This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

EMM: Energy-Aware Mobility Management for Mobile Edge Computing in Ultra Dense Networks

Yuxuan Sun, Sheng Zhou, *Member, IEEE,* and Jie Xu, *Member, IEEE*

***Abstract*—Merging mobile edge computing (MEC) functionality with the dense deployment of base stations (BSs) provides enormous benefits such as a real proximity, low latency access to computing resources. However, the envisioned integration creates many new challenges, among which mobility management (MM) is a critical one. Simply applying existing radio access oriented MM schemes leads to poor performance mainly due to the co-provisioning of radio access and computing services of the MEC-enabled BSs. In this paper, we develop a novel user-centric energy-aware mobility management (EMM) scheme, in order to optimize the delay due to both radio access and computation, under the long-term energy consumption constraint of the user. Based on Lyapunov optimization and multi-armed bandit theories, EMM works in an online fashion without future system state information, and effectively handles the imperfect system state information. Theoretical analysis explicitly takes radio handover and computation migration cost into consideration and proves a bounded deviation on both the delay performance and energy consumption compared to the oracle solution with exact and complete future system information. The proposed algorithm also effectively handles the scenario in which candidate BSs randomly switch on/off during the offloading process of a task. Simulations show that the proposed algorithms can achieve close-to-optimal delay performance while satisfying the user energy consumption constraint.**

***Index Terms*—Mobile edge computing, mobility management, Lyapunov optimization, multi-armed bandit, handover cost.**

I. Introduction

Ultra dense networking (UDN) [2] and mobile edge computing (MEC) [3] [4] are regarded as key building blocks for the next generation mobile network. UDN increases the network capacity through the ultra-dense deployment of small cell base stations (BSs), as a promising technology addressing the so-called 1000x capacity challenge [5]. MEC provides cloud computing and storage resources at the edge of the mobile network, creating significant benefits such as ultra-low latency, intensive computation capabilities while reducing the network congestion, which are necessary for emerging applications such as Internet of things, video stream analysis, augmented reality and connected cars [6].

It is envisioned that endowing each radio access node with cloud functionalities will be a major form of MEC deployment scenarios, e.g., MEC-enabled UDN [7]. However, current studies on UDN and MEC are mostly separate efforts. Despite the enormous potential benefits brought by the integration of UDN and MEC, a key challenge for the overall system performance is mobility management (MM), which is a fundamental function of associating mobile devices with appropriate BSs on the go, thereby enabling continuous mobile services (i.e. radio access and computing). Traditionally, MM was designed only for radio access in less-densified cellular networks. Merging UDN and MEC drastically complicates the problem, due to the co-provisioning of radio access and computing services, and the highly overlapped coverage areas of multiple BSs in the vicinity. Moreover, mobile terminals often has limited battery capacity, which calls for energy efficient MM schemes. In particular, MM for MEC in UDN faces the following three major challenges:

1) For computation tasks offloaded to an MEC-enabled UDN, multiple BSs are available with different radio loads and computation capabilities. Both radio and computation association need to be considered. However, the user lacks the accurate information of all candidate BSs in UDN. Therefore, it is difficult for the user to know a priori which BS offers the best performance.

2) Future information such as task-related parameters, candidate BSs, channel conditions and available edge computing resources are unavailable in advance. Since the mobile user has limited battery power, its long-term energy budget couples the short-term MM decisions across time, and yet the decisions have to be made without foreseeing the future.

3) Moreover, UDN is a very complex and volatile network environment since that many small cell BSs are owned, deployed and managed by end-users. In addition, the operator often implements BS sleeping techniques for energy saving. As a result, candidate BSs can be randomly switched on/off over time, thus demanding for an MM algorithm that can fast track the optimal BS for performance optimization.

A. Related Work

Mobile edge computing has received an increasing amount of attentions recently, see [4] for a comprehensive survey. A central theme of many prior studies is to design task offloading policies and resource management schemes, i.e. what/when/how to offload a user's workload from its device to the edge system or cloud, and how much radio and computing resources should be allocated to each user. For a single-user MEC system, an energy-optimal binary offloading policy is proposed in [8] by comparing the energy consumption of

Y. Sun and S. Zhou are with the Department of Electronic Engineering, Tsinghua University, China. Email: sunyx15@mails.tsinghua.edu.cn, sheng.zhou@tsinghua.edu.cn. (Corresponding author: S. Zhou)

J. Xu is with the Department of Electrical and Computer Engineering, University of Miami, USA. Email: jiexu@miami.edu.

This work is sponsored in part by the Nature Science Foundation of China No. 61571265, No. 91638204, No. 61621091, and Intel Collaborative Research Institute for Mobile Networking and Computing.

Part of this work has been published in IEEE ICC 2017 [1].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

local execution and offloading, while a delay-optimal task scheduling policy with random task arrivals is proposed in [9]. For multi-user MEC systems, both centralized [10] and distributed [11] radio and computation resource management schemes are studied to optimize system-level performance. However, most of the existing works consider a single MEC server, and overlook the user mobility issue.

Mobility management has been extensively investigated in LTE systems. For example, the solutions in [12] work efficiently in less-densified heterogeneous networks, but may bring new problems such as frequent handover and the Ping-Pong effect when the network density becomes high [13]. To address this challenge, an energy-efficient user association and power control policy is proposed in [14], while a learning-based MM scheme is proposed in [15] based on the multi-armed bandits (MAB) theory [16]. Both schemes work in a user-centric manner, which has been an emerging trend of MM for the future 5G network [17]. However, all these works merely consider the radio access. Endowing BSs with MEC capabilities requires new MM solutions.

There are a few works considering service migration, which is a key component of MM in MEC. An optimal computation migration policy is designed in [18], in order to reduce the migration cost while maintaining good user quality of service (QoS). The optimal policy is proved to be threshold-based w.r.t. the migration cost and backhaul data transmission cost in [19]. However, the radio access aspect has not been considered in these works.

Motivated by the limitations of the current literature, we design user-centric MM algorithms in MEC-enabled UDN in this paper. Our work aims to provide guidance to the user about which BS and MEC server should be selected and when to perform handover, with the challenges of lacking both the accurate future information and current BS-side information. By integrating the Lyapunov optimization technique [21] and MAB theory [16], we solve an average delay minimization problem under a long-term energy budget constraint, and prove that our proposed algorithms can provide strong performance guarantee. Note that our work provides the BS association decisions, which can be supported by the link layer handover protocols [13], while further served as the basis of the network layer MM protocols, such as Proxy Mobile IPv6 protocol [20]. Different from the conference version of this work [1], we introduce a more general model considering transmission delay and BS handover cost, and provide new theoretical analysis and simulation results. Moreover, we develop a new algorithm based on the volatile MAB (VMAB) framework [22] to handle random BS on/off during task offloading.

B. Contributions

1) We develop a novel energy-aware user-centric MM scheme, called EMM, to overcome the aforementioned challenges by leveraging the combined power of Lyapunov optimization and MAB theories. The proposed EMM algorithm can deal with various practical deployment scenarios, including those in which the user has limited BS-side information and the BSs dynamically switch on and off.

