $\sum_{j=1}^{k} lc_j^s$ is fixed. Therefore, the problem of minimizing the total cost shown in Eq. (1) is equivalent to the problem of minimizing cost $c = \sum_{j=1}^{k} V(\mathbf{B}_j)c_j$. Let $\mathbf{C} = \{c_1, \cdots, c_k\}$. Without loss of generality, we assume $0 < c_{j_1} \leq c_{j_2}$ if $1 \leq j_1 \leq j_2 \leq k$.

To facilitate the discussions, we define notations in Table I.

B. Attack Model and Secure Requirements

In this paper, we study the passive attack model in which each edge device can be a passive attacker or compromised by a passive attacker. Moreover, they do not collude with each

other. A similar passive attack model has been investigated in secure distributed computing [8]–[10], [15], [16]. In this paper, we consider the case where each edge device may want to know the information of data matrix $\mathbf{A}$. For example, in gradient-descent based algorithms, data matrix $\mathbf{A}$ is usually the personal data and input vector $\mathbf{x}$ in each iteration is only a temporary vector for obtaining the final weight vector [3], [8], [9]. We note that similar ideas can also be extended to protect both data matrix $\mathbf{A}$ and input vector $\mathbf{x}$ simultaneously, which will be investigated in our future work.

Let $H(\cdot)$ be entropy and $H(\cdot|\cdot)$ be conditional entropy. $K = \{1, \cdots, k\}$. We give the definition of the *information-theoretic security* (ITS) requirement [8], [9], [20] as follows:

Definition 2. ***(Security Condition)*** *An $(m+r)$ dimensional LCEC $\phi(\boldsymbol{S}, k, m, r)$ satisfies the requirements of ITS iff*

$$H(\boldsymbol{A}|\boldsymbol{B}_j\boldsymbol{T}) = H(\boldsymbol{A}), \forall j \in K. \tag{2}$$

Let $\mathbf{E}_m$ be the $m \times m$ dimensional identity matrix and $\mathbf{O}_{p,q}$ be the $p \times q$ dimensional zero matrix. Let $\overline{\boldsymbol{\lambda}} = \left[\begin{array}{c:c} \mathbf{E}_m & \mathbf{O}_{m,r} \end{array} \right]$ and $L(\cdot)$ be the span space of row vectors of a matrix. According to [20], Definition 2 is equivalent to: $dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})) = 0$, $\forall j \in K$.

C. Problem Definition

In this paper, we study the *Minimum Cost Secure Coded Edge Computing* (MCSCEC) problem as follows:

Definition 3. *Given an edge computing system $\boldsymbol{S}$, the costs of edge devices $\boldsymbol{C}$, and an $m \times l$ dimensional data matrix $\boldsymbol{A}$, the MCSCEC problem is to find a subset of edge devices to store the coded vectors and compute matrix multiplication, i.e., task allocation, and design a linear coded computing scheme, i.e., coding design, that satisfies the availability and security conditions, to minimize the total cost c.*

D. The MCSCEC Framework

In this section, we provide an overview of the framework to solve the MSCSEC problem where the key components in the framework will be elaborated in the following sections.

- **Task Allocation.** In this step, the cloud shall first determine two parameters: r (the number of random vectors to be encoded with data vectors) and i (the number of edge devices to participate in the SCEC). We will elaborate on this in Sec. IV-A.
- **Coded Data Distribution.** As explained in the system model, the cloud shall first generate r random vectors $\{\mathbf{R}_1, \mathbf{R}_2, \cdots, \mathbf{R}_r\}$, then generate encoding coefficient matrix $\mathbf{B} = \left[\mathbf{B}_1^\top, \cdots, \mathbf{B}_i^\top\right]^\top$. Finally, for each edge device s_j, the cloud computes and then distributes $\mathbf{B}_j\mathbf{T}$ to it. We will present the design of $\mathbf{B}$ in Sec. IV-B.
- **Coded Edge Computing.** After user device s_0 sends the input vector $\mathbf{x}$ to each edge device s_j, s_j multiplies the coded data matrix $\mathbf{B}_j\mathbf{T}$ by $\mathbf{x}$. Then, s_j sends the intermediate results $\mathbf{B}_j\mathbf{T}\mathbf{x}$ back to s_0.
- **Original Result Recovery.** When user device s_0 receives all the $\mathbf{B}_j\mathbf{T}\mathbf{x}$, $\forall j \in \{1, \cdots, i\}$, it can obtain $\mathbf{B}\mathbf{T}\mathbf{x}$. In Sec. IV-B, we will discuss how to use $\mathbf{B}\mathbf{T}\mathbf{x}$ to efficiently calculate the desired result $\mathbf{A}\mathbf{x}$.

III. Theoretical Analysis

In this section, we first show the necessary condition (Lemma 1) and existence (Lemma 2) of the optimal solution of the MCSCEC problem. After that, we give a lower bound (Theorem 1) of the MCSCEC problem and the condition (Corollary 1) that the lower bound can be achieved.

Lemma 1. *If an LCEC $\phi^*(\boldsymbol{S}, k, m, r)$ is the optimal solution of the MCSCEC problem, then for each edge device s_j, $V(\boldsymbol{B}_j) \leq r$, $\forall j \in K$.*

Proof. Firstly, we prove that $Rank(\mathbf{B}_j) \leq r$, $\forall j \in K$. Since LCEC $\phi^*(\mathbf{S}, k, m, r)$ satisfies the ITS requirements, $dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})) = 0$. We have

$$\begin{aligned} &dim(L(\mathbf{B}_j)) + dim(L(\overline{\boldsymbol{\lambda}})) \\ &= dim(L(\mathbf{B}_j) + L(\overline{\boldsymbol{\lambda}})) + dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})) \\ &= dim(L(\mathbf{B}_j) + L(\overline{\boldsymbol{\lambda}})) = Rank(\left[\begin{array}{c} \mathbf{B}_j \\ \hdashline \overline{\boldsymbol{\lambda}} \end{array} \right]), \end{aligned}$$

where the $(V(\mathbf{B}_j) + m) \times (m + r)$ dimensional matrix $\left[\begin{array}{c} \mathbf{B}_j \\ \hdashline \overline{\boldsymbol{\lambda}} \end{array} \right]$ is formed by the row vectors in $\mathbf{B}_j$ and $\overline{\boldsymbol{\lambda}}$. We have $Rank(\left[\begin{array}{c} \mathbf{B}_j \\ \hdashline \overline{\boldsymbol{\lambda}} \end{array} \right]) \leq m + r$. Since $dim(L(\mathbf{B}_j)) + dim(L(\overline{\boldsymbol{\lambda}})) = Rank(\mathbf{B}_j) + m = Rank(\left[\begin{array}{c} \mathbf{B}_j \\ \hdashline \overline{\boldsymbol{\lambda}} \end{array} \right]) \leq m + r$, we have $Rank(\mathbf{B}_j) \leq r$, for $\forall j \in K$.

