Dynamic Service Migration in Mobile Edge Computing Based on Markov Decision Process

Article *in* IEEE/ACM Transactions on Networking · May 2019

DOI: 10.1109/TNET.2019.2916577

Shiqiang Wang
IBM
64 PUBLICATIONS **837** CITATIONS
SEE PROFILE

Ting He
Pennsylvania State University
114 PUBLICATIONS **1,422** CITATIONS
SEE PROFILE

Dynamic Service Migration in Mobile Edge Computing Based on Markov Decision Process

Shiqiang Wang, Rahul Urgaonkar, Murtaza Zafer, Ting He, Kevin Chan, Kin K. Leung

***Abstract*—In mobile edge computing, local edge servers can host cloud-based services, which reduces network overhead and latency but requires service migrations as users move to new locations. It is challenging to make migration decisions optimally because of the uncertainty in such a dynamic cloud environment. In this paper, we formulate the service migration problem as a Markov Decision Process (MDP). Our formulation captures general cost models and provides a mathematical framework to design optimal service migration policies. In order to overcome the complexity associated with computing the optimal policy, we approximate the underlying state space by the distance between the user and service locations. We show that the resulting MDP is exact for uniform one-dimensional user mobility while it provides a close approximation for uniform two-dimensional mobility with a constant additive error. We also propose a new algorithm and a numerical technique for computing the optimal solution which is significantly faster than traditional methods based on standard value or policy iteration. We illustrate the application of our solution in practical scenarios where many theoretical assumptions are relaxed. Our evaluations based on real-world mobility traces of San Francisco taxis show superior performance of the proposed solution compared to baseline solutions.**

***Index Terms*—Mobile edge computing (MEC), Markov decision process (MDP), mobility, optimization**

I. Introduction

Mobile applications that utilize cloud computing technologies have become increasingly popular over the recent years, with examples including data streaming, real-time video processing, social networking, etc. Such applications generally consist of a front-end component running on the mobile device and a back-end component running on the cloud [3], where the cloud provides additional data processing and computational capabilities. With this architecture, it is possible to access complex services from handheld devices that have limited processing power. However, it also introduces new challenges including increased network overhead and access delay to services.

Mobile edge computing (MEC) has recently emerged as a promising technique to address these challenges by moving computation closer to users [4]–[6]. In MEC, a small number of servers or micro data-centers that can host cloud applications are distributed across the network and connected directly to entities (such as cellular base stations or wireless access points) at the network edge, as shown in Fig. 1. MEC can significantly reduce the service access delay [4], thereby enabling newly emerging delay-sensitive and data-intensive mobile applications such as augmented reality (AR) and virtual reality (VR) [7]. This idea received significant academic and commercial interest recently [8], [9]. MEC is also more robust than traditional centralized cloud computing systems [10], because the edge servers (ES) are distributed and are thus less impacted by failures at a centralized location. The MEC concept is also known as cloudlet [10], edge cloud [11], fog computing [12], follow me cloud [13], micro cloud [14], [15], and small cell cloud [16].

One new problem that MEC brings in is dynamic service placement and migration. As a user moves across different geographical locations, its service may need to be migrated to follow the user so that the benefits of MEC are maintained. The question is *when and where to migrate the service*. Migrating a service may incur service interruption and network overhead, whereas not migrating a service may increase the data transmission delay between the user and the ES that hosts its service when the user moves away from its original location. It is challenging to make an optimal migration decision because of the uncertainty in user mobility as well as the complex trade-off between the "costs" related to migration and distant data transmission.

The performance of MEC in the presence of user mobility is first studied in [17] using a Markovian mobility model, but decisions on whether and where to migrate the service are

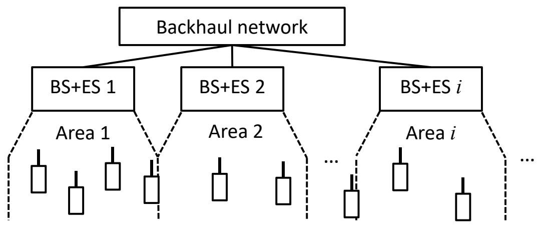


Figure 1. Application scenario of mobile edge computing where edge servers (ES) are co-located with base stations (BS).