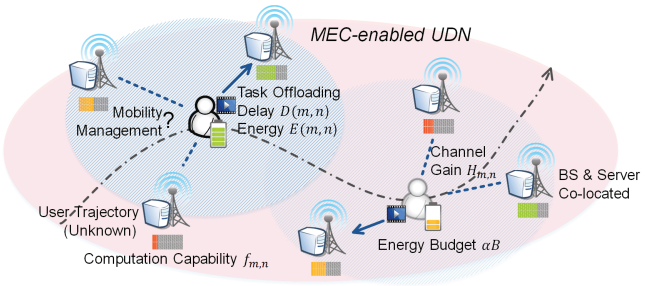


Fig. 1. Illustration of the considered user-centric MM in MEC-enabled UDN. A representative user with unknown trajectory offloads each computation task m to one of the candidate BSs n, based on the overall delay $D(m,n)$ (the sum of communication, computation and handover delay) and the energy consumption $E(m,n)$ for data transmission. The problem is to minimize the average delay under the energy consumption budget αB.

2) We rigorously characterize the performance of the proposed EMM algorithms. We prove that the EMM algorithms can achieve close-to-optimal performance within a bounded deviation without requiring future system information, while satisfying the long-term energy budget constraint. Moreover, we quantify the performance loss due to learning the BS-side information in terms of the learning regret, explicitly taking into account the additional cost caused by radio handover, computation migration and varying candidate BSs.

3) Extensive simulations are carried out to evaluate the performance of the EMM algorithm and validate our theoretic findings. The results confirm that our proposed algorithm can achieve close-to-optimal delay performance compared to the oracle solution with exact and complete future system information, while satisfying the energy consumption constraint of the user. Simulations also reveal the impact of design parameters on the system performance, thereby providing guidelines for real-world deployment of MEC in UDN.

The rest of this paper is organized as follows. We describe the system model and formulate the problem in Section II and III. Section IV and V develop EMM algorithms and conduct performance analysis. Section VI extends the algorithm to handle varying BS sets. Simulation results are provided in Section VII, followed by the conclusion in Section VIII.

II. System Model

A. System Overview

As shown in Fig. 1, we consider an MEC-enabled UDN environment with N densely deployed BSs indexed by $\mathcal{N} = \{1, 2, ..., N\}$, each endowed with cloud computing functionalities. We focus on a representative mobile user moving in the network, generating totally M computation tasks over time that need to be offloaded to the BS for computing. Let L_m denote the location where task m is generated. Due to the dense deployment, multiple BSs can provide service to the user for each task m at location L_m, and these BSs are denoted as $\mathcal{A}(L_m) \subseteq \mathcal{N}$.

We focus on a scenario where the associated BS is responsible for providing both radio access and edge computing services without further offloading the computation tasks to other BSs or the remote cloud. User-centric MM algorithm is considered, which enables the user to make the MM decision

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

on the BS association and handover based on its own QoS requirement [17]. The user will choose one BS among the set of candidate BSs $\mathcal{A}(L_m)$ to serve each task. Tasks can be further divided into subtasks and offloaded to different candidate BSs. The objective of our MM design is to minimize the average delay (consisting of communication, computation and handover delay) over M tasks under the constraint of the total energy consumption budget of the user for uplink data transmission. MM decisions are made in an online fashion without requiring future information of task-related parameters, wireless channel states, computing capability and user trajectory. Thus our work is applicable to any mobility model, such as the random waypoint model or others as described in [23]. Also, the subtask division can be exploited for system state learning, as shown in Section V.

B. Computation Task Offloading

A widely used three-parameter model (see [4] and references therein) is adopted to describe each computation task m: input data size $\lambda_m \in [0, \lambda_{\max}]$ (in bits) that needs to be offloaded, computation intensity $\gamma_m \in [0, \gamma_{\max}]$ (in CPU cycles per bit) indicating how many CPU cycles are required to compute one bit input data, and completion deadline D_m. Parameters $\lambda_{\max}$ and $\gamma_{\max}$ are the maximum possible input data size and computation intensity, respectively.

Each computation task is relatively large and hence can be further divided into subtasks that are processed in sequence. Taking video stream analytics as an example, like object detection or tracking from a video stream, the analysis can be operated on the edge server using the Hadoop MapReduce framework [24]. A relatively long video frame is further divided into short video clips through video segmentation. Note that our work is orthogonal to the video segmentation problem [25], and we omit the overhead of video segmentation for simplicity, which can be seen as an additional constant delay to the system performance. Let $K_m \leq \bar{K}$ be the number of subtasks of task m, where $\bar{K}$ is maximum number of subtasks. Assume that subtasks are of the equal size λ_0 for analytical simplicity (hence $\lambda_m = K_m \lambda_0$). Nevertheless, our framework can handle subtasks of heterogeneous sizes.

Each BS $n \in \mathcal{N}$ is equipped with an MEC server of maximum CPU frequency F_n (in CPU cycles per second), and can provide computation services for multiple tasks from multiple users simultaneously using processor sharing. We use computation capability $f_{m,n}$ to describe the CPU frequency that BS n can allocate to task m, which depends on several factors on the BS side, such as the maximum CPU frequency F_n, the current total workload intensity, etc. We assume that $f_{m,n}$ does not change during the processing of one task but can change across tasks. If BS n is selected to compute a subtask of size λ_0 and computation intensity γ_m, then given the allocated CPU frequency $f_{m,n}$, the computation delay is

$$d_c(m,n) = \frac{\lambda_0 \gamma_m}{f_{m,n}}. \tag{1}$$

C. Communication and Energy Consumption

The input data is transmitted from the user to the serving BS through the wireless uplink channel. Denote $H_{m,n}$ as the channel gain between the user at location L_m and BS $n \in \mathcal{A}(L_m)$. We assume that during the computation of each task m, the user does not move much and hence $H_{m,n}$ is constant. Nevertheless, if the user moves considerably, we consider that one task is divided into multiple subtasks, and for each subtask the user stays more or less at the same location. Given the transmission power P_{tx} of the user, the maximum achievable uplink transmission rate is given by:

$$r(m,n) = W \log_2 \left(1 + \frac{P_{\text{tx}} H_{m,n}}{\sigma^2 + I_{m,n}}\right), \tag{2}$$

where W is the channel bandwidth, σ^2 is the noise power and $I_{m,n}$ is the inter-cell interference power at BS n while offloading task m. The transmission delay for sending the input data of size λ_0 to BS n is thus

$$d_t(m,n) = \frac{\lambda_0}{r(m,n)}. \tag{3}$$

The energy consumption for offloading a subtask for task m is the transmit power P_{tx} multiplied by the transmission delay:

$$e(m,n) = \frac{P_{\text{tx}} \lambda_0}{r(m,n)}. \tag{4}$$

Remark 1. *Downlink transmission delay and packet loss are not considered in this work. Nevertheless, the following analysis and the proposed solutions are still applicable with these considerations. For example, downlink transmission delay and packet loss can be reflected by additional transmission delay that changes expression* (3)*.*

D. Handover and Migration Cost

For each computation task m, its subtasks must be computed in sequence, but can be offloaded to different BSs. This may be because the user learns that the serving BS's computing capability is weak (we will introduce the learning problem in Section V) and hence decides to switch to a different BS in its vicinity or BSs can ppear or disappear in the transmission range of the user due to dynamic BS on/off for energy saving [26]. When consecutive subtasks are processed on different BSs, an additional delay cost is incurred due to the handover procedure and the computation migration. Let C_m be the one-time handover cost for task m. Given the sequences of BSs that serve its subtasks, denoted by $\boldsymbol{a}_m = (a_m^1, a_m^2, ..., a_m^{K_m})$, the overall handover cost for task m is

$$h(m, \boldsymbol{a}_m) = C_m \sum_{k=2}^{K_m} \mathbb{I}\{a_m^k \neq a_m^{k-1}\}, \tag{5}$$

where $a_m^k \in \mathcal{A}(L_m)$ is the serving BS for subtask k of task m, and $\mathbb{I}\{x\}$ is an indicator function with $\mathbb{I}\{x\} = 1$ if event x is true and $\mathbb{I}\{x\} = 0$ otherwise.

