Risk-Sensitive Task Fetching and Offloading for Vehicular Edge Computing

Sadeep Batewela, Student Member, IEEE, Chen-Feng Liu, Student Member, IEEE, Mehdi Bennis, Senior Member, IEEE, Himal A. Suraweera, Senior Member, IEEE,  
and Choong Seon Hong, Senior Member, IEEE

Abstract—This letter studies an ultra-reliable low latency communication problem focusing on a vehicular edge computing network in which vehicles either fetch and synthesize images recorded by surveillance cameras or acquire the synthesized image from an edge computing server. The notion of risk-sensitive in financial mathematics is leveraged to define a reliability measure, and the studied problem is formulated as a risk minimization problem for each vehicle’s end-to-end (E2E) task fetching and offloading delays. Specifically, by resorting to a joint utility and policy estimation-based learning algorithm, a distributed risk-sensitive solution for task fetching and offloading is proposed. Simulation results show that our proposed solution achieves performance improvements up to 40% variance reduction and steeper distribution tail of the E2E delay over an averaged-based baseline.

Index Terms—5G and beyond, vehicular edge computing, URLLC, risk-sensitive learning.

I. INTRODUCTION

By applying the mobile edge computing (MEC) architecture [1] to vehicular networks, vehicular edge computing (VEC) [2] provides computation service for vehicles to execute time-critical applications. The main problem is that the latency exacerbates when all vehicles intend to access the MEC server’s computational resources. To this end, some works have studied the offloading and resource allocation mechanisms in various VEC scenarios [3]–[5]. However, the majority of the VEC literature aimed at reducing the latency, whereas reliably delivering messages for traffic safety maintenance is critical in VEC networks. To enable ultra-reliable low latency communication (URLLC), improving the delay performance on average is not sufficient [6] but requires further breaking down the statistics of delay, i.e., mean, variance, skewness, etc. To this end, we resort to the risk measure, a notion adopted in financial mathematics [7], for wireless communication, where risk is synonymous with losing valuable information due to the randomness of wireless transmission. For example, the stochastic channel quality incurs delay variation, and a higher variance in the delay can result in an urgent-message loss, making traffic safety at stake.

In this letter, we consider a VEC scenario in which computation tasks are remotely generated at surveillance cameras. Next, the vehicle decides either to fetch the tasks from the cameras for local computation or to obtain the computed results from the MEC server. In the latter case, tasks are offloaded from the cameras and computed at the server. Motivated by the aforementioned shortcomings, the studied problem is cast as a risk minimization problem for each vehicle’s end-to-end (E2E) delay. Moreover, the variance and higher-order statistics of the E2E delay are taken into account in the risk-minimization problem. Since all vehicles’ delays are coupled together in the considered system, each vehicle requires other vehicles’ wireless channel information, which is not available in practice, to solve the problem. To address this issue, we leverage upon a joint utility and policy estimation-based learning framework [8], [9] and propose a distributed risk-sensitive approach for task fetching and offloading. Numerical results show that in contrast to the average-based system design, our proposed solution achieves lower variance and steeper distribution tail of the E2E delay albeit higher average performance. The superiority of the proposed approach in denser networks is validated through comparison with several baseline schemes.

II. SYSTEM MODEL

We consider an urban VEC network around the road intersection, where a set of cameras are installed at the intersection to record images for aiding traffic safety. We assume that each camera has a limited angle of view, and all cameras’ images need to be synthesized to have the full view of the intersection. Since VUEs are at different locations, we simply assume that each VUE requires...
a dedicated synthesized image which is different from the other VUEs’ required synthesized images. The VUE has its own computational capability to synthesize the images that are wirelessly fetched from cameras. Moreover, due to VUEs’ limited computation capabilities, an MEC server, connected with all cameras via fiber, is deployed to provide the computational service for the VUEs. In this regard, images are first delivered to the MEC server through the fiber links. After synthesizing, the MEC server wirelessly sends the synthesized images to the VUEs. Let us further denote each VUE \( i \)’s task-fetching and offloading decision as \( \alpha_i \in \{0, 1\} \), \( \forall i \in V \), in which the VUE fetches images and synthesizes them locally if \( \alpha_i = 0 \). Otherwise, synthesizing is executed at the MEC server.