Next, we use a proof by contradiction to show that, in an optimal ϕ^*, $V(\mathbf{B}_j) \leq r$ for all j. If there exists an edge device s_j that $V(\mathbf{B}_j) > r$, since $Rank(\mathbf{B}_j) \leq r$, the row vectors in $\mathbf{B}_j$ are linearly dependent. Suppose that matrix $\overline{\mathbf{B}_j}$ is composed by the rows in the maximum independent set of the row vectors in $\mathbf{B}_j$. We can obtain a new solution ϕ' as follows. Suppose that $\mathbf{B}'$ is the encoding coefficient matrix of solution ϕ'. Let $\mathbf{B}'_{j_1} = \mathbf{B}_{j_1}$, for $\forall j_1 \neq j$, and $\mathbf{B}'_j = \overline{\mathbf{B}_j}$. Since $L(\mathbf{B}'_j) = L(\overline{\mathbf{B}_j}) = L(\mathbf{B}_j)$, $Rank(\mathbf{B}) = Rank(\mathbf{B}') = m + r$. Therefore, solution ϕ' satisfies the availability condition. Moreover, since $L(\mathbf{B}'_j) = L(\mathbf{B}_j)$, $\forall j \in K$ and ϕ^* satisfies the ITS requirements, solution ϕ' also satisfies the ITS requirements. Since $Rank(\mathbf{B}_j) \leq r$ in ϕ^*, $V(\mathbf{B}'_j) = Rank(\mathbf{B}_j) \leq r < V(\mathbf{B}_j)$. Since $V(\mathbf{B}'_{j_1}) = V(\mathbf{B}_{j_1})$, for $j_1 \neq j$, and $c_{j_1} > 0$, the cost of ϕ' is smaller than ϕ. It contradicts with the assumption that ϕ^* is the optimal solution. Therefore, we have $V(\mathbf{B}_j) \leq r$, for $\forall j \in K$. □

Remark 1. *Although Lemma 1 addresses the security aspect, it also shows that the task allocated to each device is limited to r. Therefore, the completion time is bounded with certain probability, which can guarantee the processing time [3]. Nevertheless, due to limited space, the discussions about delay will be skipped in the rest of this paper.*

Based on Lemma 1, we can obtain the following lemma.

Lemma 2. *There exists an optimal solution $\phi^*(\boldsymbol{S}, k, m, r)$ of the MCSCEC problem, which satisfies that $V(\boldsymbol{B}_j^*) = r, \forall 0 < j < \lceil \frac{m+r}{r} \rceil$, $V(\boldsymbol{B}^*_{\lceil \frac{m+r}{r} \rceil}) = m - (\lceil \frac{m+r}{r} \rceil - 2)r$, and $V(\boldsymbol{B}_j^*) = 0, \forall \lceil \frac{m+r}{r} \rceil < j \leq k$.*

Proof. We prove this statement in a constructive way. Let LCEC $\phi'(\mathbf{S}, k, m, r)$ be one of the optimal solutions for the MCSCEC problem and the minimum cost be c'. Based on the given r in ϕ', we can obtain an LCEC $\phi^*(\mathbf{S}, k, m, r)$ proposed in Sec. IV-B, in which the encoding coefficient matrix $\mathbf{B}^*$ is shown in Eq. (8). Based on this $\mathbf{B}^*$, we let $i = \lceil \frac{m+r}{r} \rceil$ and we can allocate coded vectors $\mathbf{B}_j^*\mathbf{T}$ to $s_j, \forall 1 \leq j \leq i-1$, in which $V(\mathbf{B}_j^*) = r$. For device s_i, we have $V(\mathbf{B}_i^*) = m + r - (i-1)r$. Finally, we can observe that $V(\mathbf{B}_j^*) = 0, \forall i < j \leq k$.

In Theorem 3, we prove that $\phi^*(\mathbf{S}, k, m, r)$ is a feasible solution because it satisfies both the availability and security condition. The cost of ϕ^* is $c^* = \sum_{j=1}^{i-1} V(\mathbf{B}_j^*)c_j + V(\mathbf{B}_i^*)c_i = \sum_{j=1}^{i-1} rc_j + (m + r - (i-1)r)c_i$. According to Lemma 1, for the optimal ϕ', $V(\mathbf{B}'_j) \leq r$. Since $\sum_{j=1}^{k} V(\mathbf{B}'_j) = \sum_{j=1}^{k} V(\mathbf{B}_j^*) = m + r$ and $c_{j_1} \leq c_{j_2}$, if $1 \leq j_1 \leq j_2 \leq k$, we have $c^* \leq c'$.

On the other hand, since ϕ' is an optimal solution of the MCSCEC problem while ϕ^* is a feasible solution of the MCSCEC problem, we have $c' \leq c^*$. Consequently, $c^* = c'$, i.e., ϕ^* is also an optimal solution of the MCSCEC problem. Therefore, there exists an optimal solution $\phi^*(\mathbf{S}, k, m, r)$ of the MCSCEC problem, which satisfies that $V(\mathbf{B}_j^*) = r, \forall 0 < j < \lceil \frac{m+r}{r} \rceil$, $V(\mathbf{B}^*_{\lceil \frac{m+r}{r} \rceil}) = m - (\lceil \frac{m+r}{r} \rceil - 2)r$, and $V(\mathbf{B}_j^*) = 0, \forall \lceil \frac{m+r}{r} \rceil < j \leq k$. □

Remark 2. *Lemma 2 shows the existence of an optimal solution of the MCSCEC problem which satisfies a special constraint. It will be used to design task allocations shown in Sec. IV-A.*

Before we show the lower bound of the MCSCEC problem, we present and prove the following Lemma 3 – Lemma 5. To simplify the notations in the proof, we assume that $\sum_{j=j_1}^{j_2} c_j = 0$ when $j_1 > j_2$. Let i^* be the maximum i that satisfies $\sum_{j=1}^{i-1} c_j \geq (i-2)\, c_i$, $\forall i \in \{1, \cdots, k\}$. We have $2 \leq i^* \leq k$. Based on the definition of i^*, Lemma 3 – Lemma 5 show inequalities on the unit costs of the set of edge devices, which will be used to prove Theorem 1.

Lemma 3. $\sum_{j=1}^{\alpha-1} c_j \begin{cases} \geq (\alpha-2)c_{i^*}, & \textit{when } 2 \leq \alpha \leq i^*; \\ \geq (\alpha-2)c_\alpha, & \textit{when } 2 \leq \alpha \leq i^*; \\ < (\alpha-2)c_\alpha, & \textit{when } i^*+1 \leq \alpha \leq k; \end{cases}$

Proof. When $2 \leq \alpha \leq i^*$, according to the definition of i^*, we have $\sum_{j=1}^{i^*-1} c_j \geq (i^*-2)\, c_{i^*}$. Since $\alpha \leq i^*$, $c_\alpha \leq c_{i^*}$. We have $\sum_{j=\alpha}^{i^*-1} c_j \leq (i^* - \alpha)c_{i^*}$. Therefore, $\sum_{j=1}^{i^*-1} c_j - \sum_{j=\alpha}^{i^*-1} c_j \geq (i^*-2)\, c_{i^*} - (i^*-\alpha)c_{i^*}$, i.e., $\sum_{j=1}^{\alpha-1} c_j \geq (\alpha-2)c_{i^*} \geq (\alpha-2)c_\alpha$, $2 \leq \alpha \leq i^*$.