III. Problem Formulation

Mobile users often have limited energy budgets (e.g., due to limited battery capacity). Therefore, the objective of the mobile user is to make MM decisions, specifically which BS to associate and when to perform handover, in order to minimize

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

the average delay given its limited energy budget. For task m, the overall delay is

$$D(m, \boldsymbol{a}_m) = \sum_{k=1}^{K_m} d(m, a_m^k) + h(m, \boldsymbol{a}_m), \tag{6}$$

where $d(m, a_m^k) \triangleq d_c(m, a_m^k) + d_t(m, a_m^k)$ is the sum of computation delay and uplink transmission delay for subtask k. The overall energy consumption for processing task m is

$$E(m, \boldsymbol{a}_m) = \sum_{k=1}^{K_m} e(m, a_m^k). \tag{7}$$

Formally, the problem is formulated as follows

$$\textbf{P1:} \quad \min_{\boldsymbol{a}_1, ..., \boldsymbol{a}_M} \quad \frac{1}{M} \sum_{m=1}^{M} D(m, \boldsymbol{a}_m) \tag{8}$$

$$\text{s.t.} \quad \sum_{m=1}^{M} E(m, \boldsymbol{a}_m) \leq \alpha B \tag{9}$$

$$D(m, \boldsymbol{a}_m) \leq D_m, \ \forall m \tag{10}$$

$$a_m^k \in \mathcal{A}(L_m), \forall m, \forall k = 1, 2, ..., K_m. \tag{11}$$

The first constraint (9) states that the total energy consumption is limited by the energy budget of the user, where $\alpha \in (0, 1]$ indicates the desired capping of energy consumption relative to the total battery capacity B. The second constraint (10) requires that the overall delay for processing task m does not exceed the completion deadline D_m. Note that even if we set up a deadline for each task, the user still prefers to receive the result as soon as possible. The last constraint (11) states that the associated BSs are those that cover location L_m.

There are two major challenges to solve problem **P1**. First, optimally solving **P1** requires complete non-causal information over the entire trip of the user, including parameters of all tasks, user trajectory, traffic intensity of all BSs, etc., which is impossible to acquire in advance. Furthermore, **P1** belongs to integer nonlinear programming problem. Even if the complete future information is known a priori, it is still difficult to solve due to the high complexity. Therefore, we will propose online algorithms that can efficiently make MM decisions without the future information.

A. Oracle Benchmark and Theoretical Upper Bound

In this subsection, we describe an algorithm that knows the complete future information for the next J computation tasks. Albeit impractical, the purpose of introducing this algorithm is merely to provide theoretical upper bounds on the performance of any practical online algorithm. We will prove later that our proposed algorithm achieves close-to-optimal performance by comparing to this oracle benchmark.

The J-step lookahead problem is defined as

$$\textbf{P2:} \quad \min_{\boldsymbol{a}_{rJ+1}, ..., \boldsymbol{a}_{(r+1)J}} \quad \frac{1}{J} \sum_{m=rJ+1}^{(r+1)J} D(m, \boldsymbol{a}_m) \tag{12}$$

$$\text{s.t.} \quad \sum_{m=rJ+1}^{(r+1)J} E(m, \boldsymbol{a}_m) \leq \frac{\alpha B}{R} \tag{13}$$

$$\text{constraints (10), (11).} \tag{14}$$

The entire trip of the user is divided into $R \geq 1$ frames. In each frame, the user generates $J \geq 1$ tasks and hence $M = RJ$. We assume that there is an oracle that provides accurate information of the subsequent J tasks at the beginning of each frame. Given this information, the user can obtain the MM decisions for the next J tasks by solving the J-step lookahead problem **P2**.

Clearly if $R = 1$, then the J-step lookahead problem is the original offline problem **P1**. Assume that for all $r = 0, 1, ..., R-1$, there exists at least one sequence of MM decisions $\boldsymbol{a}_{rJ+1}, ..., \boldsymbol{a}_{(r+1)J}$ that satisfy the constraints of **P2**. Denote g_r^* as the optimal average delay achieved by **P2** in the r-th frame. Thus $g^* = \frac{1}{R} \sum_{r=0}^{R-1} g_r^*$ is the minimum long-term average delay achieved by the J-step lookahead problem.

IV. Online Mobility Management Framework

In this section, we develop a framework that supports online MM requiring only causal information. Specifically, when making the MM decisions for task m, the user has no information about tasks $m+1$, $m+2$, ... We will prove that our proposed algorithm achieves close-to-optimal performance compared with the oracle algorithm with J-step lookahead. The information regarding task m can be classified into two categories depending on which entity possesses the information:

- **User-Side State Information**: The user's location L_m, the available candidate BSs $\mathcal{A}(L_m)$, the input data size λ_m and the computation intensity γ_m.
- **BS-Side State Information**: For each BS $n \in \mathcal{A}(L_m)$, the allocated CPU frequency $f_{m,n}$, the uplink channel gain $H_{m,n}$ and the inter-cell interference $I_{m,n}$.

Depending on whether the user has the BS-side state information, we will consider two deployment scenarios. In the first scenario, the user knows both the user-side state information and BS-side state information exactly, i.e., the user has Global State Information (GSI). In the second scenario, the user only has the user-side state information, i.e., the user has Local State Information (LSI). In this case, the user needs to learn the BS-side state information in order to make proper MM decisions.

A. EMM-GSI Algorithm

In this subsection, we present online MM framework for the scenario with GSI. Assume that the serving BS set does not change during one task, then it is clear that if the user has GSI, radio handover and computation migration of subtasks can be avoided. It is straightforward for the user to select the best BS for offloading and computation and stick to the BS for the entire task. Therefore, for each task m, all the subtasks are served by the optimal BS a_m^*, i.e., $a_m^1 = a_m^2 = ... = a_m^{K_m} = a_m^*$. We use $D(m, n)$ to denote the overall delay and $E(m, n)$ to denote the overall energy consumption by associating to BS n for task m with GSI.

However, a significant challenge remains in directly solving **P1** since the long-term energy consumption budget couples the MM decisions across different tasks: using more energy for the current task will potentially reduce the energy budget available

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

for future uses, and yet the decisions have to be made without foreseeing the future. To address this challenge, we leverage Lyapunov optimization technique which enables us to solve a deterministic problem for each task with low complexity, while adaptively balancing the delay performance and energy consumption over time.