When \( \alpha_i = 0 \), the VUE first fetches the images from the cameras. We assume that each camera has a dedicated bandwidth \( W_c \) to broadcast its image to all task-fetching VUEs, denoted by \( V_i = \{i \in V|\alpha_i = 0\} \). In order to ensure that all VUEs in \( V_i \) can correctly receive the same image over the full bandwidth \( W_c \), each camera \( j \in C \) broadcasts its data with the rate \( R_j = W_c \log_2 (1 + P_j h_{j,\min} / W_c N_0) \), considering the minimal channel gain among all corresponding camera-VUE links, i.e., \( h_{j,\min} = \min h_{j,i} \). Here, \( h_{j,i} \) is the channel gain, including path loss and channel fading, between camera \( j \) and VUE \( i \). We further assume that all wireless channel gains belong to the finite sets such that channels stay static during task fetching and offloading. \( P_j \) is the camera’s transmit power which is identical for all cameras, and \( N_0 \) is the noise variance. Then given the camera’s image size \( A \), the transmission power delay from camera \( j \) is \( T_{j,\text{Tx}}^i = A/R_j \). Since the VUE \( i \) has to wait till it receives all cameras’ images before synthesizing, the net transmission delay for each VUE is \( T_{\text{Fet}} = \max_{j \in C} T_{j,\text{Tx}}^i \). Here, we note that the transmission delay \( T_{\text{Fet}} \) is affected by all the task-fetching VUEs’ wireless channels as per \( h_{j,\min} \in V_i \). Assuming that all camera’s images are in the same size, we calculate the computation delay for image synthesizing at VUE \( i \) as \( T_{\text{Comp}}^i = CAL/f_i \), in which \( L \) and \( f_i \) (in the unit of cycle/sec) are, respectively, the required central processing unit (CPU) cycles per bit for computation, i.e., the processing density, and VUE \( i \)’s local CPU-cycle frequency [1]. When \( \alpha_i = 0 \), VUE \( i \)’s E2E delay \( T_{\text{E2E}}^i \) includes the transmission delay and local computation delay, i.e., \( T_{\text{E2E}}^i = T_{\text{Fet}} + T_{\text{Comp}}^i \).

For \( \alpha_i = 1 \), let us first analogously denote the set of VUEs with the task-offloading decision as \( V_o = \{i \in V|\alpha_i = 1\} \). Additionally, since all cameras deliver their images to the MEC server through fiber links, we neglect the transmission delays from cameras to the MEC server. After obtaining these images, the MEC server equally divides its total CPU-cycle frequency \( f_s \) into \( |V_o| \) parts in order to simultaneously synthesize the \( |V_o| \) VUEs’ dedicated images. Thus, the computational delay at the MEC server is expressed as \( T_{\text{Comp}}^\text{CAL} = CAL/|V_o|/f_s \). Subsequently, the MEC server orthogonally sends the synthesized image to each VUE \( i \in V_o \) with the equally allocated bandwidth \( W_c/|V_o| \). The allocated transmit power for VUE \( i \) is denoted by \( P_i \) with the constraint

\[
\sum_{i \in V_o} P_i = P_{\text{max}} \text{ and } P_i \geq 0, \forall i \in V_o,
\]

where \( P_{\text{max}} \) is the MEC server’s transmit power budget. The power allocation mechanism will be detailed in Section III-B. Accordingly, the corresponding downlink (DL) rate (from the MEC server) to the VUE \( i \) is expressed as \( R_i = \frac{W_c}{|V_o|} \log_2 \left(1 + \frac{P_i h_{s,i}}{W_c N_0} \right) \), where \( h_{s,i} \) is the DL channel gain from the MEC server to VUE \( i \). Then given the synthesized image size \( B \), the DL transmission delay of the VUE \( i \) is \( T_{\text{DL}}^i = B/R_i \). When \( \alpha_i = 1 \), VUE \( i \)’s E2E delay includes the computation delay at the MEC server and the DL transmission delay, i.e., \( T_{\text{E2E}}^i = T_{\text{Comp}}^i + T_{\text{DL}}^i \). In summary, VUE \( i \)’s E2E latency is expressed as