When $\alpha = i^* + 1$, according to the definition of i^*, we have $\sum_{j=1}^{i^*} c_j < (i^*-1)\, c_{i^*+1}$, i.e., $\sum_{j=1}^{\alpha-1} c_j < (\alpha-2)c_\alpha$. Similarly, when $\alpha > i^* + 1$, we have $\sum_{j=1}^{i^*} c_j < (i^*-1)\, c_{i^*+1}$. Since $\sum_{j=i^*+1}^{\alpha-1} c_j < (\alpha - i^* - 1)c_{\alpha-1}$ and $c_{\alpha-1} \geq c_{i^*}$, $\sum_{j=1}^{i^*} c_j + \sum_{j=i^*+1}^{\alpha-1} c_j < (i^*-1)\, c_{i^*+1} + (\alpha - i^* - 1)c_{\alpha-1} \leq (\alpha-2)c_{\alpha-1}$, we have $\sum_{j=1}^{\alpha-1} c_j < (\alpha-2)c_{\alpha-1} \leq (\alpha-2)c_\alpha$, $i^* + 1 \leq \alpha \leq k$. □

Lemma 4. *When $2 \leq \alpha < i^*$, $\frac{m}{\alpha-1}\sum_{j=1}^{\alpha} c_j \geq \frac{m}{i^*-1}\sum_{j=1}^{i^*} c_j$.*

Proof. When $2 \leq \alpha < i^*$, we have

$$\begin{aligned} &\frac{m}{\alpha-1}\sum_{j=1}^{\alpha} c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*} c_j \\ =&\frac{m}{(\alpha-1)(i^*-1)}\left((i^*-\alpha)\sum_{j=1}^{\alpha} c_j - (\alpha-1)\sum_{j=\alpha+1}^{i^*} c_j\right) \\ =&\frac{m}{(\alpha-1)(i^*-1)}\left(\sum_{j_2=1}^{i^*-\alpha}\left(\sum_{j_1=1}^{\alpha} c_{j_1} - (\alpha-1)c_{i^*-j_2+1}\right)\right). \end{aligned} \tag{3}$$

Since $\alpha < i^*$, we have $\alpha + 1 \leq i^*$. According to Lemma 3, $\sum_{j_1=1}^{\alpha} c_{j_1} \geq (\alpha-1)c_{i^*}$. Since $i^* \geq i^* - j_2 + 1$, $c_{i^*} \geq c_{i^*-j_2+1}$. We have $\sum_{j_1=1}^{\alpha} c_{j_1} - (\alpha-1)c_{i^*-j_2+1} \geq 0$. Since $m \geq 1$, $\alpha \geq 2$, $i^* \geq 2$ and $c_j > 0, \forall 1 \leq j \leq k$, from Eq. (3), we have $\frac{m}{\alpha-1}\sum_{j=1}^{\alpha} c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*} c_j \geq 0$. □

Lemma 5. *When $i^* + 1 < \alpha \leq k$, $\frac{m}{\alpha-2}\sum_{j=1}^{\alpha-1} c_j > \frac{m}{i^*-1}\sum_{j=1}^{i^*} c_j$.*

Proof. When $i^* + 1 < \alpha \leq k$, we have

$$\begin{aligned} &\frac{m}{\alpha-2}\sum_{j=1}^{\alpha-1} c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*} c_j \\ =&\frac{m}{(\alpha-2)(i^*-1)}\left((i^*-1)\sum_{j=i^*+1}^{\alpha-1} c_j - (\alpha-i^*-1)\sum_{j=1}^{i^*} c_j\right) \\ =&\frac{m}{(\alpha-2)(i^*-1)}\left(\sum_{j_2=1}^{\alpha-i^*-1}\left((i^*-1)c_{i^*+j_2} - \sum_{j_1=1}^{i^*} c_{j_1}\right)\right). \end{aligned}$$

Since $i^* + 1 > i^*$, according to Lemma 3, $\sum_{j_1=1}^{i^*} c_{j_1} < (i^*-1)c_{i^*+1}$. Since $i^*+1 \leq i^*+j_2$, $(i^*-1)c_{i^*+1} \leq (i^*-1)c_{i^*+j_2}$. We have $\sum_{j_1=1}^{i^*} c_{j_1} < (i^*-1)c_{i^*+j_2}$, $\forall j_2 \in \{1, \cdots, \alpha-i^*-1\}$. Consequently, when $m \geq 1$, $\alpha > i^* \geq 2$ and $c_j > 0, \forall 1 \leq j \leq$

k, we have $\frac{m}{\alpha-2}\sum_{j=1}^{\alpha-1}c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j > 0$, i.e., $\frac{m}{\alpha-2}\sum_{j=1}^{\alpha-1}c_j > \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j$. □

Remark 3. *Based on Lemma 2 – Lemma 5, we derive the lower bound of the MCSCEC problem in Theorem 1 and the condition that the lower bound can be achieved in Corollary 1.*

Theorem 1. *The cost of the optimal solution of the MCSCEC problem is no less than* $c^L = \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j$*, which is a lower bound of the MCSCEC problem.*

Proof. From Lemma 2, there exists an optimal solution $\phi^*(\mathbf{S}, k, m, r)$ of the MCSCEC problem. We let $i = \lceil \frac{m+r}{r} \rceil$. In the above ϕ^*, we have $V(\mathbf{B}_j) = r$, for $\forall 1 \le j \le i-1$, $V(\mathbf{B}_i) = m-(i-2)r$, and $V(\mathbf{B}_j) = 0$, for $\forall i < j$. Since $i = \lceil \frac{m+r}{r} \rceil < \frac{m+r}{r}+1$, we have $V(\mathbf{B}_i) > m-(\frac{m+r}{r}+1-2)r = 0$. Consequently, we have $m-(i-2)r > 0$, i.e., $r < \frac{m}{i-2}$. On the other hand, according to Lemma 1, $V(\mathbf{B}_i) \le r$, so we have $m-(i-2)r \le r$, i.e., $r \ge \frac{m}{i-1}$. Together, we have

$$\frac{m}{i-1} \le r < \frac{m}{i-2}. \tag{4}$$

Next, we consider the cost of the optimal solution ϕ^*:

$$\begin{aligned} c^* &= r\sum_{j=1}^{i-1}c_j + (m-(i-2)r)c_i \\ &= r\left(\sum_{j=1}^{i-1}c_j - (i-2)c_i\right) + mc_i. \end{aligned} \tag{5}$$

Since $m > 0, r > 0$, $i = \lceil \frac{m+r}{r} \rceil \ge 2$, there are totally k edge devices and the number of vectors allocated on each of them is no more than r in ϕ^*, $k \ge \lceil \frac{m+r}{r} \rceil = i$. Therefore, $2 \le i \le k$. We now consider i in two cases. First, if $2 \le i \le i^*$, according to Lemma 3, we have $\sum_{j=1}^{i-1}c_j \ge (i-2)c_i$. In this case, the cost shown in Eq. (5) increases with the increase of r. Therefore, the cost of the optimal solution ϕ^* satisfies:

$$c^* \ge \frac{m}{i-1}\left(\sum_{j=1}^{i-1}c_j - (i-2)c_i\right) + mc_i = \frac{m}{i-1}\sum_{j=1}^{i}c_j. \tag{6}$$

From Ineq. (6) and Lemma 4, we have

$$c^* - c^L \ge \frac{m}{i-1}\sum_{j=1}^{i}c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j \ge 0.$$

From Ineq. (6), we note that, if $i = i^*$, then

$$c^* - c^L \ge \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j = 0.$$

Secondly, if $2 \le i^* < i \le k$, according to the definition of i^*, we have $\sum_{j=1}^{i-1}c_j < (i-2)c_i$, then the cost shown in Eq. (5) decreases with the increase of r. Therefore, we have

$$c^* > \frac{m}{i-2}\left(\sum_{j=1}^{i-1}c_j - (i-2)c_i\right) + mc_i = \frac{m}{i-2}\sum_{j=1}^{i-1}c_j. \tag{7}$$

Furthermore, when $i = i^*+1$, from Ineq. (7), we have

$$c^* - c^L > \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j = 0.$$

When $i^*+1 < i \le k$, from Ineq. (7) and Lemma 5, we have

$$c^* - c^L > \frac{m}{i-2}\sum_{j=1}^{i-1}c_j - \frac{m}{i^*-1}\sum_{j=1}^{i^*}c_j > 0.$$

Therefore, $\forall i, 2 \le i \le k$, the minimum cost c^* achieved by optimal solution $\phi^*(\mathbf{S}, k, m, r)$ satisfies that $c^* \ge c^L$, i.e., c^L is the lower bound of the MCSCEC problem. □

We next prove that the lower bound can be achieved.