To guide the MM decisions with Lyapunov optimization technique, we first construct a virtual energy deficit queue. Specifically, the energy deficit queue evolves as

$$q(m+1) = \max\{q(m) + E(m, a_m^*) - \alpha B/M, 0\}, \quad (15)$$

with $q(0) = 0$. The virtual queue length $q(m)$ indicates how far the current energy usage deviates from the battery energy budget. Since the battery capacity of the user device is finite, it is necessary to consider the case with finite tasks and propose an approach that can guarantee the worst-case delay performance over the finite time horizon. Moreover, both the user-side state information and BS-side state information may not follow a well-defined stochastic process. Therefore, we do not make any ergodic assumptions on the state information. Instead, we adopt a non-ergodic version of Lyapunov optimization, which applies to any arbitrary sample path of the task and system dynamics. The algorithm is called EMM-GSI, as shown in Algorithm 1.

Algorithm 1 EMM-GSI Algorithm

1: **Input**: L_m, $\mathcal{A}(L_m)$, λ_m, γ_m, and $\forall n \in \mathcal{A}(L_m)$, $f_{m,n}$, $H_{m,n}$, $I_{m,n}$ at the beginning of offloading each task m.
2: **if** $m = rJ + 1, \forall r = 0, 1, ..., R-1$ **then**
3: $q(m) \leftarrow 0$ and $V \leftarrow V_r$.
4: **end if**
5: Choose a_m^* subject to (10), (11) by solving

$$(\mathbf{P3}) \quad \min_{n \in \mathcal{A}(L_m)} VD(m,n) + q(m)E(m,n).$$

6: Update $q(m)$ according to (15).

Note that EMM-GSI algorithm works in an online fashion, because it only requires the currently available information as the inputs. $V_0, V_1, ..., V_{R-1}$ is a sequence of positive control parameters to dynamically adjust the tradeoff between delay performance and energy consumption over the R frames, each with J periods. Lines 2 - 4 reset the energy deficit virtual queue at the beginning of each frame. Line 5 defines an online optimization problem **P3** to decide the MM decisions for each task, which is a minimum seeking problem with computational complexity $O(|\mathcal{A}(L_m)|)$, where $|\mathcal{A}(L_m)|$ is the number of candidate BSs for task m. The optimization problem aims to minimize a weighted sum of the delay cost and energy consumption where the weight depends on the current energy deficit queue length and is varying over time. A large weight will be placed on the energy consumption if the current energy deficit is large. The energy deficit queue maintains without foreseeing the future, thereby enabling online decisions. Note that since there is no radio handover and computation migration, **P3** is equivalent to

$$\min_{n \in \mathcal{A}(L_m)} Vd(m,n) + q(m)e(m,n). \quad (16)$$

Conveniently, we write $z(m,n) \triangleq Vd(m,n) + q(m)e(m,n)$.

B. Performance Bound

In this subsection, we present the performance analysis of the EMM-GSI algorithm. Under the feasibility assumption that there exists at least one solution to **P2**, Theorem 1 provides the performance guarantee of EMM-GSI algorithm.

Theorem 1. *For any fixed integer $J \in \mathbb{Z}_+$ and $R \in \mathbb{Z}_+$ such that $M = RJ$, the following statements hold.*

(1) The average delay performance achieved by EMM-GSI algorithm satisfies:

$$d_G^* \leq \frac{1}{R}\sum_{r=0}^{R-1} g_r^* + \frac{UJ}{R}\sum_{r=0}^{R-1}\frac{1}{V_r}, \quad (17)$$

where g_r^* is the optimal average delay of the J-step lookahead problem for frame r, and U is a constant defined as $U \triangleq \frac{1}{2}\max\{(E(m,a_m) - \alpha B/M)^2\}$.

(2) The total energy consumption is within a bounded deviation:

$$e_G^* \leq \alpha B + \sum_{r=0}^{R-1}\sqrt{2UJ^2 + 2V_r J g_r^*}. \quad (18)$$

Proof. See Appendix A in [27]. □

Theorem 1 shows that using the proposed EMM-GSI algorithm, the worst-case average delay is no more than $O(1/V)$ with respect to the optimal average delay achieved by the J-step lookahead problem. Meanwhile, the energy consumption is within a bounded deviation $O(V)$ compared to the given energy budget. Hence, there exists a delay-energy tradeoff of $[O(1/V), O(V)]$. By adjusting V, we can balance the average delay and energy consumption.

V. Learning with LSI Only

In this section, we consider the scenario that the user has LSI only. We augment our EMM algorithm with online learning based on the MAB framework in order to learn the optimal BS (i.e. the solution to **P3**) without initially requiring the BS-side information. Learning the optimal BS incurs additional costs since (1) suboptimal BSs will be selected during the learning process, and (2) radio handover and computation migration is inevitable. We also provide theoretical bounds on the performance loss of the proposed algorithm due to learning.

A. EMM-LSI Algorithm

When the user has only LSI, MM is much more difficult since there is no *a priori* information about which BS provides the best delay performance while incurring less energy consumption. Specifically, the user cannot directly solve **P3** since $d(m,n)$ and $e(m,n)$ rely on BS-side information such as $f_{m,n}$, $H_{m,n}$ and $I_{m,n}$, which are unknown. Thus the user has to learn the optimal BS on-the-fly.

A straightforward learning scheme is as follows: the user offloads one subtask of task m to every BS n in $\mathcal{A}(L_m)$ and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

observes the computation delay $\tilde{d}(m,n)$ and energy consumption $\tilde{e}(m,n)$ (and hence the observed $\tilde{z}(m,n) = V\tilde{d}(m,n) + q(m)\tilde{e}(m,n)$). If observations are accurate, namely $\tilde{d}(m,n) = d(m,n)$ and $\tilde{e}(m,n) = e(m,n)$ (and hence $\tilde{z}(m,n) = z(m,n)$), then learning can be terminated and the remaining $K_m - |\mathcal{A}(L_m)|$ subtasks of task m will be offloaded to the BS that is the solution to $\min_n \tilde{z}(m,n)$. However, due to the variance in computation intensity, wireless channel state and many other factors, $\tilde{z}(m,n)$ is only a noisy version of $z(m,n)$. In the presence of such measurement variance, this simple learning algorithm can perform very poorly since the user may get trapped in a BS whose $z(m,n)$ is actually large. Therefore, a more sophisticated and effective learning algorithm requires continuous learning to smooth out the measurement noise. In fact, MM with only LSI manifests a classic sequential decision making problem that involves a critical tradeoff between exploration and exploitation: the user needs to explore the different BSs by offloading subtasks to them in order to learn good estimates of $z(m,n), \forall n \in \mathcal{A}(L_m)$, while at the same time it wants to offload as many subtasks as possible to the *a priori unknown* optimal BS.

Sequential decision making problems under uncertainties have been studied under the MAB framework and efficient learning algorithms have been developed that provide strong performance guarantee. In this paper, we augment our EMM algorithm with the so-called UCB1 algorithm [16] to learn the optimal BS. Specifically, UCB1 is an index-based algorithm, which assigns an index to each candidate BS and updates the indices of the BSs as more subtasks of a task have been offloaded. Then the next subtask will be offloaded to the BS with the largest index. The index for a BS $n \in \mathcal{A}(L_m)$ is in fact an upper confidence bound on the empirical estimate of $z(m,n)$. Nevertheless, learning algorithms other than UCB1 can also be incorporated in our framework.

