Coded Computation Against Straggling Decoders for Network Function Virtualization

Malihe Aliasgari*, **Jörg Kliewer** *, and **Osvaldo Simeone**†
* Helen and John C. Hartmann Department of Electrical and Computer Engineering
New Jersey Institute of Technology, Newark, New Jersey 07102-1982
{ma839, jkliewer}@njit.edu
† Department of Informatics, King's College London, London, UK
osvaldo.simeone@kcl.ac.uk

***Abstract*—The uplink of a cloud radio access network architecture is studied in which decoding at the cloud takes place via network function virtualization (NFV) on commercial off-the-shelf (COTS) servers. In order to mitigate the impact of straggling decoders in the cloud computing platform, a novel coding strategy is proposed, whereby the cloud re-encodes the received frames via a linear code before distributing them to the decoding processors. Upper bounds on the resulting frame unavailability probability (FUP) as a function of the decoding latency are derived by assuming a binary symmetric channel for uplink communications. The bounds leverage large deviation results for correlated variables, and depend on the properties of both the uplink linear channel code adopted at the user and the NFV linear code applied at the cloud. Numerical examples demonstrate that the bounds are useful tools for code design, and that coding is instrumental in obtaining a desirable trade-off between FUP and decoding latency.**

***Index Terms*—Coded computation, network function virtualization, C-RAN, large deviation.**

I. Introduction

Promoted by the European Telecommunications Standards Institute (ETSI), network function virtualization (NFV) has become a cornerstone of the envisaged architecture of 5G systems [1]. NFV leverages virtualization technologies in order to implement network functionalities on commercial off-the-shelf (COTS) programmable hardware, such as general purpose servers, potentially reducing both capital and operating costs. An important challenge in the deployment of NFV is ensuring carrier grade performance while relying on COTS components. Such components may be subject to temporary unavailability due to malfunctioning, and are generally characterized by randomness in their execution runtimes. The typical solution to these problems involves replicating the virtual machines (VMs) that execute given network functions on multiple processors, e.g., cores or servers [1].

The problem of straggling processors, that is, of processors lagging behind in the execution of a certain orchestrated function, has been well studied in the context of distributed computing. Recently, it has been pointed out that, for the important case of linear functions, it is possible to improve over repetition strategies in terms of the trade-off between performance and latency by carrying out linear precoding of the data prior to processing, see, e.g., [2], [3]. The key idea is that, by employing suitable linear (erasure) block codes operating over fractions of size $1/K$ of the original data, a function may be completed as soon as a number of K or more processors, depending on the minimum distance of the code, has finalized its operation, irrespective of their identity.

In this paper we consider the function of decoding in the uplink of a cloud radio access network (C-RAN). As shown in Fig. 1, each Remote Radio Head (RRH) of a C-RAN architecture is connected to a cloud processor by means of a fronthaul (FH) link. Baseband functionalities are carried out on a distributed computing platform at the cloud, which can be conveniently programmed and reconfigured using NFV. The most computationally demanding baseband function to be carried out at the cloud is uplink channel decoding [4]. Keeping the decoding latency to a minimum is a major challenge in the implementation of C-RAN owing to timing constraints from the MAC layer retransmission protocol. Reference [4] argued that exploiting parallelism across multiple cores in the cloud can reduce the overall decoding latency. However, parallel processing alone does not address the discussed unreliability of COTS hardware. In [5], it was hence proposed to perform linear precoding of the received frames at the cloud in order to mitigate the impact of unreliable decoding servers.

In this paper we extend the treatment of linear coding against unreliable processors in the C-RAN uplink in the following ways. First, while reference [5] considered only a toy example with three processors, here we generalize the approach to any number of processors. Second, unlike the simple binary availability model of [5], in which a processor is either on or off, here we adopt a set-up in which the computing runtime of each processor is random and generally dependent on the computational load as in, e.g., [2], [3]. Third, while [5] relied solely on numerical results, here we develop two analytical upper bounds on the FUP as a function of the decoding delay. The employed model is significantly more general than the toy example in [5]. The bounds leverage large deviation results for correlated variables [6], and depend on the properties of both the uplink linear channel code adopted at the user and the NFV linear code applied at the cloud. Further, as a

This work was supported in part by U.S. NSF grants CNS-1526547 and CCF-1525629. O. Simeone has also received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No 725731).

byproduct of the analysis, we introduce the dependency graph of a linear code and its chromatic number as novel relevant parameters of a linear code beside the minimum distance, blocklength, and rate.