Corollary 1. *If m is divisible by i^*-1, there exists an optimal solution $\phi^*(\boldsymbol{S}, k, m, r)$ achieves the lower bound c^L, in which $r = \frac{m}{i^*-1}$.*

Proof. If $r = \frac{m}{i^*-1}$ and r is an integer, then $i = \lceil \frac{m+r}{r} \rceil = i^*$ and $r = \frac{m}{i-1}$ in Eq. (5) and Eq. (4). According to Eq. (6), we know that $c^* = c^L$. Moreover, based on i and r, as shown in Sec. IV-B an LCEC scheme $\phi^*(\mathbf{S}, k, m, r)$ can be designed to achieve the total cost c^* i.e., the total cost of $\phi^*(\mathbf{S}, k, m, r)$ equals to the lower bound. □

IV. The MCSCEC Schemes

In this section, we will show the optimal strategy for the MCSCEC problem, which can be divided into two stages, task allocation and coding design respectively. In task allocation shown in Sec. IV-A, we try to determine the number of edge devices needed to participate in the SCEC, i.e., i, and the number of random vectors to be encoded with data vectors, i.e., r. Based on the task allocation, in Sec. IV-B, we give a linear coding scheme for SCEC which satisfies the availability and security conditions, and achieves the minimum total cost. Moreover, in Sec. IV-C, we prove that these proposed algorithms are optimal.

A. Task Allocation Algorithms

In this subsection, we will present two task allocation algorithms, namely, TA_1 and TA_2, respectively, from two different points of view. Specifically, in TA_1, based on Lemma 3 and Corollary 1, we first determine the set of edge devices which participate in the SCEC. Based on the set of edge devices, we obtain the number of coded vectors to be allocated on each of the selected edge devices. In TA_2, based on Lemma 2 and Theorem 2, we first determine the range of r, and then find the optimal value of r based on the exhaustive method. Finally, we obtain the set of edge devices which participate in the SCEC.

Algorithm 1: Task Allocation Algorithm 1 (TA_1)

Input: m, k, C
Output: r, i, c^*
1 **if** $k = 2$ **then**
2 $\quad i^* = 2$;
3 **else if** $k > 2$ **then**
4 $\quad i^* = 2$;
5 $\quad c' = c_1$;
6 $\quad$ **while** $i^* \leq k$ **do**
7 $\quad\quad c' = c' + c_{i^*}$;
8 $\quad\quad$ **if** $c' < (i^* - 1)c_{i^*}$ **then**
9 $\quad\quad\quad$ break;
10 $\quad\quad$ **else**
11 $\quad\quad\quad i^* = i^* + 1$;
12 **if** $\lfloor \frac{m}{i^*-1} \rfloor < \lceil \frac{m}{k-1} \rceil$ **then**
13 $\quad r = \left\lceil \frac{m}{i^*-1} \right\rceil$;
14 **else**
15 $\quad r = \lfloor \frac{m}{i^*-1} \rfloor$ and $i = \lceil \frac{m+r}{r} \rceil$;
16 $\quad c_E = r \sum_{j=1}^{i-1} c_j + (m + r - (i-1)r)c_i$;
17 $\quad r = \lceil \frac{m}{i^*-1} \rceil$ and $i = \lceil \frac{m+r}{r} \rceil$;
18 $\quad c_F = r \sum_{j=1}^{i-1} c_j + (m + r - (i-1)r)c_i$;
19 $\quad$ **if** $c_E \leq c_F$ **then**
20 $\quad\quad r = \left\lfloor \frac{m}{i^*-1} \right\rfloor$;
21 $\quad$ **else if** $c_E > c_F$ **then**
22 $\quad\quad r = \left\lceil \frac{m}{i^*-1} \right\rceil$;
23 $i = \lceil \frac{m+r}{r} \rceil$;
24 **return** r, i, and $c^* = r \sum_{j=1}^{i-1} c_j + (m - (i-2)r)c_i$.

Before we find the optimal values of i and r in TA_1 algorithm and TA_2 algorithm, we first theoretically analyze the range of the values of r by the following theorem.

Theorem 2. *In the optimal solution $\phi^*(\boldsymbol{S}, k, m, r)$ of the MCSCEC problem, in which $V(\boldsymbol{B}_j) = r, \forall 0 < j < \lceil \frac{m+r}{r} \rceil$, $V(\boldsymbol{B}_{\lceil \frac{m+r}{r} \rceil}) = m - (\lceil \frac{m+r}{r} \rceil - 2)r$, and $V(\boldsymbol{B}_j) = 0, \forall \lceil \frac{m+r}{r} \rceil < j \leq k$, r satisfies $\lceil \frac{m}{k-1} \rceil \leq r \leq m$.*

Proof. In the above ϕ^*, we let $i = \lceil \frac{m+r}{r} \rceil$. According to our system model, we have $m \geq 1$, $r \geq 1$, and $i \geq 2$. Next, since the total number of coded vectors allocated to all edge devices is $m + r$, and the number of coded vectors allocated to each edge device is no more than r (according to Lemma 1), we have $rk \geq m + r$. Since k is an integer, $k \geq \lceil \frac{m+r}{r} \rceil = i$. Therefore, $2 \leq i \leq k$.

We now consider two types of i. In the first case, $i = 2$, so $2r \geq m + r$ and $r \geq m$. On the other hand, since ϕ^* is optimal, $r \leq m$. Therefore, in this case $r = m$.

In the second case, $3 \leq i \leq k$. Since $i = \lceil \frac{m+r}{r} \rceil$, $\frac{m+r}{r} \leq i < \frac{m+r}{r} + 1$. Therefore, we have $\frac{m}{i-1} \leq r < \frac{m}{i-2}$. Since $3 \leq i \leq k$ and r is an integer, $\lceil \frac{m}{k-1} \rceil \leq r < m$.

Therefore, in the optimal solution ϕ^*, $\lceil \frac{m}{k-1} \rceil \leq r \leq m$. □

1) Task Allocation Algorithm 1 (TA_1) : In Corollary 1, we have proved that if m is divisible by $i^* - 1$, and $r = \frac{m}{i^*-1}$, then the lower bound c^L can be achieved. Therefore, in *Task Allocation 1* (TA_1) algorithm, we first determine the value of i^* according to its definition shown in Sec. III. Specifically, as shown in Lemma 3, if $2 \leq \alpha \leq i^*$, $\sum_{j=1}^{\alpha-1} c_j \geq (\alpha - 2)\, c_\alpha$.

Algorithm 2: Task Allocation Algorithm 2 (TA_2)

Input: m, k, C
Output: r, i, c^*
1 $r = \lceil \frac{m}{k-1} \rceil$, $i = \lceil \frac{m+r}{r} \rceil$;
2 $c^* = r \sum_{j=1}^{i-1} c_j + (m + r - (i-1)r)c_i$;
3 $r^* = r + 1$;
4 **while** $r^* \leq m$ **do**
5 $\quad i' = \lceil \frac{m+r^*}{r^*} \rceil$;
6 $\quad c = r^* \sum_{j=1}^{i'-1} c_j + (m + r^* - (i'-1)r^*)c_{i'}$;
7 $\quad$ **if** $c < c^*$ **then**
8 $\quad\quad r = r^*$, $i = i'$ and $c^* = c$;
9 $\quad r^* = r^* + 1$;
10 **return** r, i, and c^*.

Therefore, we can find the value of i^* by using the search method (line 1 to 14 in Algorithm 1).

Then, if $\frac{m}{i^*-1}$ is an integer, we set $i = i^*$ and $r = \frac{m}{i^*-1}$. The number of coded vectors allocated on each edge device with index no larger than i^* is r and totally $i^* r = m + r$ coded vectors are allocated on the first i edge devices. In this case, according to Corollary 1, the minimum cost $\frac{m}{i^*-1} \sum_{j=1}^{i^*} c_j$ can be achieved.