This research was sponsored in part by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001 and W911NF-16-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

S. Wang is with IBM T. J. Watson Research Center, Yorktown Heights, NY, United States, Email: wangshiq@us.ibm.com

R. Urgaonkar is with Amazon Inc., Seattle, WA, United States. Email: rahul.urgaonkar@gmail.com

M. Zafer is with Nyansa Inc., Palo Alto, CA, United States, Email: murtaza.zafer@gmail.com

T. He is with Pennsylvania State University, University Park, PA, USA. Email: t.he@cse.psu.edu

K. Chan is with Army Research Laboratory, Adelphi, MD, United States, Email: kevin.s.chan.civ@mail.mil

K. K. Leung is with the Department of Electrical and Electronic Engineering, Imperial College London, United Kingdom, Email: kin.leung@imperial.ac.uk

This paper has been accepted for publication in the IEEE/ACM Transactions on Networking. A preliminary version of this paper was presented at IFIP Networking 2015 [1]. Part of this work also appeared in S. Wang's Ph.D. thesis [2].

not considered. A preliminary work on mobility-driven service migration based on Markov Decision Processes (MDPs) is given in [18], which mainly considers one-dimensional (1-D) mobility patterns with a specific cost function. Standard solution procedures are used to solve this MDP, which can be time consuming when the MDP has a large number of states. Because the cost functions and transition probabilities of the MDP may change over time and the ES processing power is limited, it is desirable to solve the MDP in an effective manner. With this motivation, a more efficient solution to the 1-D mobility case was proposed in [19], where the transmission and migration costs are assumed to be constant whenever transmission/migration occurs. To the best of our knowledge, two-dimensional (2-D) mobility in an MDP setting of the service migration problem has not been considered in the literature, which is a much more realistic case compared to 1-D mobility and we consider it in this paper.

In this paper, we use the MDP framework to study service migration in MEC. We provide novel contributions beyond [18] and [19], by considering general cost models, 2-D user mobility, and application to real-world mobility traces. We focus on the case where ESs are co-located with base stations (BS)[1] in this paper, which is a possible configuration option according to a recently established MEC specification group [20] and this setting has also been proposed for commercial products [9]. However, our proposed solution is not restricted to such cases and can be easily extended to more general scenarios as long as the costs are location-dependent (see Section III-A for cost definitions). Our main contributions are summarized as follows.

1) Our formulation captures general cost models and provides a mathematical framework to design optimal service migration policies. We note that the resulting problem becomes difficult to solve due to the large state space. In order to overcome this challenge, we propose an approximation of the underlying state space by defining the states as the *distance* between the user and the service locations[2]. This approximation becomes exact for uniform 1-D mobility[3]. We prove several structural properties of the distance-based MDP, which includes a closed-form solution to the discounted sum cost. We leverage these properties to develop an algorithm for computing the optimal policy, which reduces the complexity from $O(N^3)$ (by policy iteration [21, Chapter 6]) to $O(N^2)$, where the number of states in the distance-based MDP is $N+1$.
2) We show how to use the distance-based MDP to approximate the solution for 2-D mobility models, which allows us to efficiently compute a service migration policy for 2-D mobility. For the uniform 2-D mobility, the approximation error is bounded by a constant. Simulation results comparing our approximation solution to the optimal solution (where the optimal solution is obtained from a 2-D MDP directly) suggest that the proposed approach performs very close to optimal and obtains the solution much faster.
3) We demonstrate how to apply our algorithms in a practical scenario driven by real mobility traces of taxis in San Francisco which involve multiple users and services. The practical scenario includes realistic factors, e.g., not every BS has an ES attached to it, and each ES can only host a limited number of services. We compare the proposed policy with several baseline strategies that include myopic, never-migrate, and always-migrate policies. It is shown that the proposed approach offers significant gains over these baseline approaches.