The rest of the paper is organized as follows. In Sec. II, we present the system model focusing, as in [5], on a binary symmetric channel (BSC) for uplink communications. Sec. III presents the two proposed upper bounds on the Frame Unavailability Probability (FUP), and Sec. IV provides numerical results.

II. System Model

As illustrated in Fig. 1, we consider the uplink of a C-RAN system in which a user communicates with the cloud via a remote radio head (RRH). The user is connected to the RRH via a BSC with bit flipping probability δ, while the RRH-to-cloud link, typically referred to as fronthaul, is assumed to be noiseless. Note that the BSC is a simple model for the uplink channel, while the noiseless fronthaul accounts for a typical deployment with higher capacity fiber optic cables. The cloud contains a master server, or Server 0, and N slave servers, i.e., servers $1, \dots, N$. The slave servers are characterized by random computing delays as in related works on coded computation [2], [3], [7]. Note that we use here the term "server" to refer to a decoding processor, although, in a practical implementation, this may correspond to a core of the cloud computing platform [4].

The user encodes a file $\mathbf{u}$ consisting L bits. Before encoding, the file is divided into K blocks $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_K \in \{0,1\}^{L/K}$ of equal size, each of them containing L/K bits. As shown in Fig. 1, in order to combat noise on the BSC, the transmitted frames are encoded by an (n,k) binary linear code $\mathcal{C}_u$ of rate $r = k/n$ defined by generator matrix $\mathbf{G}_u \in \mathbb{F}_2^{n\times k}$, where $n = L/(rK)$ and $k = L/K$. Let $\mathbf{x}_j \in \{0,1\}^n$ with $j \in \{1, \dots, K\}$ be the K transmitted packets of length n. At the output of the BSC, the length-n received vector for the jth packet at the RRH is given as

$$\mathbf{y}_j = \mathbf{x}_j \oplus \mathbf{z}_j, \tag{1}$$

where $\mathbf{z}_j$ is a vector of i.i.d. $\mathrm{Bern}(\delta)$ random variables (rvs). The K received packets $(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_K)$ by the RRH are transmitted to the cloud via the fronthaul link, and the cloud performs decoding. Specifically, as detailed next, we assume that each server $1, \dots, N$ performs decoding of a single packet of length n bits while server 0 acts as coordinator.

Assuming the overprovisioning of servers, which entails the condition $N \geq K$, we adopt the idea of NFV coding proposed in [5]. Accordingly, as seen in Fig. 2, the K packets are first linearly encoded by Server 0 into $N \geq K$ coded blocks of the same length n bits, each forwarded to a different server for decoding. This form of encoding is meant to mitigate the effect of straggling servers in a manner similar to [2], [3], [7]. Using an (N,K) binary linear NFV code $\mathcal{C}_c$ with $K \times N$ generator matrix $\mathbf{G}_c \in \mathbb{F}_2^{N\times K}$, the encoded packets are obtained as

$$\tilde{\mathbf{Y}} = \mathbf{Y}\mathbf{G}_c, \tag{2}$$

where $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_K]$ is the $n \times K$ matrix obtained by including the received signal $\mathbf{y}_j$ as the jth column and $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_1, \dots, \tilde{\mathbf{y}}_N]$ is the $n \times N$ matrix whose ith column $\tilde{\mathbf{y}}_i$ is the input to server i, where $i \in \{1, \dots, N\}$. From (1), this vector can be written as

$$\tilde{\mathbf{y}}_i = \sum_{j=1}^{K} \mathbf{y}_j g_{c,ji} = \sum_{j=1}^{K} \mathbf{x}_j g_{c,ji} + \sum_{j=1}^{K} \mathbf{z}_j g_{c,ji}, \tag{3}$$

where $g_{c,ij}$ is the (i,j)th entry of matrix $\mathbf{G}_c$.