On the other hand, if $\frac{m}{i^*-1}$ is not an integer, when $\lfloor \frac{m}{i^*-1} \rfloor \leq \lceil \frac{m}{k-1} \rceil \leq \lceil \frac{m}{i^*-1} \rceil$, since $r \geq \lceil \frac{m}{k-1} \rceil$, we set $r = \lceil \frac{m}{i^*-1} \rceil$. When $\lfloor \frac{m}{i^*-1} \rfloor > \lceil \frac{m}{k-1} \rceil$, we consider two cases in which $r = \lfloor \frac{m}{i^*-1} \rfloor$ and $r = \lceil \frac{m}{i^*-1} \rceil$, respectively. In both cases, the number of edge devices selected to participate the SCEC is $i = \lceil \frac{m+r}{r} \rceil$, where the first $i - 1$ edge devices are allocated with r coded vectors and edge device s_i is allocated with $m + r - (i-1)r = m - (i-2)r$ coded vectors.

According to previous discussions, the cost of the task allocation is $r \sum_{j=1}^{i-1} c_j + (m - (i-2)r)c_i$. We denote c_E as the cost when $r = \lfloor \frac{m}{i^*-1} \rfloor$ and c_F as the cost when $r = \lceil \frac{m}{i^*-1} \rceil$, respectively. We then compare c_E and c_F to choose the optimal solution. Specifically, if $c_E \leq c_F$, then we set $r = \lfloor \frac{m}{i^*-1} \rfloor$ and $i = \lceil \frac{m+r}{r} \rceil$. Otherwise, if $c_E > c_F$, we set $r = \lceil \frac{m}{i^*-1} \rceil$ and $i = \lceil \frac{m+r}{r} \rceil$. The TA_1 algorithm are shown in Algorithm 1, in which the time complexity is $O(k)$.

2) Task Allocation Algorithm 2 (TA_2) : According to Lemma 2, there exists an optimal solution $\phi^*(\mathbf{S}, k, m, r)$. Let $i = \lceil \frac{m+r}{r} \rceil$. In ϕ^*, $V(\mathbf{B}_j) = r, \forall 0 < j < i$, $V(\mathbf{B}_i) = m - (i-2)r$ and $V(\mathbf{B}_j) = 0, \forall j > i$. The total cost of $\phi^*(\mathbf{S}, k, m, r)$ is $c^* = r \sum_{j=1}^{i-1} c_j + (m - (i-2)r)c_i$. From Theorem 2, we know the value range of r in the optimal solution $\phi^*(\mathbf{S}, k, m, r)$. To minimize c^*, according to the value range of r, we can obtain the optimal r by exploiting the exhaustion algorithm. After that, we obtain the number of edge devices which participate in the SCEC, i.e., $i = \lceil \frac{m+r}{r} \rceil$. The details of TA_2 algorithm are shown in Algorithm 2. The time complexity of Algorithm 2 is $O(m + k)$.

B. Secure Linear Coding Design

From the task allocation algorithms in Sec. IV-A, we have determined the number of edge devices needed to participate

in the SCEC, i.e., i, and the number of random vectors to be encoded with data vectors, i.e., r. In this subsection, we give secure linear coding design based on the task allocation. Let $\mathbf{E}_t$ be a $t \times t$ dimensional identity matrix and let $\mathbf{E}_{m,r}$ be an $m \times r$ dimensional matrix

$$\mathbf{E}_{m,r} = \begin{bmatrix} \mathbf{E}_r \\ \vdots \\ \mathbf{E}_r \\ \{\mathbf{E}_r\}_{m-(i-2)r}^{1} \end{bmatrix}.$$

Specifically, the first $(i-2)r$ rows of $\mathbf{E}_{m,r}$ are composed by $i-2$ identity matrices $\mathbf{E}_r$ and the last $m-(i-2)r$ rows of it are composed by the first $m-(i-2)r$ rows of $r \times r$ dimensional identity matrix $\mathbf{E}_r$.

Let $\mathbf{O}_{p,q}$ be the $p \times q$ dimensional zero matrix. We define the $(m+r)\times(m+r)$ dimensional encoding coefficient matrix $\mathbf{B}$ as follows.

$$\mathbf{B} = \begin{bmatrix} \mathbf{O}_{r,m} & \mathbf{E}_r \\ \mathbf{E}_m & \mathbf{E}_{m,r} \end{bmatrix}. \tag{8}$$

Therefore, we have $\mathbf{B}_1 = \begin{bmatrix} \mathbf{O}_{r,m} & \mathbf{E}_r \end{bmatrix}$, $\mathbf{B}_j = \{\mathbf{B}\}_{jr}^{(j-1)r+1} = \begin{bmatrix} \{\mathbf{E}_m\}_{(j-1)r}^{(j-2)r+1} & \mathbf{E}_r \end{bmatrix}, \forall j \in \{2, \cdots, i-1\}$, and $\mathbf{B}_i = \mathbf{B}_{m+r}^{(i-1)r+1} = \begin{bmatrix} \{\mathbf{E}_m\}_{m}^{(i-2)r+1} & \{\mathbf{E}_r\}_{m-(i-2)r}^{1} \end{bmatrix}$.

Next, we prove that the LCEC, designed based on encoding coefficient matrix $\mathbf{B}$, satisfies the availability and security conditions. Let $(\mathbf{B})_{p,q}$ is the p-th row q-th column element of $\mathbf{B}$, we have the following theorem:

Theorem 3. If the encoding coefficient matrix of an LCEC $\phi(\mathbf{S}, k, m, r)$ is $\mathbf{B}$ defined above, then the LCEC $\phi(\mathbf{S}, k, m, r)$ satisfies the availability and security conditions.

Proof. Let $\mathbf{B}' = \begin{bmatrix} \mathbf{E}_m & \mathbf{E}_{m,r} \\ \mathbf{O}_{r,m} & \mathbf{E}_r \end{bmatrix}$. $Rank(\mathbf{B}) = Rank(\mathbf{B}')$. Since $\mathbf{B}'$ is an upper triangular matrix and $(\mathbf{B}')_{p,p} = 1, \forall p \in \{1, \cdots, m+r\}$, $\mathbf{B}'$ is full rank. Therefore, $\mathbf{B}$ is full rank and the LCEC $\phi(\mathbf{S}, k, m, r)$ satisfies the availability condition.

For each selected edge device s_j, we have $1 \le j \le i$.

When $j = 1$, since all the elements in the 1-th to m-th column of matrix $\mathbf{B}_1$ are 0, $\mathbf{B}_1\mathbf{T}$ are linear combinations of random vectors, i.e., s_1 cannot obtain any nonzero vector which is the linear combination of row vectors of $\mathbf{A}$.

When $2 \le j \le i$, let $\overline{\boldsymbol{\lambda}} = \begin{bmatrix} \mathbf{E}_m & \mathbf{O}_{m,r} \end{bmatrix}$. We have

$$\begin{aligned} &dim(L(\mathbf{B}_j)) + dim(L(\overline{\boldsymbol{\lambda}})) \\ &= dim(L(\mathbf{B}_j) + L(\overline{\boldsymbol{\lambda}})) + dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})) \\ &= Rank(\mathbf{B}'_j) + dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})), \end{aligned}$$

in which $\mathbf{B}'_j = \begin{bmatrix} \overline{\boldsymbol{\lambda}} \\ \mathbf{B}_j \end{bmatrix}$.