[1]The notion of base station (BS) in this paper can refer to a cellular tower, a wireless access point, or any physical entity that can have an ES attached to it. We do not distinguish among them for simplicity.

[2]Throughout this paper, we mean by *user location* the location of the BS that the user is associated to.

[3]The 1-D mobility is an important practical scenario often encountered in transportation networks, such as vehicles moving along a road.

The remainder of this paper is organized as follows. Section II summarizes the related work. Section III describes the problem formulation. The distance-based MDP model and its optimal policy is discussed in Section IV. Section V focuses on using the distance-based MDP to solve problems with 2-D mobility. Section VI discusses the application to real-world scenarios. Section VII provides some additional discussions and Section VIII draws conclusions.

II. Related Work

Existing work on service migration focuses on workload and energy patterns that vary slowly [22], [23]. In MEC, user mobility is the driving factor of migration, which varies much faster than parameters in conventional clouds.

Service migration in MEC has been studied in an MDP setting in [18], [19], as mentioned earlier. Besides using the MDP framework, it has also been studied in other settings very recently. The work in [14] relies on a separate prediction module that can predict the future costs of each individual user running its service in every possible ES. A migration mechanism based on user mobility prediction utilizing low level channel information was proposed in [24]. Perfect knowledge of user mobility within a given time frame is assumed in [25]. These approaches are difficult to implement in practice because they require a detailed tracking of every user over a long time duration and may also require access to physical-layer information. In contrast, the approach we propose in this paper only requires knowledge on the number of users at and leaving a BS in each timeslot.

Other work on MEC service migration assumes no knowledge on user mobility, but their applicable scenarios are more limited. In [26], an online algorithm was proposed for service migration from a remote cloud to an ES, where only a single ES is considered and does not apply to cases where users move across areas close to different ESs. An online resource allocation and migration mechanism was proposed in [27], where it is assumed that the computational workloads (code and data) are "fluid" and can be split up into infinitely small pieces. This assumption generally does not hold in practice, because usually a computer program can only be separated in a small number of ways and needs to have a minimal size of data to run. The work in [28] considers non-realtime applications that allow the queueing of user requests before they are served. This is not applicable for steaming applications such as real-time AR/VR. In this paper, we consider the case where each user continuously accesses its service without queueing, and focus on user mobility and realistic fix-sized computational entities.

Due to the importance of service migration triggered by user and system dynamics in MEC, recent work has also focused on the implementation of service migration mechanisms. Effective migration mechanisms of virtual machines and containers in an MEC environment were proposed in [29], [30], which did not study the decision making of when/where to migrate and imply an "always migrate" mechanism where the service always follows the user. These migration methods can work together with migration decision algorithms, such as the one we propose in this paper, as suggested in [11]. Other work focuses on developing protocols for MEC service migration [7], [31]. In [31], a simple thresholding method for migration decision making, which only looks at the system state at the current time (thus "myopic"), was proposed, while suggesting that other migration decision algorithms can be plugged into their framework as well. In [7], a standard MDP approach where the state space is polynomial in the total number of BS and ES was applied for migration decision making, which can become easily intractable when the number of BS and ES is large. It is also mentioned in [7] that it is important to reduce the complexity of finding migration decisions. The complexity of our proposed approach in this paper does *not* depend on the number of BS and ES, and the state space of the MDP in this paper is much smaller than that in [7].

A related area of work relevant to user mobility studies handover policies in the context of cellular networks [32]. However, the notion of service migration is very different from cellular handover. Handover is usually decided by signal strengths from different BSs, and a handover must occur if a user's signal is no longer provided satisfactorily by its original BS. In the service migration context, a user may continue receiving service from an ES even if it is no longer associated with that BS, because the user can communicate with a remote ES via its currently associated BS and the backhaul network. As a result, the service for a particular user can potentially be placed on any ES, and the service migration problem has a much larger decision space than the cellular handover problem.