The signal part $\sum_{j=1}^{K} \mathbf{x}_j g_{c,ji}$ in (3) is a linear combination of d_i codewords for the rate-r binary code with generator matrix $\mathbf{G}_u$, and hence it is a codeword of the same code. The parameter d_i, $i \in \{1, \dots, N\}$, denotes the Hamming weight of the ith column of matrix $\mathbf{G}_c$, where $0 \leq d_i \leq K$. Each server i receives as input the packets $\tilde{\mathbf{y}}_i$ from which it can decode the codeword $\sum_{i=1}^{K} \mathbf{x}_i g_{c,ji}$. This decoding operation is affected by the noise vector $\sum_{j=1}^{K} \mathbf{z}_j g_{ji}$ in (3), which has i.i.d. $\mathrm{Bern}(\gamma_i)$ elements. Here, γ_i is obtained as the first row and second column's entry of the matrix $\mathbf{Q}^{d_i}$, with $\mathbf{Q}$ being the channel matrix $\mathbf{Q} = (1-2\delta)\ \mathbf{I} + \delta \mathbf{1}\mathbf{1}^T$ ($\mathbf{I}$ is the identity matrix and $\mathbf{1}$ the all-one column vector). Note that a larger value of d_i yields a larger bit flipping probability γ_i. We define as $\mathrm{P}_{n,k}(\gamma_i)$ the decoding error probability of Server i.

Server i requires a random time $T_i = T_{1,i} + T_{2,i}$ to complete decoding, which is modeled as the sum of a component $T_{1,i}$ that is independent of the workload and a component $T_{2,i}$ that instead grows with the size n of the packet processed at each server, respectively. The first component accounts, e.g., for processor unavailability periods, while the second models the execution runtime from the start of the computation. The first variable $T_{1,i}$ is assumed to have an exponential probability density function (pdf) $f_1(t)$ with mean $1/\mu_1$, while the variable $T_{2,i}$ has a shifted exponential distribution with cumulative distribution function (cdf) [8]

$$F_2(t) = 1 - \exp\left(-\frac{rK\mu_2}{L}\left(t - a\frac{L}{rK}\right)\right), \tag{4}$$

for $t \geq aL/(rK)$ and $F_2(t) = 0$ otherwise. The parameter a represents the minimum processing time per input bit, while $1/\mu_2$ is the average additional time needed to process one bit. The cdf of the time T_i can hence be written as the integral $F(t) = \int_0^t f_1(\tau)F_2(t-\tau)d\tau$. We also assume that the runtime rvs $\{T_i\}_{i=1}^N$ are mutually independent. Due to (4), the probability that a given set of l out of N servers has finished decoding by time t is given as

$$a_l(t) = F(t)^l (1 - F(t))^{N-l}. \tag{5}$$

Let $d_{\min}$ be the minimum distance of the NFV code $\mathcal{C}_c$. Due to (3), Server 0 in the cloud is able to decode the message $\mathbf{u}$ or equivalently the K packets $\mathbf{u}_j$ for $j \in \{1, \dots, K\}$, as soon as $N - d_{\min} + 1$ servers have decoded successfully. Let $\hat{\mathbf{u}}_i$ be the output of the ith server in the cloud upon decoding. The output $\hat{\mathbf{u}}$ of the decoder at Server 0 at time t is then a function of the vectors $\hat{\mathbf{u}}_i(t)$ for $i \in \{1, \dots, N\}$, where

$$\hat{\mathbf{u}}_i(t) = \begin{cases} \hat{\mathbf{u}}_i, & \text{if } T_i \leq t, \\ \emptyset, & \text{otherwise.} \end{cases}$$

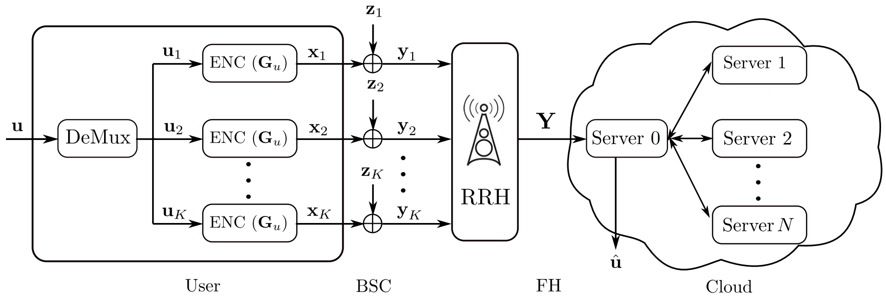

Fig. 1: NFV model for uplink channel decoding. The input file $\mathbf{u}$ is divided into smaller packets and encoded with a linear code $\mathcal{C}_u$ having generator matrix $\mathbf{G}_u$. The packets are received by the RRH through a BSC and forwarded to a cloud. Server 0 in the cloud re-encodes the received packet with a linear code $\mathcal{C}_c$ in order to enhance robustness against the potentially straggling servers $1, \ldots, N$.