For $2 \le j < i$, since $dim(L(\mathbf{B}_j)) = r$ and $dim(L(\overline{\boldsymbol{\lambda}})) = m$, we have $dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})) = m + r - Rank(\mathbf{B}'_j)$.

$$\mathbf{B}'_j = \begin{bmatrix} \mathbf{E}_m & \mathbf{O}_{m,r} \\ \{\mathbf{E}_m\}_{(j-1)r}^{(j-2)r+1} & \mathbf{E}_r \end{bmatrix}.$$

Since $\mathbf{B}'_j$ is a lower triangular matrix and $(\mathbf{B}'_j)_{p,p} = 1, \forall p \in \{1, \cdots, m+r\}$, $Rank(\mathbf{B}'_j) = m + r$. Therefore, we have $dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})) = m + r - (m + r) = 0$.

For $j = i$. Since $dim(L(\mathbf{B}_i)) = m - (i-2)r$ and $dim(L(\overline{\boldsymbol{\lambda}})) = m$, we have $dim(L(\mathbf{B}_i) \cap L(\overline{\boldsymbol{\lambda}})) = 2m - (i-2)r - Rank(\mathbf{B}'_j)$.

$$\begin{aligned} \mathbf{B}'_j &= \begin{bmatrix} \mathbf{E}_m & \mathbf{O}_{m,r} \\ \{\mathbf{E}_m\}_{m}^{(i-2)r+1} & \{\mathbf{E}_r\}_{m-(i-2)r}^{1} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{E}_m & \mathbf{O}_{m,m-(i-2)r} & \mathbf{O}_{m,(i-1)r-m} \\ \{\mathbf{E}_m\}_{m}^{(i-2)r+1} & \mathbf{E}_{m-(i-2)r} & \mathbf{O}_{m-(i-2)r,(i-1)r-m} \end{bmatrix}. \end{aligned}$$

$$Rank\left(\mathbf{B}'_j\right) = Rank\left(\begin{bmatrix} \mathbf{E}_m & \mathbf{O}_{m,m-(i-2)r} \\ \{\mathbf{E}_m\}_{m}^{(i-2)r+1} & \mathbf{E}_{m-(i-2)r} \end{bmatrix}\right).$$

$\begin{bmatrix} \mathbf{E}_m & \mathbf{O}_{m,m-(i-2)r} \\ \{\mathbf{E}_m\}_{m}^{(i-2)r+1} & \mathbf{E}_{m-(i-2)r} \end{bmatrix}$ is a $(2m-(i-2)r) \times (2m-(i-2)r)$ dimensional lower triangular matrix. $Rank(\mathbf{B}'_j) = 2m - (i-2)r$. Thus, $dim(L(\mathbf{B}_i) \cap L(\overline{\boldsymbol{\lambda}})) = 0$.

Therefore, for $\forall 1 \le j \le i$, $dim(L(\mathbf{B}_j) \cap L(\overline{\boldsymbol{\lambda}})) = 0$, i.e., $L(\mathbf{B}_j)$ does not include any $m+r$ dimensional nonzero vector in $L(\overline{\boldsymbol{\lambda}})$. $\mathbf{B}_j$ satisfies the $H(\mathbf{A}|\mathbf{B}_j\mathbf{T}) = H(\mathbf{A})$ as shown in Definition 1. Therefore, $\mathbf{B}$ satisfies the security condition. □

Based on the encoding coefficient matrix $\mathbf{B}$, the coded data matrix $\mathbf{B}_j\mathbf{T}$ is migrated and stored on each edge device s_j, $\forall j \in \{1, \cdots, i\}$. There is no coded data matrix stored on $\{s_{i+1}, \cdots, s_k\}$. After user device s_0 sends the input vector $\mathbf{x}$ to each edge device s_j, $\forall j \in \{1, \cdots, i\}$, s_j multiplies the coded data matrix $\mathbf{B}_j\mathbf{T}$ by $\mathbf{x}$. Then, it sends the intermediate results $\mathbf{B}_j\mathbf{T}\mathbf{x}$ back to s_0. After the user device receives the intermediate results $\{\mathbf{B}_1\mathbf{T}\mathbf{x}, \cdots, \mathbf{B}_i\mathbf{T}\mathbf{x}\}$ from i edge devices, it can obtain $\mathbf{BTx} = \begin{bmatrix} \mathbf{B}_1\mathbf{T}\mathbf{x} \\ \vdots \\ \mathbf{B}_i\mathbf{T}\mathbf{x} \end{bmatrix}$. Then, it can decode and recover the required result $\mathbf{Ax} = [\mathbf{A}_1\mathbf{x}, \cdots, \mathbf{A}_m\mathbf{x}]^\top$, in which $\mathbf{A}_p\mathbf{x} = (\mathbf{BTx})_{r+p,1} - (\mathbf{BTx})_{p-(\lceil \frac{p}{r} \rceil - 1)r,1}, \forall 1 \le p \le m$. For the computational complexity of decoding operations in the user device, we note that in the proposed SCEC scheme, the user device only needs to perform subtractions m times on the received $m+r$ intermediate results, in order to obtain the result of $\mathbf{Ax}$, which is much lower than that when $\mathbf{Ax}$ is locally computed at the user device.

C. Optimality Analysis

In this subsection, we will prove that the proposed task allocation in Sec. IV-A and coding design in Sec. IV-B are optimal.

Theorem 4. *The solution composed by TA_1 algorithm and secure coding scheme proposed in Sec. IV-B is the optimal solution of MCSCEC problem.*

Proof. From Lemma 2, there exists an optimal solution that satisfies $V(\mathbf{B}_j) = r, \forall 0 < j < i_r$, $V(\mathbf{B}_{i_r}) = m + r - (i_r - 1)r$

and $V(\mathbf{B}_j) = 0, \forall j > i_r$, in which $i_r = \lceil \frac{m+r}{r} \rceil$. We have the total cost of the optimal solution is

$$c^{(r)} = r \sum_{j=1}^{i_r - 1} c_j + (m + r - (i_r - 1)r) c_{i_r}.$$

When r is not determined, we can treat the total cost $c^{(r)}$ as the function of r. Let $i_{r-1} = \lceil \frac{m+r-1}{r-1} \rceil$ and $i_{r+1} = \lceil \frac{m+r+1}{r+1} \rceil$. We have $2 \leq i_{r+1} \leq i_r \leq i_{r-1}$ and

$$\begin{aligned} c^{(r+1)} - c^{(r)} = (r+1) & \sum_{j=1}^{i_{r+1}-1} c_j \\ & + (m + r + 1 - (i_{r+1} - 1)(r+1)) c_{i_{r+1}} \\ & - r \sum_{j=1}^{i_r - 1} c_j - (m + r - (i_r - 1)r) c_{i_r}. \end{aligned} \tag{9}$$

If $i_r = i_{r+1}$,

$$c^{(r+1)} - c^{(r)} = \sum_{j=1}^{i_{r+1}-1} c_j - (i_{r+1} - 2) c_{i_{r+1}}. \tag{10}$$

If $i_r = i_{r+1} + 1$,

$$\begin{aligned} c^{(r+1)} - c^{(r)} = & \sum_{j=1}^{i_{r+1}-1} c_j \\ & - (r - (m + r + 1 - (i_{r+1} - 1)(r+1))) c_{i_{r+1}} \\ & - (m + r - (i_r - 1)r) c_{i_r}, \end{aligned} \tag{11}$$

in which the sum of coefficients of $c_{i_{r+1}}$ and c_{i_r} is $-(r-(m+r+1-(i_{r+1}-1)(r+1)))+(m+r-(i_r-1)r) = -(i_{r+1}-2)$.

If $i_r \geq i_{r+1} + 2$,

$$\begin{aligned} c^{(r+1)} - c^{(r)} = & \sum_{j=1}^{i_{r+1}-1} c_j \\ & - (r - (m + r + 1 - (i_{r+1} - 1)(r+1))) c_{i_{r+1}} \\ & - r \sum_{j=i_{r+1}+1}^{i_r - 1} c_j - (m + r - (i_r - 1)r) c_{i_r}, \end{aligned} \tag{12}$$

in which the sum of coefficients of $\{c_{i_{r+1}}, c_{i_{r+1}+1}, \cdots, c_{i_r}\}$ is $-(r-(m+r+1-(i_{r+1}-1)(r+1)))+r(i_r-1-i_{r+1}-1+1)+(m+r-(i_r-1)r) = -(i_{r+1}-2)$.