III. Problem Formulation

Consider a mobile user in a 2-D geographical area that accesses a cloud-based service hosted on the ESs. The set of possible locations is given by $\mathcal{L}$, where $\mathcal{L}$ is assumed to be finite (but arbitrarily large). We consider a time-slotted model (see Fig. 2) where the user's location remains fixed for the duration of one slot and changes from one slot to the next according to a Markovian mobility model. The time-slotted model can be regarded as a sampled version of a continuous-time model, and the sampling can be performed either at equal or non-equal intervals over time. In addition, we assume that each location $l \in \mathcal{L}$ is associated with an ES that can host the service for the user (this assumption will be relaxed in Section VI). The locations in $\mathcal{L}$ are represented as 2-D vectors and there exists a distance metric $\|l_1 - l_2\|$ that can be used to calculate the distance between locations l_1 and l_2. Note that the distance metric may not be Euclidean distance. An example of this model is a cellular network in which the user's location is considered as the location of its current BS and the ESs are co-located with the BSs. As shown in Section V, these locations can be represented as 2-D vectors (i, j) with respect to a reference location (represented by $(0, 0)$) and the distance between any two locations can be calculated in terms of the number of hops to reach from one cell to another cell. We denote the user and service locations at timeslot t as $u(t)$ and $h(t)$ respectively.

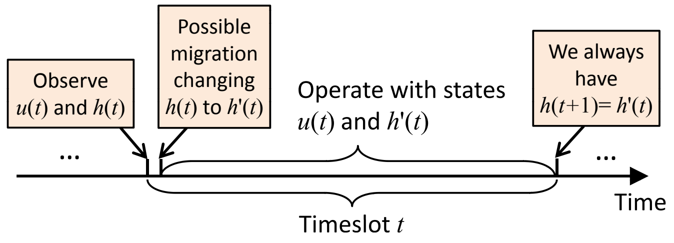

Observe $u(t)$ and $h(t)$
Possible migration changing $h(t)$ to $h'(t)$
Operate with states $u(t)$ and $h'(t)$
We always have $h(t+1)= h'(t)$
...
...
Time
Timeslot t

Figure 2. Timing of the proposed service migration mechanism.

Remark: Although the problem is formulated for the case of a single user accessing a single service, our solution can be applied to manage services for multiple users. We will illustrate such an application in Section VI, where we also consider aspects including that ESs are only deployed at a subset of BSs and a limited number of services can be hosted at each ES. We assume in this paper that different services are independent of each other, and one service serves a single user. The notion of "service" can also stand for an instance of a particular type of service, but we do not distinguish between services and instances in this paper for simplicity.

Table I summarizes the main notations in this paper.

A. Control Decisions and Costs

At the beginning of each slot, the MEC controller can choose from one of the following control options:

1) Migrate the service from location $h(t)$ to some other location $h'(t) \in \mathcal{L}$. This incurs a *migration cost* $b(x)$ that is assumed to be a non-decreasing function of x, where x is the distance between $h(t)$ and $h'(t)$, i.e., $x = \|h(t) - h'(t)\|$. Once the migration is completed, the system operates under state $(u(t), h'(t))$. The migration cost can capture the service interruption time of the migration process, as recent experimental work has shown that a non-zero interruption time exists whenever a migration occurs (i.e., the migration distance is non-zero) [11], [29], [30]. The interruption time can increase with the migration distance due to increased propagation and switching delays of data transmission.
2) Do not migrate the service. In this case, we have $h'(t) = h(t)$ and the migration cost is $b(0) = 0$.