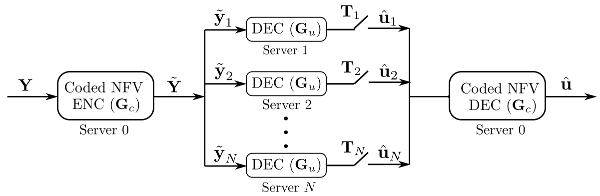

Fig. 2: Coded NFV at the cloud: Server 0 re-encodes the received packets in $\mathbf{Y}$ by a linear NFV code $\mathcal{C}_c$ with generator $\mathbf{G}_c$. Each encoded packet $\tilde{\mathbf{y}}_i$ then conveyed to server i for decoding.

Finally, the frame unavailability probability (FUP) at time t is defined as the probability

$$\mathrm{P}_u(t) = \Pr\left[\hat{\mathbf{u}}(t) \neq \mathbf{u}\right]. \tag{6}$$

III. Large Deviation Bound on the Frame Unavailability Probability

In this section we derive analytical bounds on the error probability $\mathrm{P}_u(t)$ in (6).

A. Preliminaries

Each Server i with $i \in \{1, \ldots, N\}$ decodes successfully its assigned packet $\tilde{\mathbf{y}}_i$ if: (*i*) the server completes decoding by time t; (*ii*) the decoder is able to correct the errors caused by the BSC. Furthermore as discussed, an error occurs at time t if the number of servers that have successfully decoded by time t is smaller than $N - d_{\min} + 1$. To evaluate the FUP, we hence define the indicator variables $C_i(t) = \mathbb{1}\{T_i \leq t\}$ and $D_i = \mathbb{1}\{\hat{\mathbf{u}}_i = \mathbf{u}_i\}$, which are equal to 1 if the events (*i*) and (*ii*) described above occur, respectively, and zero otherwise. Based on these definitions, the FUP is equal to

$$\mathrm{P}_u(t) \quad = \quad \Pr\left[\sum_{i=1}^{N} C_i(t) D_i \leq N - d_{\min}\right]. \tag{7}$$

The indicator variables $C_i(t)$ are independent Bernoulli rvs across the servers $i \in \{1, \ldots, N\}$, due to the independence assumption on the rvs T_i. However, the indicator variable D_i are dependent Bernoulli rvs. The dependence of the variables D_i is caused by the fact that the noise terms $\sum_{i=1}^{K} \mathbf{z}_j g_{c,ji}$ in (3) generally have common terms. In particular, if two columns i and j of the generator matrix $\mathbf{G}_c$ have at least a 1 in the same row, then the decoding indicators D_i and D_j are correlated. This complicates the evaluation of bounds on the FUP (7).

B. Dependency Graph and Chromatic Number of a Linear Code

To capture the correlation among the indicator variables D_i, we introduce here the notion of the *dependency graph* and its chromatic number for a linear code. These appear to be novel properties of a linear code, which will be argued below to determine a code's performance for the application at hand.

Definition 1. *Let $\boldsymbol{G} \in \mathbb{F}_2^{K' \times N'}$ be a generator matrix of a linear code. The dependency graph $\mathcal{G}(\boldsymbol{G}) = (\mathcal{V}, \mathcal{E})$ comprises a set $\mathcal{V}$ of N' vertices and a set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ of edges, where edge $(i, j) \in \mathcal{E}$ is included if both the ith and jth columns of $\boldsymbol{G}$ have at least a 1 in the same row.*

The chromatic number $\mathcal{X}(\mathbf{G})$ of the graph $\mathcal{G}(\mathbf{G})$ will play an important role in the analysis. We recall that the chromatic number is the smallest number of colors needed to color the vertices of $\mathcal{G}(\mathbf{G})$, such that no two adjacent vertices share the same color. Generally, finding the chromatic number of a graph is NP-hard [9]. However, a simple upper bound on $\mathcal{X}(\mathbf{G})$ is given as [10], $\mathcal{X}(\mathbf{G}) \leq \Delta(\mathbf{G}) + 1$, where $\Delta(\mathbf{G})$ is the maximum degree of a graph $\mathcal{G}(\mathbf{G})$.