\[
T_{\text{E2E}}^i = \begin{cases}
\max_{j \in C} \left\{ \frac{A}{W_c \log_2 \left(1 + \frac{P_j h_{j,\min}}{W_c N_0}\right)} + CAL/f_j \right\}, & \text{when } \alpha_i = 0, \\
\frac{CAL |V_o|}{f_s} + \frac{B |V_o|}{W_c \log_2 \left(1 + \frac{P_i h_{s,i}}{W_c N_0}\right)}, & \text{when } \alpha_i = 1.
\end{cases}
\]

Furthermore, we note that the E2E delay in (2) is random.

### III. Problem Formulation

#### A. Risk Minimization for the VUE’s End-to-End Delay

As mentioned in Section I, risk is considered as our concerned reliability metric. In this regard, we focus on the entropic risk measure\(^2\) \( \frac{1}{\rho} \ln \mathbb{E}[\exp(\rho T_{\text{E2E}}^i)] \) and formulate a risk minimization problem for each VUE \( i \in V \) as follows:

\[
\begin{aligned}
& \text{minimize} & & \frac{1}{\rho} \ln \mathbb{E}[\exp(\rho T_{\text{E2E}}^i)] \\
& \text{subject to} & & \sum_{\alpha_i \in \{0, 1\}} \Pr(\alpha_i | h_i) = 1, \forall h_i \in H_i,
\end{aligned}
\]

where \( \mathbb{E}[\cdot] \) is the expectation operator, and \( h_i = [h_{j,i}, h_{s,i} : j \in C] \in H_i \), \( \forall i \in V \), denotes VUE \( i \)’s channel vector with a finite set \( H_i \). By taking the Maclaurin series expansion, we can express the objective function (3a) as

\[
\frac{1}{\rho} \ln \mathbb{E}[\exp(\rho T_{\text{E2E}}^i)] = \mathbb{E}[T_{\text{E2E}}^i] + \frac{\alpha_0}{\rho^2} \text{Var}(T_{\text{E2E}}^i) + \frac{\alpha_1}{\rho^3} \text{Cov}(T_{\text{E2E}}^i, \text{Var}(T_{\text{E2E}}^i)) + \cdots,
\]

in which \( \alpha_0 = \mathbb{E}[T_{\text{E2E}}^i] \) and \( \alpha_1 = \mathbb{E}[(T_{\text{E2E}}^i - \mathbb{E}[T_{\text{E2E}}^i])^3] \) is the third central moment of \( T_{\text{E2E}}^i \), and the skewness of \( T_{\text{E2E}}^i \) is equal to \( \frac{\alpha_1}{\alpha_3} \). In other words, the mean, variance, skewness, and other higher-order statistics of the E2E delay are taken into account in the objective. Moreover, the risk-sensitivity parameter \( \rho > 0 \) reflects the weight of higher-order statistics in the formulated risk minimization problem. Owing to the randomness of \( h_i \) and other unavailable information, e.g., the other VUEs’ channels, each VUE \( i \) focuses on the probabilistic policy \( \Pr(\alpha_i| h_i) \) of the task-fetching and offloading decision in problem (3). Since \( \frac{1}{\rho} \ln(\cdot) \) in (3a) is a monotonically

\(^1\)For example, by quantizing the continuous channel fading values in consecutive coherence time blocks into the same level such that the corresponding transmission rate is sustainable.

\(^2\)Because entropic risk measure directly incorporates the statistics of the delay, i.e., mean, variance, skewness, etc.
increasing function, we can directly remove it and focus on an equivalent problem

\[
\min_{P_t} \mathbb{E}[\exp(\rho T^\text{E2E}_t)] \text{ subject to (3b).} \quad (4)
\]

Analogously, by taking the Maclaurin series expansion, i.e., \( \mathbb{E}[\exp(\rho T^\text{E2E}_t)] = 1 + \rho \mathbb{E}[T^\text{E2E}_t] + \frac{\rho^2}{2} \mathbb{E}[(T^\text{E2E}_t)^2] + \cdots \), we can see that the higher-order moments/statistics are still considered in the objective function of the equivalent problem.