According to Eq. (10)-Eq. (12), since $c_{i_{r+1}} \leq c_{i_r}$, we have

$$c^{(r+1)} - c^{(r)} \geq \sum_{j=1}^{i_{r+1}-1} c_j - (i_{r+1} - 2) c_{i_r}. \tag{13}$$

When $r \geq \lceil \frac{m}{i^*-1} \rceil$, we have $i^* \geq \frac{m+r}{r}$. Since i^* is an integer, $i^* \geq \lceil \frac{m+r}{r} \rceil = i_r \geq i_{r+1}$. According to Lemma 3, $\sum_{j=1}^{i_{r+1}-1} c_j \geq (i_{r+1}-2)c_{i^*} \geq (i_{r+1}-2)c_{i_r}$. Therefore, when $r \geq \lceil \frac{m}{i^*-1} \rceil$, $c^{(r+1)} - c^{(r)} \geq 0$.

According to Eq. (10) - Eq. (12), with similar analysis, since $c_{i_r} \leq c_{i_{r-1}}$, we have

$$c^{(r)} - c^{(r-1)} \leq \sum_{j=1}^{i_r - 1} c_j - (i_r - 2) c_{i_r}. \tag{14}$$

When $r \leq \lfloor \frac{m}{i^*-1} \rfloor$, we have $i^* \leq \frac{m+r}{r} \leq \lceil \frac{m+r}{r} \rceil = i_r$. When $i^* = i_r$, we have $\frac{m}{i^*-1} = r$. From Corollary 1, c_{i_r} is the lower bound c^L. When $i^* < i_r$, according to Lemma 3, $\sum_{j=1}^{i_r-1} c_j < (i_r - 2)c_{i_r}$. Therefore, when $r \leq \lfloor \frac{m}{i^*-1} \rfloor$, we have $c^{(r)} - c^{(r-1)} < 0$.

From the above theoretical analysis, when $r \leq \lfloor \frac{m}{i^*-1} \rfloor$, $c^{(r)}$ decreases as r increases and when $r \geq \lceil \frac{m}{i^*-1} \rceil$, $c^{(r)}$ increases as r increases. From Theorem 2, we have the value range of r is $\lceil \frac{m}{k-1} \rceil \leq r \leq m$. Since $i^* \geq 2$, $\lceil \frac{m}{i^*-1} \rceil \leq m$. If $\lfloor \frac{m}{i^*-1} \rfloor < \lceil \frac{m}{k-1} \rceil < \lceil \frac{m}{i^*-1} \rceil$, the total cost can get the optimal value when $r = \lceil \frac{m}{i^*-1} \rceil$. If $\lfloor \frac{m}{i^*-1} \rfloor \geq \lceil \frac{m}{k-1} \rceil$, since $\lfloor \frac{m}{i^*-1} \rfloor = \lceil \frac{m}{i^*-1} \rceil$ or $\lfloor \frac{m}{i^*-1} \rfloor + 1 = \lceil \frac{m}{i^*-1} \rceil$, the minimum total cost is the lower cost achieved when we select $r = \lfloor \frac{m}{i^*-1} \rfloor$ or $r = \lceil \frac{m}{i^*-1} \rceil$. According to the above theoretical analysis, we can find the minimum total cost by line 15 to 29 in Algorithm 1. Therefore, the solution composed by TA_1 algorithm and the coding scheme proposed in Sec. IV-B is the optimal solution of MCSCEC problem. □

Theorem 5. *The solution composed by TA_2 algorithm and secure coding scheme proposed in Sec. IV-B is the optimal solution of the MCSCEC problem.*

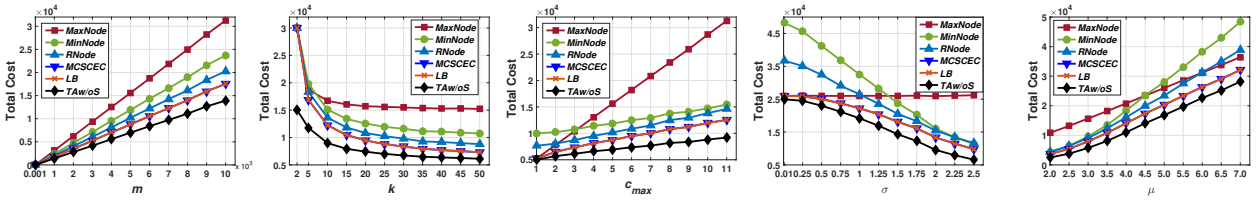
Proof. According to Lemma 2, there exists an optimal solution $\phi^*(\mathbf{S}, k, m, r)$, in which $V(\mathbf{B}_j) = r, \forall 0 < j < \lceil \frac{m+r}{r} \rceil$, $V(\mathbf{B}_{\lceil \frac{m+r}{r} \rceil}) = m - (\lceil \frac{m+r}{r} \rceil - 2)r$ and $V(\mathbf{B}_j) = 0, \forall j > \lceil \frac{m+r}{r} \rceil$. Given the value of r and $i = \lceil \frac{m+r}{r} \rceil$, the total cost of it is $c = r \sum_{j=1}^{i-1} c_j + (m - (i-2)r)c_i$, which can be achieved by the coding design shown in Sec. IV-B. From Theorem 2, we know the value range of r in the optimal solution $\phi^*(\mathbf{S}, k, m, r)$ is $\lceil \frac{m}{k-1} \rceil \leq r \leq m$. Therefore, the minimum value of total cost c can be obtained by exhaustively selecting the values of r in its range and computing the total cost. Therefore, the solution composed by TA_2 algorithm and coding scheme proposed in Sec. IV-B is optimal. □

Theorem 4 and Theorem 5 show that both the TA_1 algorithm and the TA_2 algorithm can derive the optimal values of i and r.

As shown in Sec. IV-A, the computational complexity of TA_1 algorithm is $O(k)$, where k is the number of edge devices. On the other hand, the computational complexity of TA_2 algorithm is $O(k + m)$, where m is the number of rows in data matrix $\mathbf{A}$. In practice, the cloud can select the task allocation algorithm with lower computational complexity according to k and m. For the user device, as shown in Sec. IV-B, to decode and obtain the final result, the user device only needs to perform subtraction operation m times on the received $m + r$ intermediate values. Therefore, the proposed task allocation and secure linear coding design have low computational complexity and decoding complexity.

V. Numerical Experiments

In this section, we conduct simulation to evaluate the performance of the proposed solution for the *MCSCEC* problem. In particular, we compare the performance of the proposed



a) $k = 25, c_{max} = 5$ b) $m = 5000, c_{max} = 5$ c) $k = 25, m = 5000$ d) $k = 25, m = 5000, \mu = 5$ e) $k = 25, m = 5000, \sigma = 1.25$

Fig. 2. Total costs when changing different parameters: m, k, c_{max}, σ and μ.

solution for the *MCSCEC* problem with the lower bound shown in Theorem 1 and the following baseline algorithms.