C. Large Deviation Upper Bound

In this subsection, we provide a first upper bound on the FUP. The bound is based on the large deviation result in [6] for the tail probabilities of rvs $X = \sum_{i=1}^{M} X_i$, where the rvs X_i are generally dependent. The correlation of rvs $\{X_i\}$ is described in [6] by a dependency graph. This is defined as any graph $\mathcal{G}(X)$ with X_i as vertices, such that, if a vertex $i \in \{1, \ldots, M\}$ is not connected to any vertex in a subset $\mathcal{J} \subset \{1, \ldots, M\}$, then X_i is independent of $\{X_j\}_{j \in \mathcal{J}}$.

Lemma 1 ([6])**.** *Let $X = \sum_{i=1}^{M} X_i$, where $X_i \sim$ Bern(p_i) and $p_i \in (0, 1)$ are generally dependent. For any $b \geq 0$, such that the inequality $X_i - \mathbb{E}(X_i) \geq -b$ holds for all $i \in \{1, \ldots, M\}$ with probability one, and for any $\tau \geq 0$ we have*

$$\Pr[X \leq \mathbb{E}(X) - \tau] \leq \exp\left(-\frac{S}{b^2 \mathcal{X}(\mathcal{G}(X))} \varphi\left(\frac{4b\tau}{5S}\right)\right), \tag{10}$$

where $S \triangleq \sum_{i=1}^{N}$ Var(X_i) and $\varphi(x) \triangleq (1+x)\ln(1+x) - x$. The same bound (10) *holds for* $\Pr(X \geq \mathbb{E}(X) + \tau)$*, where $X_i - \mathbb{E}(X_i) \leq b$ with probability one.*

The following theorem uses Lemma 1 to derive a bound on the FUP.

Theorem 1. *Let* $\mathrm{P}_{n,k}^{\min} = \min_i\{\mathrm{P}_{n,k}(\gamma_i)\}_{i=1}^N$. *For all*

$$t \geq F^{-1}\left(\frac{N-d_{\min}}{N-\sum_{i=1}^N \mathrm{P}_{n,k}(\gamma_i)}\right), \tag{11}$$

the FUP is upper bounded as in (8)*, shown at the bottom of the page, where* $b(t) \triangleq F(t)\left(1-\mathrm{P}_{n,k}^{\min}\right)$ *and* $S(t) \triangleq \sum_{i=1}^N F(t)\left(1-\mathrm{P}_{n,k}(\gamma_i)\right)\left(1-F(t)(1-\mathrm{P}_{n,k}(\gamma_i))\right)$.

Proof. Let $X_i(t) \triangleq C_i(t)D_i$ and $X(t) = \sum_{i=1}^N X_i(t)$, where $X_i(t)$ are dependent Bernoulli rvs with probability $\mathbb{E}[X_i(t)] = \Pr[X_i(t)=1] = F(t)\left(1-\mathrm{P}_{n,k}(\gamma_i)\right)$. It can be seen that a valid dependency graph $\mathcal{G}(X)$ for the variables $\{X_i\}$ is the dependency graph $\mathcal{G}(\mathbf{G}_c)$ defined above. This is due to the fact, discussed in Section III-C, that the rvs X_i and X_j are dependent if and only if the ith and jth column of $\mathbf{G}_c$ have at least a 1 in a common row. We can hence apply Lemma 1 for every time t by selecting $\tau = \mathbb{E}(X) - N + d_{\min}$, and $b(t)$ as defined above. Note that this choice of $b(t)$ meets the constraint for b in Lemma 1. For $1/\mu_1 = 0$, (11) simplified as follows

$$t \geq n\left(a - \frac{1}{\mu}\ln\left(\frac{d_{\min}-\sum_{i=1}^N \mathrm{P}_{n,k}(\gamma_i)}{N-\sum_{i=1}^N \mathrm{P}_{n,k}(\gamma_i)}\right)\right). \tag{12}$$

□

The upper bound (8), on the FUP captures the dependency of the FUP on both the channel code and the NFV code. In particular, the bound is an increasing function of the error probabilities $\mathrm{P}_{n,k}(\gamma_i)$, which depend on both codes. It also depends on the NFV code through parameters $d_{\min}$ and $\mathcal{X}(\mathbf{G}_c)$.