**B. Transmit Power Allocation at the MEC Server**

Referring to the motivation of considering (4), we formulate the MEC server’s power allocation problem as

\[
\min_{P_t} \sum_{i \in V_o} \exp(\rho T^\text{DL}_t) \text{ subject to (5).} \quad (5)
\]

In the objective, we consider the DL transmission delay since the allocated transmit power only affects this delay. Following the results in [10, Eq. (3.10)], we can verify that \( \exp(\rho T^\text{DL}_t) = \exp \left( \frac{W_i \log_2 \left( 1 + \frac{P_{\text{max}}}{W_i + \rho T^\text{DL}_t} \right)}{W_i} \right) \) is a convex function with respect to \( P_t \). Hence, we directly apply the Karush–Kuhn–Tucker (KKT) conditions and derive the following optimal solution. For the DL transmission, the MEC server allocates the transmit power \( P^*_t > 0, \forall i \in V_o \), which satisfies

\[
\theta \kappa_i \exp \left( \frac{\theta \kappa_i}{1 + \theta \kappa_i} \right) = \nu \quad \text{with} \quad \theta = \frac{D^\eta [V_i] \ln 2}{W_i} \text{ and } \kappa_i = \frac{h_i [V_i]}{W_i \rho T^\text{DL}_t}. \]

Here, \( \nu \) is chosen such that \( \sum_{i \in V_o} P^*_t = P^\text{max} \).

**IV. DISTRIBUTED RISK-SENSITIVE APPROACH FOR TASK FETCHING AND OFFLOADING**

Note that each VUE’s objective \( \mathbb{E}[\exp(\rho T^\text{E2E}_t)] \) in problem (4) is affected by not only its own policy of the task fetching and offloading decision but also the other VUEs’ as per (1) and (2). Hence, the studied risk minimization for all VUEs is a multi-agent problem, where the stable policy of (4) is desired at each VUE. To solve (4), each VUE \( i \) requires the full information of \( [V_i], V_i, \text{ and } h_i, \forall i' \in V \setminus i, \) as per (2).

In other words, the VUE needs to know all possible values of \( \exp(\rho T^\text{E2E}_t) \) in the objective function. However, the required information is cumbersome to fetch in practice. To address this issue while achieving the stable policy, we consider a joint utility and policy estimation-based learning framework, i.e., the distributed no-regret learning algorithm [9], which iteratively estimates the average impact of the aforementioned unavailable information on \( \exp(\rho T^\text{E2E}_t) \) and further obtain a task-fetching and offloading solution to problem (4). The main steps of the distributed no-regret learning algorithm are outlined as follows:

- Each VUE \( i \) is given an initial policy \( \Pr(\alpha_t | h_i) \).
- In each iteration \( t \), VUE \( i \) observes the independent and identically distributed (i.i.d.) realization \( h_i(t) \) and then makes a decision \( \alpha_i(t) \) based on its policy \( \Pr(\alpha_t | h_i) \).

While all the other VUEs follow their stable policies, the VUE deviating from the stable policy cannot further decrease its objective \( \mathbb{E}[\exp(\rho T^\text{E2E}_t)] \).

- Subsequently, cameras and the MEC server are implicitly informed about the VUE compositions of \( V_i(t) \) and \( V_o(t) \), respectively, and send the images to the corresponding VUEs. Once the synthesized image is available, each VUE \( i \) can observe its E2E delay \( T^\text{E2E}_i(t) \) and calculate the utility \(- \exp(\rho T^\text{E2E}_i(t))\).
- Each VUE calculates the average utilities over all past iterations by 1) always making the same decision \( \alpha_t \); and 2) following its iteration-variant policy \( \Pr(\alpha_t | h_i) \). The regret of not insisting on a specific decision is the difference between the average utilities of 1) and 2).
- VUE \( i \) iteratively updates its policy such that the likelihood of making a specific decision is proportional to the corresponding regret calculated over the past iterations.