The EMM-LSI algorithm is shown in Algorithm 2. The major difference from Algorithm 1 is that instead of solving **P3** exactly, we use the UCB1 algorithm as a subroutine to learn the optimal BS to minimize the objective in **P3**, which is reflected from Lines 5 through 15. Let $\bar{z}_{m,n,k}$ denote empirical sample-mean estimate of $z(m,n)$ after the first k subtasks have been offloaded and their corresponding delay and energy performance have been measured. We use $\theta_{m,n,k}$ to denote the number of subtasks that have been offloaded to BS n up to subtask k. Lines 5-9 is the initialization phase, and Lines 10-15 is the continuous learning phase. The decision making problem for each subtask is a minimum seeking problem with computational complexity $O(|\mathcal{A}(L_m)|)$, thus for each task, the computational complexity of the EMM-LSI algorithm is $O(K_m|\mathcal{A}(L_m)|)$.

Algorithm 2 EMM-LSI Algorithm

1: **Input**: L_m, $\mathcal{A}(L_m)$, λ_m, γ_m at the beginning of offloading each task m.
2: **if** $m = rJ+1, \forall r = 0,1,...,R-1$ **then**
3: $q(m) \leftarrow 0$ and $V \leftarrow V_r$.
4: **end if**
5: **for** $k = 1,...,|\mathcal{A}(L_m)|$ **do** ▷ *UCB1 Learning*
6: Connect to each BS $n \in \mathcal{A}(L_m)$ once.
7: Update $\bar{z}_{m,n,k} = V\tilde{d}(m,n) + q(m)\tilde{e}(m,n)$.
8: Update $\theta_{m,n,k} = 1$.
9: **end for**
10: **for** $k = |\mathcal{A}(L_m)|+1,...,K_m$ **do**
11: Connect to $a_m^k = \arg\min_n \left\{ \bar{z}_{m,n,k} - \beta\sqrt{\frac{2\ln k}{\theta_{m,n,k}}} \right\}$.
12: Observe $\tilde{d}(m,a_m^k)$ and $\tilde{e}(m,a_m^k)$.
13: $\bar{z}_{m,a_m^k,k} \leftarrow \frac{\theta_{m,a_m^k,k}\bar{z}_{m,a_m^k,k} + V\tilde{d}(m,a_m^k) + q(m)\tilde{e}(m,a_m^k)}{\theta_{m,a_m^k,k}+1}$.
14: $\theta_{m,a_m^k,k} \leftarrow \theta_{m,a_m^k,k} + 1$.
15: **end for**
16: Update $q(m)$ according to (15).

B. Algorithm Performance

In this subsection, we analyze the performance of EMM-LSI. We first bound the gap between the exact solution of **P3** with GSI and the UCB1 learning algorithm with LSI for each task. We adopt the concept of *learning regret* to measure the performance loss for each task due to learning, which is commonly used in the MAB framework [16]. Formally, the learning regret is defined as follows

$$R_m = \mathbb{E}[Z(m,\boldsymbol{a}_m) - Z(m,a_m^*)], \tag{19}$$

where $Z(m,\boldsymbol{a}_m) = VD(m,\boldsymbol{a}_m) + q(m)E(m,\boldsymbol{a}_m)$ is the weighted cost achieved by the sequence of MM decisions $\boldsymbol{a}_m$ resulted from UCB1, and $Z(m,a_m^*) = VD(m,a_m^*) + q(m)E(m,a_m^*)$ is achieved by always connecting to the optimal BS a_m^* that solves **P3**.

Although the learning regret of the UCB1 algorithm has been well understood, characterizing that in our setting faces new challenges: the learning regret is a result of not only offloading subtasks to suboptimal BSs, but also radio handover and computation migration. Specifically, the learning regret can be decomposed into two terms [28], namely the *sampling regret* and the *handover regret*:

$$R_m = \underbrace{\mathbb{E}\left[\sum_{k=1}^{K_m} z(m,a_m^k) - Z(m,a_m^*)\right]}_{\text{sampling regret}} + V\underbrace{\mathbb{E}\left[h(m,\boldsymbol{a}_m)\right]}_{\text{handover regret}}. \tag{20}$$

We provide an upper bound on the learning regret of UCB1 considering the handover regret in the following proposition.

Proposition 1. *For task m comprising K_m subtasks, the learning regret R_m is upper bounded as follows:*

$$R_m(K_m) \le \beta\left[8\sum_{n\neq a_m^*}\frac{\ln K_m}{\delta_{m,n}} + \left(1+\frac{\pi^2}{3}\right)\sum_{n\neq a_m^*}\delta_{m,n}\right] + VC_m\left[2\sum_{n\neq a_m^*}\left(\frac{8\ln K_m}{\delta_{m,n}^2} + 1 + \frac{\pi^2}{3}\right) + 1\right], \tag{21}$$

where $\beta = \sup_n \tilde{z}(m,n)$ and $\delta_{m,n} = (Z(m,n) - Z(a_m^*))/\beta K_m$.

Proof. See Appendix B in [27]. □

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

Remark 2. *Parameter β is used to normalize the utility function. In real implementations, it is difficult to obtain the exact value of β due to lack of the BS-side state information. However, a reasonably good estimate of β can be obtained based on the history data, e.g., setting β as the maximum $\tilde{z}(m,n)$ that has been observed.*

The bound on the learning regret established in Proposition 1 is logarithmic in the number of subtasks K_m. It also implies that **P3** can be approximately solved by UCB1 within a bounded deviation, denoted by W, since K_m is upper bounded by $\bar{K}$. The performance of EMM-LSI can then be expressed in Theorem 2.

Theorem 2. *For any fixed integer $J \in \mathbb{Z}_+$ and $R \in \mathbb{Z}_+$ such that $M = RJ$, the following statements hold.*

(1) The average delay performance achieved by EMM-LSI algorithm satisfies:

$$d_L^* \leq \frac{1}{R}\sum_{r=0}^{R-1} g_r^* + \frac{UJ+W}{R}\sum_{r=0}^{R-1}\frac{1}{V_r}. \tag{22}$$

(2) The total energy consumption is within a bounded deviation:

$$e_L^* \leq \alpha B + \sum_{r=0}^{R-1}\sqrt{2[UJ^2 + V_r J g_r^* + WJ]}. \tag{23}$$

Proof. See Appendix C in [27]. □

Theorem 2 shows that the proposed EMM-LSI algorithm can provide a strong performance guarantee: even if the user cannot acquire the exact BS-side state information, the average delay performance can still be guaranteed through the proposed algorithm, while the energy consumption is within a bounded deviation from the given energy budget.

C. Implementation Considerations

In the proposed EMM-LSI algorithm, the user keeps learning the optimal BS while offloading all K_m subtasks of task m. Although Proposition 1 provides an upper bound on the performance loss due to continuous learning, in practice, the loss can be large when the one-time handover cost is relatively large. For instance, when the second-best BS has a similar value of $z(m,n)$ as the optimal BS, the UCB1 algorithm can keep alternating between these two BSs for many subtasks, thereby incurring a significant handover and migration cost. To circumvent this issue, there are two possible heuristic schemes.

1) The first scheme stops learning after a pre-determined finite number K_s of times of subtask offloading. That is, UCB1 is applied only for the first K_s subtasks. The remaining $K_m - K_s$ subtasks, if any, will all be offloaded to the BS with the lowest value of $\bar{z}_{m,n,K_s}$. Clearly, there is a tradeoff for deciding K_s: if K_s is too small, the probability that a suboptimal BS is regarded as the optimal is high, and hence, leading to a large cost for offloading the remaining subtasks to the suboptimal BS. On the other hand, if K_s is too large, a large handover cost may be incurred. We will quantify this tradeoff in our simulation results.