D. Union Bound

As indicated in Theorem 1, the large deviation based bound in (8) is only valid for large enough t, as can be observed from (12). Furthermore, it may generally not be tight, since it neglects the independence of the indicator variables C_i. In this subsection, a generally tighter but more complex bound is derived that is valid for all times t.

Theorem 2. *For any subset* $\mathcal{A} \subseteq \{1,\ldots,N\}$*, define*

$$\mathrm{P}_{n,k}^{\min(\mathcal{A})} \triangleq \min\{\mathrm{P}_{n,k}(\gamma_i)\}_{i\in\mathcal{A}} \quad \text{and} \quad \mathrm{P}_{n,k}^{\mathcal{A}} \triangleq \sum_{i\in\mathcal{A}} \mathrm{P}_{n,k}(\gamma_i),$$

and let $\mathbf{G}_{\mathcal{A}}$ *be the* $K \times |\mathcal{A}|$*, submatrix of* $\mathbf{G}_c$*, with column indices in the subset* $\mathcal{A}$*. Then, the FUP is upper bounded by* (9)*, shown at the bottom of the page, where* $S_{\mathcal{A}} \triangleq \sum_{i\in\mathcal{A}} \mathrm{P}_{n,k}(\gamma_i)\left(1-\mathrm{P}_{n,k}(\gamma_i)\right)$ *and* $b_{\mathcal{A}} \triangleq 1-\mathrm{P}_{n,k}^{\min(\mathcal{A})}$.

Proof. Let $I_i = 1 - D_i$ be the indicator variable which equals 1 if Server i fails decoding. Accordingly, we have $I_i \sim \mathrm{Bern}(\mathrm{P}_{n,k}(\gamma_i))$. For each subset $\mathcal{A} \subseteq \{1,\ldots,N\}$, let $I_{\mathcal{A}} = \sum_{i\in\mathcal{A}} I_i$. The complement of the FUP $\mathrm{P}_s(t) = 1 - \mathrm{P}_u(t)$ can hence be written as

$$\mathrm{P}_s(t) = \Pr\left[\sum_{i=1}^N C_i(t)D_i > N - d_{\min}\right] \tag{13}$$

$$= \sum_{l=N-d_{\min}+1}^{N} a_l(t) \sum_{\substack{\mathcal{A}\subseteq\{1,\ldots,N\}:\\ |\mathcal{A}|=l}} \left(1 - \Pr\left[I_{\mathcal{A}} \geq l - N + d_{min}\right]\right). \tag{14}$$

We can now apply Lemma 1 to the probability in (14) by noting that $\mathcal{G}(\mathbf{G}_{\mathcal{A}})$ is a valid dependency graph for the variables $\{I_i\}$, $i \in \mathcal{A}$. In particular, we apply Lemma 1 by setting $\tau_{\mathcal{A}} = l - N + d_{\min} - \mathbb{E}(I_{\mathcal{A}})$, $b_{\mathcal{A}} \geq I_i - \mathbb{E}[I_i]$, and $S_{\mathcal{A}} = \sum_{i\in\mathcal{A}} \mathrm{Var}\,(I_i)$, leading to

$$\Pr\left[I_{\mathcal{A}} \geq l - N + d_{\min}\right] \leq \exp\left(-\frac{S_{\mathcal{A}}}{b_{\mathcal{A}}^2 \mathcal{X}(\mathbf{G}_{\mathcal{A}})}\varphi\left(\frac{4b_{\mathcal{A}}\left(l-N+d_{\min}-\mathrm{P}_{n,k}^{\mathcal{A}}\right)}{5S_{\mathcal{A}}}\right)\right). \tag{15}$$

By substituting (15) into (14), the proof is completed. □

IV. Simulation Results

In this section we provide some numerical result to validate the two bounds presented in Theorems 1 and 2, as well as to assess the importance of coding in obtaining desirable trade-offs between decoding latency and FUP. We employ a frame length of $L = 504$ and $N = 8$ servers. The user code $\mathcal{C}_u$ is selected to be a randomly designed $(3,6)$ regular (Gallager-type) LDPC code with $r = 0.5$, decoded via belief propagation.