Let us explain the details in the remainder of this section. Firstly, we denote all unavailable information of VUE \( i \) as \( \omega_i \). Since the objective function of problem (4) is affected by \( \omega_i, h_i, \text{ and } \omega_i \), we define VUE \( i \)’s utility in the \( t \)-th iteration as \( u_i(t) = u_i(\alpha_i(t), h_i(t), \omega_i(t)) = -\exp(\rho T^\text{E2E}_i(t)) \). Additionally, for notational clarity, we use \( \alpha^m \text{ and } h^m \) to express the specific realizations of \( \alpha_i(t) \) and \( h_i(t) \), respectively. The state realizations of \( h_i(t) \) in all iterations are i.i.d., whereas the action realization of \( \alpha_i(t) \) is generated based on some iteration-variant distribution \( \Pr(\alpha_t | h_i) = \pi_t(\cdot - \alpha_t) \). Subsequently, considering the state realization \( h^m \), the conditional regret of the action \( \alpha^m \) in iteration \( t \) is defined as

\[
r^m(t; \alpha^m) = \frac{1}{\sum_{\tau=1}^{t} \mathbb{E}_t[\mathbb{1}_{(h_t(\tau) = h^m)}]} \times \sum_{\tau=1}^{t} [u_i(\alpha^m, h^m, \omega_i(t)) - u_i(t)] \times \mathbb{E}_t[\mathbb{1}_{(h_t(\tau) = h^m)}], \quad (6)
\]

where \( \mathbb{1}_{\{\cdot\}} \) represents the indicator function. In (6), we calculate the average utility over the past iteration instants with \( h_t(\tau) = h^m \) and constantly changing \( \alpha_i(\tau) \), and the average utility of fixing the specific action \( \alpha^m \) over these instants. The conditional regret \( r^m(t; \alpha^m) \) is interpreted as the average utility improvement by fixing the decision \( \alpha^m \) in the iteration instants with \( h_t(\tau) = h^m \). However, since the realization of \( \omega_i(t) \) is unknown, it is not feasible to calculate \( u_i(\alpha^m, h^m, \omega_i(t)) \) and the conditional regret (6). To deal with this issue, the VUE estimates the expected utility with respect to \( \omega_i \) given \( \alpha^m \text{ and } h^m \), i.e., \( \mathbb{E}_i[u_i(\alpha^m, h^m)] \). To this end, the expected utility \( \mathbb{E}_i[u_i(\alpha^m, h^m)] \) and conditional regret \( r^m(t; \alpha^m) \) are estimated, in iteration \( t \), as per

\[
\hat{u}_t^m(t; \alpha^m) = \hat{u}_t^m(t; \alpha^m) + \eta_t(t) \times \mathbb{1}_{(h(t) = h^m)} \times \mathbb{1}_{(\alpha_t(\cdot) = \alpha^m)} \times [u(t) - \hat{u}_t^m(t; \alpha^m)], \quad (7)
\]

\[
\hat{r}_t^m(t; \alpha^m) = \hat{r}_t^m(t; \alpha^m) + \eta_t(t) \times \mathbb{1}_{(h(t) = h^m)} \times [\hat{u}_t^m(t; \alpha^m) - u(t) - \hat{r}_t^m(t; \alpha^m)], \quad (8)
\]

\[ \forall \alpha^m \in \{0, 1\}, h^m \in H_i. \]

Given all the estimated conditional regrets \( \hat{r}_t^m(t; \alpha^m) \) in the current iteration \( t \), a straightforward solution to the task-fetching and offloading problem is making the decision (for each channel value \( h^m \)) with the maximal regret. On the other hands, we also want to comprehensively explore the impacts of the randomness \( \omega_i \) on both \( u_i(\alpha_i = 0, h^m) \) and \( u_i(\alpha_i = 1, h^m) \).
Algorithm 1 Distributed No-Regret Learning Algorithm

1: Initialize \( t = 1 \) and set initial values for \( \hat{u}_h(t; \alpha_i^m), \tilde{r}_h^m(t; \alpha_i^m), \pi_h^m(t; \alpha_i^m), \forall i \in \mathcal{V}, \alpha_i^m \in \{0, 1\}, h_i \in \mathcal{H}_i. \)

2: \textbf{repeat}

3: Observing a realization \( h_i(t) = h_i^l \), each VUE \( i \in \mathcal{V} \) makes a decision \( \alpha_i(t) = \alpha_i^m \) based on its \( \Pr(\alpha_i(t) = \alpha_i^m | h_i(t) = h_i^l) = \pi_h^m(t - 1; \alpha_i^m) \).