2) The second scheme stops learning when the best and second-best BSs have very similar performance. Specifically, the stopping criteria is

$$\bar{z}_{m,n^*,k} - \bar{z}_{m,n^\dagger,k} \leq \epsilon \tag{24}$$

$$\theta_{m,n^*,k} \geq K_0, \theta_{m,n^\dagger,k} \geq K_0, \tag{25}$$

where n^* is the learned best BS and $n^\dagger$ is the learned second-best BS so far, and ϵ, K_0 are pre-determined parameters.

VI. Varying BS Set

In this section, we consider a more general setting in which the set of candidate BSs during the offloading of one task can vary. For example, BSs are turned on/off according to the BS sleeping strategy for energy saving purposes [26] or small cell owner-governed processes. We develop a modified version of the EMM-LSI algorithm, called EMM-LSI-V, based on the VMAB framework and characterize its performance.

A. EMM-LSI-V Algorithm

The varying set of BSs creates a big challenge in learning the optimal BS that solves **P3**. With the conventional UCB1 algorithm, the user has to restart the learning process whenever a new BS appears. Apparently, this learning strategy is very inefficient since it simply restarts the learning process without reusing what has been learned. Although the available BS set changes, the states of other BSs are likely to remain the same. Therefore, proper learning algorithms that effectively reuse the already learned information are needed.

To efficiently learn the optimal BS among a varying BS set, we adopt the VMAB framework [22], in which BSs can appear or disappear unexpectedly with unknown lifespan. Define an epoch as the interval in which the available BS set is invariant, and let B_m be the total number of epochs for task m, which is unknown in advance. Note that $B_m = 1, \forall m$ corresponds to the case that we considered in Section V. The available BS set for epoch $b = 1, 2, ..., B_m$ is denoted as $\mathcal{A}_{m,b}$ and let $\mathcal{A}_m$ be the union of $\mathcal{A}_{m,b}, \forall b = 1, ..., B_m$. To simplify the problem, we assume that each BS only appears once during each task. If a BS appears for the second time, it can be treated as a new BS. For each BS $n \in \mathcal{A}_m$, the lifespan is denoted as $[u_n, v_n]$ with $1 \leq u_n, v_n \leq K_m$, which indicates that BS n is present from subtask u_n through subtask v_n. We also denote $K_{m,b}$ as the total number of subtasks of task m completed by the end of epoch b. Clearly, $K_{m,B_m} = K_m$.

The EMM-LSI-V algorithm developed on volatile UCB1 (VUCB1) learning is proposed in Algorithm 3. In VUCB1 learning, a UCB1-like algorithm is implemented for each epoch. The differences are two-fold. First, the initialization for each epoch (Lines 6-10) only applies to the newly appeared BSs, while the information for the remaining BSs is retained and hence reused. Second, the index term on Line 12 used to guide the subtask offloading decision takes into account the appearance time of the BS.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

Algorithm 3 EMM- LSI-V Algorithm

Input: L_m, λ_m, γ_m at the beginning of offloading each task m.
if $t = rJ + 1, \forall r = 0, 1, ..., R-1$ **then**
  $q(m) \leftarrow 0$ and $V \leftarrow V_r$.
end if
for $k = 1, ..., K_m$ **do** ▷ *VUCB1 Learning*
  if k is the first block of an epoch **then**
    Input: $\mathcal{A}_{m,b}$
    Connect to each first appeared BS $n \in \mathcal{A}_{m,b}$ once.
    Update $\bar{z}_{m,n,k} = V\tilde{d}(m,n) + q(m)\tilde{e}(m,n)$.
    Update $\theta_{m,n,k} = 1$.
  else
    $a_m^k = \arg\min_n \left\{ \bar{z}_{m,n,k} - \beta\sqrt{\frac{2\ln(k-u_n)}{\theta_{m,n,k}}} \right\}$, connect to BS a_m^k.
    Observe $\tilde{d}(m, a_m^k)$ and $\tilde{e}(m, a_m^k)$.
    $\bar{z}_{m,a_m^k,k} \leftarrow \frac{\theta_{m,a_m^k,k}\bar{z}_{m,a_m^k,k} + V\tilde{d}(m,a_m^k) + q(m)\tilde{e}(m,a_m^k)}{\theta_{m,a_m^k,k}+1}$.
    $\theta_{m,a_m^k,k} \leftarrow \theta_{m,a_m^k,k} + 1$.
  end if
end for
Update $q(m)$ according to (15).

B. Algorithm Performance

We characterize the performance of the VUCB1 learning as follows. Let $a_{m,b}^*$ as the optimal BS at epoch b for task m. The learning regret is thus

$$R_m = \underbrace{\sum_{b=1}^{B_m} \mathbb{E}\left[\sum_{k=K_{m,b-1}+1}^{K_{m,b}} z(m, a_m^k) - Z(m, a_{m,b}^*) \right]}_{\text{sampling regret}} + V \underbrace{\mathbb{E}\left[h(m, \boldsymbol{a}_m)\right]}_{\text{handover regret}}. \quad (26)$$

Proposition 2. *For task m comprising K_m subtasks, if there are B_m epochs, the total regret R_m of VUCB1 is of $O(B_m \ln K_m)$.*

Proof. See Appendix D in [27]. □

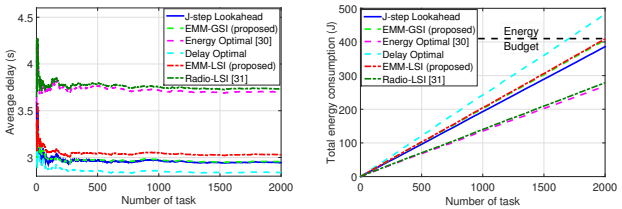
Proposition 2 states that VUCB1 learning can provide a bounded deviation, defined as W', from exactly solving **P3**. Therefore, our EMM-LSI-V algorithm can still provide strong performance guarantee by substituting the bounded deviation W with W' in Theorem 2.

VII. Simulations

In this section, we evaluate the average delay performance and total energy consumption of the proposed EMM algorithms and verify the theoretical results through simulations using MATLAB. We simulate a 1km×1km square area with 49 BSs deployed on a regular grid network. The user can associate with BSs within a radius of 150m. The user trajectory is generated by the random walk model. The wireless channel gain is modeled as $H_{m,n} = 127 + 30 \times \log d$, as suggested in [29]. Besides, channel bandwidth $W = 20$MHz, noise power $\sigma^2 = 2 \times 10^{-13}$W, and transmit power $P_{\text{tx}} = 0.5$W.

We consider an application of video stream analysis with totally $M = 500$ video tasks generated during the entire trip. Each subtask is a one-second video clip. According to [24], we set $\lambda_0 = 0.62$Mbits, which is the data size of a one-second QCIF format video with 176×144 video resolution, 24.8k pixels per frame and 25 fps (frame per second). Each video is set to be 1min to 2min long, i.e., K_m is uniformly selected from $\{60, 61, ..., 120\}$, thus $\lambda_m \in [37.2, 74.4]$ Mbits. Each subtask has completion deadline 150ms, and the computation intensity γ_m is uniformly distributed within $[500, 1000]$ cycles/bit. Each MEC sever is equipped with multiple CPU cores, and the sum frequency $F_n = 25$GHz. The available computation capability for each task follows uniform distribution with $f_{m,n} \in [0, F_n]$ GHz. In addition, one-time handover cost $C_m = 5$ms, and battery capacity $B = 1000$J.