Fig. 3 and Fig. 4 compare the performance of the following solutions: (*i*) *Standard single-server decoding*, whereby we assume the use of a single server, that is $N = 1$, that decodes the entire frame ($K = 1$); (*ii*) *Repetition coding*, whereby the entire frame ($K = 1$) is replicated at all servers; (*iii*) *Parallel (or uncoded) processing*, whereby the frame is divided into $K = N$ disjoint parts processed by different servers; (*iv*)

$$\mathrm{P}_u(t) \leq \exp\left(-\frac{S(t)}{b^2(t)\mathcal{X}(\mathbf{G}_c)}\varphi\left(\frac{4b(t)\left(NF(t) - F(t)\sum_{i=1}^N \mathrm{P}_{n,k}(\gamma_i) - N + d_{\min}\right)}{5S(t)}\right)\right), \tag{8}$$

$$\mathrm{P}_u(t) \leq 1 - \sum_{l=N-d_{\min}+1}^{N} a_l(t) \sum_{\substack{\mathcal{A}\subseteq\{1,\ldots,N\}:\\ |\mathcal{A}|=l}} \left(1 - \exp\left(-\frac{S_{\mathcal{A}}}{b_{\mathcal{A}}^2 \mathcal{X}(\mathbf{G}_{\mathcal{A}})}\varphi\left(\frac{4b_{\mathcal{A}}\left(l-N+d_{\min}-\mathrm{P}_{n,k}^{\mathcal{A}}\right)}{5S_{\mathcal{A}}}\right)\right)\right). \tag{9}$$

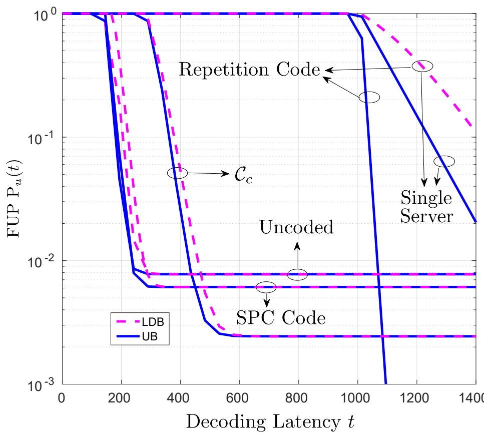

Fig. 3: Large deviation (LDB) bound of Theorem 1 and union bound (UB) in Theorem 2 for single-server decoding, repetition coding, uncoded approach, SPC code and the NFV code $\mathcal{C}_c$ defined in the text ($L = 504, N = 8, 1/\mu_1 = 0, \mu_2 = 10, a = 1, \delta = 0.01, r = 0.5$).

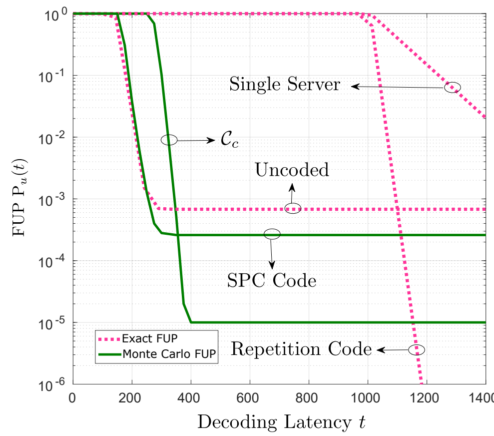

Fig. 4: Exact FUP for single-server decoding, repetition coding, uncoded approach and Monte Carlo simulation results for the SPC code and the NFV code $\mathcal{C}_c$ defined in the text ($L = 504, N = 8, 1/\mu_1 = 0, \mu_2 = 10, a = 1, \delta = 0.01, r = 0.5$).