In addition to the migration cost, there is a data *transmission cost* incurred by the user for connecting to the currently active service instance. The transmission cost is related to the distance between the service and the user after possible migration,

Table I
SUMMARY OF MAIN NOTATIONS

Notation	Description
$\triangleq$	Is defined to be equal to
$\mathcal{L}$	Set of locations
l	Location
$\|l_1 - l_2\|$	Distance between locations l_1 and l_2
$u(t)$	User location at timeslot t
$h(t)$	Service location at timeslot t
$b(x)$	Migration cost
$c(y)$	Transmission cost
$s(t)$	Initial state at slot t
$s'(t)$	Intermediate state at slot t
π	Decision policy
$a(s)$ $(a^*(s))$	(Optimal) action taken when system is in state $s(t)$
$C_a(s)$	Sum of migration and transmission costs when taking action $a(s)$ in slot t
$V(s_0)$	Discounted sum cost when starting at state s_0
$P[s'_0, s_1]$	Transition probability from intermediate state s'_0 to the next initial state s_1 (in the next slot)
γ	Discount factor of the MDP
$d(t)$	User-service distance in slot t before possible migration (state in the distance-based MDP)
N	Number of states (excluding state zero) in the distance-based MDP
p_0, p, q	Transition probabilities of the distance-based MDP (see Fig. 3)
$\beta_c, \beta_l, \delta_c,$ δ_l, μ, θ	Parameters related to the constant-plus-exponential cost function (see (4) and (5))
$A_k, B_k, D,$ H, m_1, m_2	Parameters related to the closed-form solution of the distance-based MDP (see (7)–(10))
$\{n_k : k \geq 0\}$	Series of migration states (i.e., all n_k such that $a(n_k) \neq n_k$)
r	Transition probability to one of the neighbors in the 2-D model
$e(t)$	Offset of the user from the service as a 2-D vector (state in the 2-D offset-based MDP)

and it is defined as a general non-decreasing function $c(y)$, where $y = \|u(t) - h'(t)\|$. The transmission cost can capture the delay of data transmission, where a high delay increases the service response time. As discussed in [4], [25], [29], [33], the delay is usually a function of the geographical or topological distance between two nodes and it increases with the distance. We set $c(0) = 0$.

We assume that the transmission delay and the service interruption time caused by migration is much smaller than the length of each timeslot (see Fig. 2), thus the costs do not change with the timeslot length.

B. Performance Objective

Let us denote the overall system state at the beginning of each timeslot (before possible migration) by $s(t) = (u(t), h(t))$. The state $s(t)$ is named as the *initial state* of slot t. Consider any policy[4] π that makes control decisions based on the state $s(t)$ of the system, and we use $a_\pi(s(t))$ to represent the control action taken when the system is in state $s(t)$. This action causes the system to transition to a new *intermediate state* $s'(t) = (u(t), h'(t)) = a_\pi(s(t))$. We use $C_{a_\pi}(s(t))$ to denote the sum of migration and transmission costs incurred by a control $a_\pi(s(t))$ in slot t, and we have $C_{a_\pi}(s(t)) = b(\|h(t) - h'(t)\|) + c(\|u(t) - h'(t)\|)$. Starting from any initial state $s(0) = s_0$, the long-term expected *discounted sum cost* incurred by policy π is given by

$$V_\pi(s_0) = \lim_{t\to\infty} \mathbb{E}\left\{\sum_{\tau=0}^{t} \gamma^\tau C_{a_\pi}(s(\tau)) \middle| s(0) = s_0\right\} \quad (1)$$

where $0 < \gamma < 1$ is a discount factor. Note that we consider deterministic policies in this paper and the expectation is taken over random user locations.

Our objective is to design a control policy that minimizes the long-term expected discounted sum total cost starting from any initial state, i.e.,

$$V^*(s_0) = \min_\pi V_\pi(s_0) \quad \forall s_0. \quad (2)$$

This problem falls within the class of MDPs with infinite horizon discounted cost. It is well known that the optimal solution is given by a stationary policy[5] and can be obtained as the unique solution to the Bellman's equation [21]:

$$V^*(s_0) = \min_a \left\{ C_a(s_0) + \gamma \sum_{s_1 \in \mathcal{L}\times\mathcal{L}} P[a(s_0), s_1] \cdot V^*(s_1) \right\} \quad (3)$$

where $P[a(s_0), s_1]$ denotes the probability of transitioning from state $s'(0) = s'_0 = a(s_0)$ to $s(1) = s_1$. Note that the intermediate state $s'(t)$ has no randomness when $s(t)$ and $a(\cdot)$ are given, thus we only consider the transition probability from $s'(t)$ to the next state $s(t+1) = (u(t+1), h'(t)) = (u(t+1), h(t+1))$ in (3), where we note that we always have $h(t+1) = h'(t)$.