We introduce four benchmark algorithms to evaluate the performance of the proposed EMM algorithms: 1) **J-step Lookahead**: this is the oracle benchmark described in Section III-A. We set $J = 5$ and thus $R = M/J = 100$. Note that solving the J-step lookahead problem is extremely computationally complex. 2) **Delay Optimal (GSI)**: the user always associates with the BS with the lowest delay and disregards the energy consumption constraint. 3) **Energy Optimal (GSI)**: the user always associates with the BS with the best channel condition without considering the delay performance. In fact, this is the standard 3GPP LTE handover protocol with Event A3 handover condition where the handover offset is set to be zero (see [30], Sec. 5.5.4). Both delay optimal and energy optimal benchmarks are implemented in the GSI scenario. 4) **Radio-LSI**: this benchmark learns the BS with best channel condition based on the MAB theory [31]. It is implemented in the LSI scenario to compare with the EMM-LSI algorithm.



(a) Average delay (b) Total energy consumption

Fig. 2. Performance of EMM ($V = 0.01$, $\alpha B = 410$J, $K_s = 20$, 30% observation variance).

Fig. 2 compares the average delay performance and total energy consumption over the M tasks of EMM-GSI, EMM-LSI and four benchmark algorithms. Here we set 30% observation variance in the LSI scenario and let EMM-LSI algorithm stop learning after offloading $K_s = 20$ subtasks to avoid frequent radio handover and computation migration, as discussed in Section V-C. As can be seen, our two EMM algorithms satisfy the energy consumption constraint while keeping the delay low. In the GSI scenario, EMM-GSI algorithm effectively balances delay and energy consumption and achieves the delay close to the J-step Lookahead. EMM-LSI algorithm is just slightly worse than EMM-GSI algorithm. Compared with the Radio-LSI algorithm, EMM-LSI algorithm performs better in delay performance since it learns both radio and computation states rather than only the wireless channel condition.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSAC.2017.2760160, IEEE Journal on Selected Areas in Communications

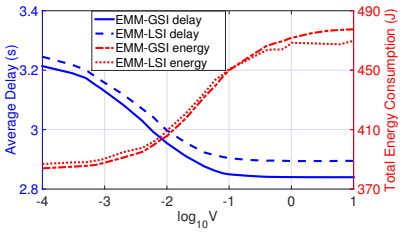
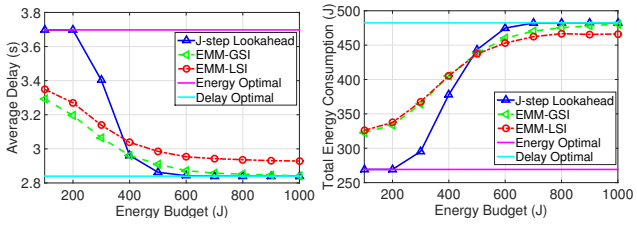


Fig. 3. Impact of V ($\alpha B = 410$J, $K_s = 20$, 30% observation variance).

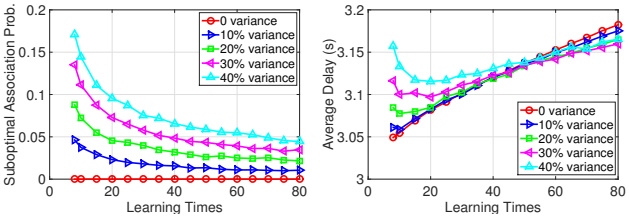
Fig. 3 shows the impact of control parameter V on the average delay and total energy consumption. By increasing V from 10^{-4} to 10, both EMM-GSI and EMM-LSI algorithms care more about the delay performance, and thus the average delay decreases. However, with less concern on the energy consumption, the total energy consumption increases and will finally exceed the given budget. The delay-energy performance follows the $[O(1/V), O(V)]$ tradeoff, which verifies Theorem 1 and Theorem 2. Meanwhile, the results also provide guidelines for selecting V in real implementations: under the energy budget constraint, one should choose appropriate V that can minimize the average delay performance.



(a) Average delay (b) Total energy consumption

Fig. 4. Impact of energy budget αB ($V = 0.01$, $K_s = 20$, 30% observation variance).

By varying the energy capping parameter α from 10% to 100%, we explore the impact of energy budget on the average delay and total energy consumption, as shown in Fig. 4. When the energy budget is large, EMM-GSI achieves the optimal delay since the energy constraint is always satisfied, while EMM-LSI incurs additional performance loss due to the learning process. When the energy budget is too low, there is possibly no feasible solution, thus the energy constraint is violated. In between, both EMM algorithms can tradeoff between the average delay and energy consumption, and the performance of EMM-GSI is very close to the J-step Lookahead.



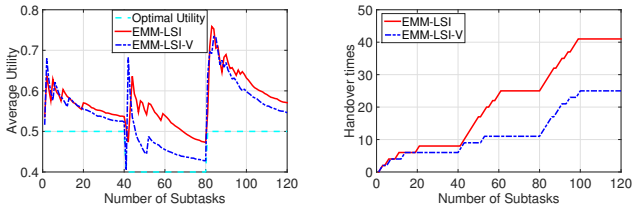
(a) Probability of connecting suboptimal BS after learning (b) Average delay

Fig. 5. Impact of learning times K_s ($V = 0.01$, $\alpha B = 410$J).

For implementation considerations, the impact of the number of subtasks K_s used for learning in EMM-LSI algorithm is further evaluated. We set K_s to vary from 8 to 80, carry out simulations under different observation variance, and repeat 10 times for average. Fig. 5(a) shows the probability of connecting to a suboptimal BS after using K_s subtasks to learn. When there is no observation variances, the user can always select the optimal BS after connecting to each available BS once. When the observation variance increases, the probability of connecting to a suboptimal BS increases. However, as K_s increase, the probability of connecting to a suboptimal BS decreases drastically. Fig. 5(b) shows the impact of K_s on the average delay. With K_s increasing, the average delay decreases first and then increases, except for the case with zero variance where learning always increases the regret. This is because when K_s is small, the probability of connecting to a suboptimal BS after learning is large, which leads to high additional cost. When K_s is large, the frequent handover increases the handover regret and thus degrades the delay performance. Therefore, learning time K_s should be carefully selected to balance the aforementioned two factors. For example, in our settings, under 30% observation variance, $K_s = 20$ can obtain the best delay performance.

TABLE I
AVAILABLE BSS AND NORMALIZED UTILITY

Index of BS	1	2	3	4	5
Normalized utility	0.5	0.8	0.4	0.9	0.7
Epoch 1	√	√	–	–	–
Epoch 2	√	√	√	√	–
Epoch 3	√	√	×	√	√



(a) Average utility (b) Handover times

Fig. 6. EMM-LSI-V algorithm vs. EMM-LSI algorithm.