Single parity check code (SPC), with $K = 7$; (*v*) NFV code $\mathcal{C}_c$ with the following generator matrix

$$\mathbf{G}_C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

Note that, with both single-server decoding and repetition coding, we have the blocklength $n = 1008$ for the channel code. Single-server decoding is trivially characterized by $\mathcal{X}(\mathbf{G}_c) = d_{\min} = 1$, while repetition coding is such that the equalities $\mathcal{X}(\mathbf{G}_c) = d_{\min} = 8$ hold. Furthermore, the uncoded approach is characterized by $n = 126$, $d_{\min} = 1$ and $\mathcal{X}(\mathbf{G}_c) = 1$; the SPC code has $n = 144, d_{\min} = 2$ and $\mathcal{X}(\mathbf{G}_c) = 2$; and the NFV code $\mathcal{C}_c$ has $n = 252$, $d_{\min} = 3$ and $\mathcal{X}(\mathbf{G}_c) = 3$. The exact FUP for a given $\mathrm{P}_{n,k}(\cdot)$ can easily be computed for cases (*i*)-(*iii*) (not reported due to lack of space).

Fig. 3 shows both LDB in Theorem 1 and UB in Theorem 2, respectively, for all five schemes. In this figure, we assume that the latency contribution that is independent of the workload is negligible, i.e., $1/\mu_1 = 0$, and we concentrate on the effect of the load-dependent term by setting $a = 1$ and $\mu_2 = 10$. As a first observation, Fig. 3 confirms that the UB bound is tighter than the LDB. Furthermore, we note the significant gains in terms of the trade-off between latency and FUP that can be obtained by leveraging multiple servers in parallel for decoding, as argued also in [4] using experimental results. With regard to the comparison among different NFV coding schemes, we first observe that the bounds indicate that the uncoded scheme is to be preferred for lower latencies. This is due to the shorter blocklength n, which entails a smaller average decoding time. However, the error floor of the uncoded scheme is large given the higher error probability on the BSC for short blocklengths. In a similar manner, repetition coding requires a larger latency in order to obtain acceptable FUP performance owing to the larger blocklength n, but it achieves a significantly lower error floor. For intermediate latencies, SPC and, at larger latencies, also the NFV code $\mathcal{C}_c$ provide a lower FUP according to the bounds. This demonstrates the effectiveness of NFV encoding in obtaining a desirable trade-off between latency and FUP. Finally, we observe that the relative performance of the coding schemes would be different in the presence of a larger value for the load-independent average delay $1/\mu_1$ (not shown).

Fig. 4 shows the exact FUP for the schemes (*i*)-(*iii*), as well as Monte Carlo simulation results for schemes (*iv*) and (*v*). While the absolute numerical values of the bounds in Fig. 3 are not uniformly tight with respect to the actual performance evaluated by Fig. 4, the relative performance of the coding schemes in the two figures are well matched. This suggests that the derived bounds can serve as a useful tool for code design in NFV systems.

References

[1] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 2016.

[2] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *Proc. IEEE International Symposium on Information Theory*, pp. 1143–1147, 2016.

[3] 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, 2018.

[4] V. Q. Rodriguez and F. Guillemin, "Towards the deployment of a fully centralized cloud-ran architecture," in *Wireless Communications and Mobile Computing Conference (IWCMC), 2017 13th International*, Valencia, Spain, 2017, pp. 1055–1060.

[5] A. Al-Shuwaili, O. Simeone, J. Kliewer, and P. Popovski, "Coded network function virtualization: Fault tolerance via in-network coding," *IEEE Wireless Communications Letters*, vol. 5, no. 6, pp. 644–647, 2016.

[6] S. Janson, "Large deviations for sums of partly dependent random variables," *Random Structures & Algorithms*, vol. 24, no. 3, pp. 234–248, 2004.

[7] A. Severinson, A. G. i Amat, and E. Rosnes, "Block-diagonal coding for distributed computing with straggling servers," in *Information Theory Workshop (ITW)*, 2017, pp. 464–468.

[8] A. Reisizadehmobarakeh, S. Prakash, R. Pedarsani, and S. Avestimehr, "Coded computation over heterogeneous clusters," [Online] www.arxiv.org, arXiv:1701.05973 [cs.IT], 2017.

[9] A. Sánchez-Arroyo, "Determining the total colouring number is NP-hard," *Discrete Mathematics*, vol. 78, no. 3, pp. 315–319, 1989.

[10] R. L. Brooks, "On colouring the nodes of a network," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 37, no. 02, pp. 194–197, 1941.