4: The MEC server allocates the transmit power for the VUE set \( \mathcal{V}_t(t) \) as per Section III-B.

5: Each VUE \( i \) observes its \( T_i^{E2E}(t) \) and locally updates (7)–(10).

6: \( t \leftarrow t + 1. \)

7: \textbf{until} Stopping criteria are satisfied.

Incorporating these two concerns, the conditional probability distribution for the task-fetching and offloading decision in iteration \( t \) can be modeled as \([9], \forall h_i^l \in \mathcal{H}_i,\)

\[
\beta(\hat{r}_h^m(t); \alpha_i^m) = \arg \max_{\pi_h^m(t; \alpha_i^m)} \left\{ \frac{1}{\xi} \Pr(\alpha_i^m | h_i(t)) \ln \left( \Pr(\alpha_i^m | h_i(t)) \right) \right\}
\]

whose solution is given by

\[
\beta(\hat{r}_h^m(t); \alpha_i^m) = \frac{\exp \left( \xi \hat{r}_h^m(t; \alpha_i^m) \right)}{\sum_{\alpha_i^m=0}^{1} \exp \left( \xi \hat{r}_h^m(t; \alpha_i^m) \right)}, \tag{9}
\]

\( \forall \alpha_i^m \in \{0, 1\}, \) with \( \hat{r}_h^m(t) = \hat{r}_h^m(t; 0), \tilde{r}_h^m(t; 1) \) and \( \hat{r}_h^m(t; \alpha_i^m) = \max \{\hat{r}_h^m(t; \alpha_i^m), 0\} \). Here, \( \xi > 0 \) is the parameter which trades off exploitation (maximizing the expected conditional regret) and exploration (maximizing information entropy). Then combining the estimation rules in (7) and (8) with the distribution (9), we update the task-fetching and offloading policy for VUE as \( i \) as

\[
\pi_h^m(t; \alpha_i^m) = \pi_h^m(t - 1; \alpha_i^m) + \eta_r(t) \times \chi_{h_i(t) = h_i^l}(t) \times \left[ \beta(\hat{r}_h^m(t; \alpha_i^m)) - \pi_h^m(t - 1; \alpha_i^m) \right], \tag{10}
\]

\( \forall \alpha_i^m \in \{0, 1\}, h_i \in \mathcal{H}_i. \) Additionally, in (7), (8), and (10), the learning rates \( \eta_u(t), \eta_r(t), \) and \( \eta_r(t), \) which satisfy \( \Sigma_{t=1}^{\infty} \eta_u(t) = \infty, \Sigma_{t=1}^{\infty} \eta_r(t) = \infty, \Sigma_{t=1}^{N} \eta_r(t) < \infty, \Sigma_{t=1}^{N} \eta_r(t) < \infty, \Sigma_{t=1}^{\infty} \eta_r(t) = 0, \) and \( \Sigma_{t=1}^{\infty} \eta_r(t) = 0, \) can be chosen by referring to \( p \)-series. In the next iteration \( t + 1, \) the action realization of \( \alpha_i(t + 1) \) is generated based on \( \Pr(\alpha_i(t + 1) = \alpha_i^m | h_i(t + 1) = h_i^l) = \pi_h^m(t; \alpha_i^m) \). Finally, the converged distribution \( \pi_h^m(\infty; \alpha_i^m) \) provides a stable solution to the studied problem (4). The related proof of stability can be found in [9]. The steps of the distributed no-regret learning algorithm are outlined in Algorithm 1. Moreover, given that \( E_{\omega_i}[\exp(\rho T_i^{E2E}) | \alpha_i, h_i], \forall \alpha_i, h_i \) is known, problem (4) is a linear programming problem which includes \( 2|\mathcal{H}_i| \) optimization variables and \( 3|\mathcal{H}_i| \) constraints. Here, \( |\mathcal{H}_i| \) increases exponentially with the number \( |\mathcal{C}| + 1. \) In Algorithm 1 which addresses the unavailability dilemma of \( E_{\omega_i}[\exp(\rho T_i^{E2E}) | \alpha_i, h_i], 6|\mathcal{H}_i| \) memory elements (for