Finally, we compare the proposed EMM-LSI-V algorithm with EMM-LSI under the dynamic BS set. We illustrate the results by dividing one task into 3 epochs. The available BSs and their normalized utility (defined in **P3**, which reflects both the delay performance and energy consumption) are shown in Table I. In epoch 2, there appears an optimal BS and a suboptimal BS, while in epoch 3, an optimal BS disappears and a suboptimal BS appears. Each epoch has 40 subtasks and $K_s = 20$. As shown in Fig. 6, $K_s = 40$ and $K_s = 80$ are the beginning of epoch 2 and epoch 3, thus both algorithms start to learn the environmental change and the average utility suffers sudden increases. However, the EMM-LSI-V algorithm converges faster than EMM-LSI algorithm does, while efficiently reduces the handover times. This is because EMM-LSI-V algorithm is able to retain the information of remaining BSs while EMM-LSI algorithm restarts the learning process whenever there is a change of the BS set.

VIII. CONCLUSIONS

In this paper, we studied the MM problem for MEC-enabled UDN. We developed a novel user-centric MM framework and designed MM algorithms, called EMM, that can be applied

to both GSI and LSI scenarios by integrating Lyapunov optimization and MAB techniques. Taking radio handover and computation migration cost into consideration, we proved that our proposed algorithms can optimize the delay performance while approximately satisfying the energy consumption budget of the user. Furthermore, we proposed a generalized EMM algorithm that can handle varying BS sets based on the VMAB framework. Simulations show that our proposed EMM algorithm can achieve close-to-optimal delay performance while satisfying the energy consumption constraint. Future research directions include designing MM schemes for high mobility scenarios where the user may move a lot during the processing of a task, and considering cooperative computing among BSs.

REFERENCES

[1] J. Xu, Y. Sun, L. Chen, and S. Zhou, "E2M2: Energy efficient mobility management in dense small cells with mobile edge computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017.

[2] T. Q. Quek, G. de la Roche, I. Guvenc, and M. Kountouris, *Small cell networks: Deployment, PHY techniques, and resource management*. Cambridge University Press, 2013.

[3] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young,"Mobile edge computing: A key technology towards 5G," *ETSI White Paper*, vol. 11, 2015.

[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, to be published.

[5] S. Chen and J. Zhao, "The requirements, challenges, and technologies for 5g of terrestrial mobile telecommunication," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 36–43, May 2014.

[6] ETSI. Mobile edge computing: Service scenarios. [Online]. Available: http://www.etsi.org/deliver/etsi_gs/MEC-IEG/001_099/004/01.01.01_60/gs_MEC-IEG004v010101p.pdf

[7] ETSI. Mobile edge computing: Technical requirements. [Online]. Available: http://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/01.01.01_60/gs_MEC002v010101p.pdf

[8] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sept. 2013.

[9] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Spain, Jul. 2016.

[10] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2016.

[11] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[12] D. Xenakis, N. Passas, L. Merakos, and C. Verikoukis, "Mobility management for femtocells in lte-advanced: key aspects and survey of handover decision algorithms," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 64–91, 2014.

[13] D. Lopez-Perez, I. Guvenc, and X. Chu, "Mobility management challenges in 3GPP heterogeneous networks," *IEEE Commun. Mag.*, vol. 50, no. 12, Dec. 2012.

[14] J. Park, S. Y. Jung, S. L. Kim, M. Bennis, and M. Debbah, "User-centric mobility management in ultra-dense cellular networks under spatio-temporal dynamics," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Washington, DC, USA, Dec. 2016.

[15] C. Shen, C. Tekin, and M. van der Schaar, "A non-stochastic learning approach to energy efficient mobility management," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3854–3868, Dec. 2016.

[16] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

[17] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 74-80, Feb. 2014.

[18] T. Taleb, A. Ksentini, and P. Frangoudis, "Follow-me cloud: When cloud services follow mobile users," *IEEE Trans. Cloud Comput.*, to be published.

[19] S. Wang, R. Urgaonkar, T. He, M. Zafer, K. Chan, and K. K. Leung, "Mobility-induced service migration in mobile micro-clouds," in *Proc. IEEE Military Commun. Conf. (MILCOM)*, Baltimore, MD, Oct. 2014.

[20] H. Modares, A. Moravejosharieh, J. Lloret, and R. B. Salleh, "A survey on proxy mobile ipv6 handover," *IEEE Syst. J.*, vol. 10, no. 1, pp. 208–217, Mar. 2016.

[21] M. J. Neely, *Stochastic network optimization with application to communication and queueing systems*. San Rafael, CA, USA: Morgan & Claypool Publishers, 2010.

[22] Z. Bnaya, R. Puzis, R. Stern, and A. Felner, "Social network search as a volatile multi-armed bandit problem," *HUMAN*, vol. 2, no. 2, pp. 84–98, 2013.

[23] S. Batabyal and P. Bhaumik, "Mobility models, traces and impact of mobility on opportunistic routing algorithms: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1679–1707, 2015.

[24] A. Anjum, T. Abdullah, M. Tariq, Y. Baltaci, and N. Antonopoulos, "Video stream analysis in clouds: An object detection and classification framework for high performance video analytics," *IEEE Trans. Cloud Comput.*, to be published.

[25] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, San Francisco, CA, USA, Jun. 2010.

[26] S. Zhang, J. Gong, S. Zhou, and Z. Niu, "How many small cells can be turned off via vertical offloading under a separation architecture?" *IEEE Trans. Wireless Commun.*, vol. 14, no. 10, pp. 5440–5453, Oct. 2015.

[27] Y. Sun, S. Zhou, and J. Xu, "EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks." [Online]. Available: https://arxiv.org/abs/1709.02582

[28] R. Agrawal, M. Hedge, and D. Teneketzis, "Asymptotically efficient adaptive allocation rules for the multiarmed bandit problem with switching cost," *IEEE Trans. Autom. Control*, vol. 33, no. 10, pp. 899–906, 1988.

[29] C. Niu, Y. Li, R. Q. Hu, and F. Ye, "Fast and efficient radio resource allocation in dynamic ultra-dense heterogeneous networks," *IEEE Access*, vol. 5, pp. 1911–1924, 2017.

[30] T. 36.331, "Radio resource control (RRC); protocol specification," *3GPP TS 36.331*, May 2017.

[31] C. Shen and M. van der Schaar, "A learning approach to frequent handover mitigations in 3gpp mobility protocols," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, San Francisco, CA, USA, Mar. 2017.

Yuxuan Sun received her B.S. degree in telecommunications engineering from Tianjin University, Tianjin, China, in 2015. She is currently pursuing the Ph.D. degree in Electronic Engineering at Tsinghua University, Beijing, China. Her research interests include mobile edge computing and vehicular cloud computing.

Sheng Zhou is an Associate Professor in Electronic Engineering Department, Tsinghua University, China. He received the B.S. and Ph.D. degrees in Electronic Engineering from Tsinghua University in 2005 and 2011, respectively. His research interests include cross-layer design for multiple antenna systems, mobile edge computing, and green wireless communications.

Jie Xu is an Assistant Professor in Electrical and Computer Engineering Department at the University of Miami. He received the B.S. and M.S. degrees in Electronic Engineering from Tsinghua University, Beijing, China, in 2008 and 2010, respectively and the Ph.D. degree in Electrical Engineering from UCLA in 2015. His primary research interests include mobile edge computing, energy-efficient networking, machine learning and game theory.