- *Task Allocation without Security consideration (TAw/oS)* algorithm where a total of m row vectors of data matrix $\mathbf{A}$ are equally allocated on the i^* edge devices without security consideration.
- *MaxNode* algorithm where we set $r = \lceil \frac{m}{k-1} \rceil$, which is the smallest value of r as shown in Theorem 2, and $i = \lceil \frac{m+r}{r} \rceil$. In this case, the maximum number of edge devices are selected.
- *MinNode* algorithm where we let $r = m$, which is the largest value of r as shown in Theorem 2 and $i = 2$. In this case, only two edge devices with the lowest unit cost are selected.
- *Random Node selection (RNode)* algorithm where the value of r is randomly selected from its range $\lceil \frac{m}{k-1} \rceil \leq r \leq m$ and $i = \lceil \frac{m+r}{r} \rceil$.

Since the total costs achieved by both the TA_1 algorithm and the TA_2 algorithm are the same, we denote the optimal scheme composed by the two algorithms and coding scheme as the *MCSCEC* algorithm. In the simulation, we consider the following five parameters: (1) m which is the number of row vectors in data matrix $\mathbf{A}$; (2) k which is the number of edge devices; (3) c_{max} where we consider that $\mathbf{C}$ (the set of unit costs of edge devices) obeys a uniform distribution $\mathcal{U}(1, c_{max})$; (4) μ and (5) σ where we assume that $\mathbf{C}$ follows a normal distribution $\mathcal{N}(\mu, \sigma^2)$. The default values of these parameters are: $m = 5000$, $k = 25$, $c_{max} = 5$, $\mu = 5$ and $\sigma = 1.25$. For each combination of parameters, we generate 1000 instances and report the average results.

In Fig. 2 (a)-(e), it shows that *MCSCEC* always outperforms the *MaxNode*, *MinNode* and *RNode* algorithms. Specifically, in Fig. 2 (a)-(c), it shows that, compared with these three algorithms, the *MCSCEC* algorithm can reduce the total cost by more than 43%, 18%, and 13%, respectively, when m, k and c_{max} are sufficiently large. In Fig. 2 (b), although the larger the number of edge devices, the total cost will be reduced, but in practice multiple edge devices participate in the calculation, which will bring additional communication costs and communication delays, especially in dynamic networks. In Fig. 2 (d), when σ is 0.01, the unit costs of all the edge devices are almost the same. In this case, the more that edge devices are utilized, the lower the total cost is achieved. Therefore, in this case, the total cost of *MaxNode* is almost the same as the minimum total cost achieved by *MCSCEC*. On the other hand, when σ is 2.5, a lower total cost can be achieved when selecting a small number of edge devices with the lowest unit costs. Therefore, the total cost of *MinNode* can lead to the minimum total cost achieved by *MCSCEC*. We can see in the figure that the lines of *MaxNode* and *MinNode* have a cross. Specifically, to the left of the cross, *MaxNode* outperforms *MinNode* and to the right of the cross, *MinNode* outperforms *MaxNode*. In Fig. 2 (e), when μ increases and σ is fixed, the relative difference of costs between different edge devices becomes smaller, which has the same effect as the case that μ is fixed and σ decreases.

In Fig. 2 (a)-(e), it also shows that the performance of *MCSCEC* is very close to the *LB*, and the relative difference between the total cost of *MCSCEC* and *LB* is less than 0.5% when all the paramters are sufficiently large. In this case, to provide security, random vectors should be involved in the computation task. Although the cost of *MCSCEC* is larger than *TAw/oS*, the cost only increases less than 26%, 19% and 14%, respectively, even when m, k and μ are sufficiently large. When c_{max} and σ increase, the relative differences of costs between different edge devices become larger. To reduce the total cost, smaller number of edge devices will be selected in *MCSCEC*. Therefore, more random vectors should be utilized, which leads to the increase in the relative differences of costs between *MCSCEC* and *TAw/oS*. The ratio is no more than 36% and 48%, respectively, even when c_{max} and σ become sufficiently large.

VI. Conclusion

In this paper, we address the design of secure coded distributed computing in edge computing, with the objective to *minimize the total resource usage*. For this fundamental issue, we theoretically analyze the necessary conditions and the lower bound of the problem. Based on the theoretical analysis, we develop optimal algorithms for task allocation which is to select a set of edge devices for computing and assign a certain number of coded row vectors of the matrix to each of them. We then design an efficient secure coded computing scheme to achieve information theoretical security with minimal cost and low decoding complexity. Finally, we conduct extensive simulation experiments, which demonstrate the effectiveness of the proposed schemes. We will consider implement the proposed MCSCEC scheme in real edge computing systems and study a more general case that more than one edge devices can attack cooperatively.

REFERENCES

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[2] S. Li, Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Coded distributed computing: Fundamental limits and practical challenges," in *Proc. of the 50th Asilomar Conference on Signals, Systems and Computers*. IEEE, nov 2016, pp. 509–513.

[3] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding Up Distributed Machine Learning Using Codes," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1514–1529, mar 2018.

[4] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, "A Fundamental Tradeoff Between Computation and Communication in Distributed Computing," *IEEE Transactions on Information Theory*, vol. 64, no. 1, pp. 109–128, jan 2018.

[5] Q. Yu, S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "How to optimally allocate resources for coded distributed computing?" in *Proc. of IEEE International Conference on Communications (ICC)*. IEEE, may 2017, pp. 1–7.

[6] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "Coding for Distributed Fog Computing," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 34–40, apr 2017.

[7] ——, "Communication-aware computing for edge processing," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*, jun 2017, pp. 2885–2889.

[8] R. Bitar, P. Parag, and S. E. Rouayheb, "Minimizing latency for secure distributed computing," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*. IEEE, jun 2017, pp. 2900–2904.

[9] ——, "Minimizing Latency for Secure Coded Computing Using Secret Sharing via Staircase Codes," arXiv, Tech. Rep., feb 2018. [Online]. Available: https://arxiv.org/abs/1802.02640

[10] H. Yang and J. Lee, "Secure Distributed Computing With Straggling Servers Using Polynomial Codes," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 1, pp. 141–150, jan 2019.

[11] S. Dutta, V. Cadambe, and P. Grover, ""Short-Dot": Computing Large Linear Transforms Distributedly Using Coded Short Dot Products," arXiv, Tech. Rep., apr 2017. [Online]. Available: https://arxiv.org/abs/1704.05181

[12] K. Lee, C. Suh, and K. Ramchandran, "High-dimensional coded matrix multiplication," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*. IEEE, jun 2017, pp. 2418–2422.

[13] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Polynomial Codes: an Optimal Design for High-Dimensional Coded Matrix Multiplication," arXiv, Tech. Rep., may 2017. [Online]. Available: https://arxiv.org/abs/1705.10464

[14] R. Tandon, L. Qi, A. Dimakis, and N. Karampatziakis, "Gradient Codings," arXiv, Tech. Rep., dec 2016. [Online]. Available: https://arxiv.org/abs/1612.03301

[15] C. Wang, K. Ren, and J. Wang, "Secure Optimization Computation Outsourcing in Cloud Computing: A Case Study of Linear Programming," *IEEE Transactions on Computers*, vol. 65, no. 1, pp. 216–229, jan 2016.

[16] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers," in *Proc. Annual Cryptology Conference*. Springer, Berlin, Heidelberg, 2010, pp. 465–482.

[17] P. Mohassel, "Efficient and Secure Delegation of Linear Algebra," Cryptology ePrint Archive, Report 2011/605, Tech. Rep., 2011. [Online]. Available: https://eprint.iacr.org/2011/605

[18] B. Zvika and V. Vinod, "Efficient Fully Homomorphic Encryption from (Standard) LWE," in *Proc. of IEEE Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, oct 2011, pp. 97–106.

[19] S. Halevi and V. Shoup, "Faster homomorphic linear transformations in HElib," Cryptology ePrint Archive, Report 2018/244, Tech. Rep., 2018.

[20] Ning Cai and T. Chan, "Theory of Secure Network Coding," *Proceedings of the IEEE*, vol. 99, no. 3, pp. 421–437, mar 2011.