V. NUMERICAL RESULTS

We consider Rayleigh fading with unit variance and the path loss model 68.5 + 16.1 log_{10} d (dB) at the 5.9 GHz carrier frequency [11], where \( d \in [1, 100] \) measured in meters is the distance between the camera/MEC server and vehicle. The channel gain is further quantized to two levels. There are four cameras in the simulated network. Considering that the MEC server must have sufficient resources to serve a large number of VUEs, we select \( W_s = 20 \) MHz, \( W_c = 100 \) kHz, \( f_s = 2 \times 10^{11} \) cycle/s, and \( f_t = 10^9 \) cycle/s by referring to [11], [12]. To synthesize images, the same objects in different images need to be recognized first. Therefore, we refer to the processing density for face recognition, i.e., \( L = 2339 \) cycle/bit [13], in the simulations. Some parameters will be specified while investigating their impacts on performance. The rest simulation parameters are \( A = 20 \) kbit, \( B = 60 \) kbit, \( N_0 = -174 \) dBm/Hz, \( P_c = 20 \) dBm, \( P_{max} = 30 \) dBm, \( \xi = 10, \) \( \eta_r(t) = t^{-0.51}, \) \( \eta_r(t) = t^{-0.52}, \) and \( \eta_r(t) = t^{-0.53} \). We run 10000 iterations in Algorithm 1. For performance comparison, we consider the following four baselines:

1. **Average-based** scheme in which \( E[T_i^{E2E}] \), \( \sum_{i \in \mathcal{V}_t} T_i^{DL} \), and \( u_i(t) = -T_i^{E2E}(t) \) are considered in (4), (5), and (7), respectively;
2. **Fully-fetching** scheme with \( \Pr(\alpha_i = 0 | h_i) = 1, \forall i \in \mathcal{V}, h_i \in \mathcal{H}_i; \)
3. **Fully-offloading** scheme with \( \Pr(\alpha_i = 1 | h_i) = 1, \forall i \in \mathcal{V}, h_i \in \mathcal{H}_i; \) and \( \sum_{i \in \mathcal{V}} \exp(\rho T_i^{DL}) \) in (5);
4. **Half-fetching half-offloading** scheme with \( \Pr(\alpha_i = 0 | h_i) = \Pr(\alpha_i = 1 | h_i) = 0.5, \forall i \in \mathcal{V}, h_i \in \mathcal{H}_i; \) and \( \sum_{i \in \mathcal{V}} \exp(\rho T_i^{DL}) \) in (5).
In addition, focusing on the region figure, the CCDF of the proposed approach decays faster is almost equivalent to the fully-offloading scheme with the low-density regime, our approach gives a solution which is slightly higher than the variance of the fully-offloading baseline. When the network becomes dense, less resources will be allocated from the MEC server. In contrast with the fully-offloading baseline, our approach increases the probability of the task-fetching decision \( \Pr(\alpha_i = 0| h_i) \) so that the average E2E delay is improved. The superior mean performance of our approach is rather significant compared with the standard deviation degradation. Finally, due to equal likelihood for both local computation and task offloading, the VUE in the half-fetching half-offloading scheme has the highest delay variance regardless of the network density.

VI. Conclusions

In this letter, we have studied the task-fetching and offloading problem in VEC networks towards URLLC. The studied problem was formulated as a risk minimization problem per VUE’s E2E delay in which the entropic risk measure is defined as our reliability metric. Leveraging a distributed no-regret learning algorithm, we have proposed a risk-sensitive task-fetching and offloading solution. Numerical results have shown a reduction in delay variance compared to the average-based approach and other baseline schemes.

References