C. Characteristics of Optimal Policy

We next characterize some structural properties of the optimal solution. The following theorem states that it is not optimal to migrate the service to a location that is farther away from the user than the current service location, as one would intuitively expect.

Theorem 1. *Let $a^*(s) = (u, h')$ denote the optimal action at any state $s = (u, h)$. Then, we have $\|u - h'\| \leq \|u - h\|$. (If the optimal action is not unique, then there exists at least one such optimal action.)*

Proof. See Appendix A. □

Corollary 1. *If $b(x)$ and $c(y)$ are both* constants *(possibly of different values) for $x, y > 0$, and $b(0) < b(x)$ and $c(0) < c(y)$ for $x, y > 0$, then migrating to locations other than the current location of the mobile user is not optimal.*

Proof. See Appendix B. □

D. Simplifying the Search Space

Theorem 1 simplifies the search space for the optimal policy considerably. However, it is still very challenging to derive the optimal control policy for the general model presented

[4]A policy represents a decision rule that maps a state to a new state while (possibly) incurring a cost.

[5]A stationary policy is a policy where the same decision rule is used in each timeslot.

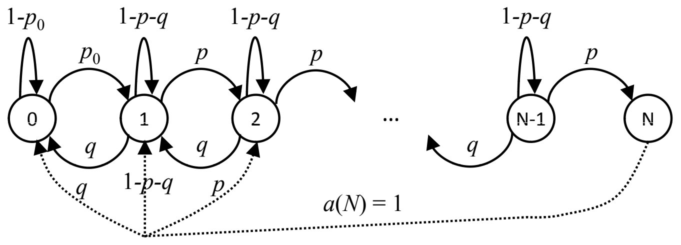


Figure 3. Distance-based MDP with the distances $\{d(t)\}$ (before possible migration) as states. In this example, migration is only performed at state N, and only the possible action of $a(N) = 1$ is shown for compactness. The solid lines denote state transitions without migration.

above, particularly when the state space $\{s(t)\}$ is large. One possible approach to address this challenge is to re-define the state space to represent only the *distance* between the user and service locations $d(t) = \|u(t) - h(t)\|$. The motivation for this comes from the observation that the cost functions in our model only depend on the distance. Note that in general, the optimal control actions can be different for two states s_0 and s_1 that have the same user-service distance. However, it is reasonable to use the distance as an approximation of the state space for many practical scenarios of interest, and this simplification allows us to formulate a far more tractable MDP. We discuss the distance-based MDP in the next section, and show how the results on the distance-based MDP can be applied to 2-D mobility and real-world scenarios in Sections V and VI.

In the remainder of this paper, where there is no ambiguity, we reuse the notations P, $C_a(\cdot)$, $V(\cdot)$, and $a(\cdot)$ to respectively represent transition probabilities, one-timeslot costs, discounted sum costs, and actions of different MDPs.

IV. Optimal Policy for Distance-Based MDP

In this section, we consider a distance-based[6] MDP where the states $\{d(t)\}$ represent the distances between the user and the service before possible migration (an example is shown in Fig. 3), i.e., $d(t) = \|u(t) - h(t)\|$. We define the parameter N as an application-specific maximum allowed distance, and we always perform migration when $d(t) \geq N$. We set the actions $a(d(t)) = a(N)$ for $d(t) > N$, so that we only need to focus on the states $d(t) \in [0, N]$. After taking action $a(d(t))$, the system operates in the intermediate state $d'(t) = a(d(t))$, and the value of the next state $d(t+1)$ follows the transition probability $P[d'(t), d(t+1)]$ which is related to the mobility model of the user. To simplify the solution, we restrict the transition probabilities $P[d'(t), d(t+1)]$ according to the parameters p_0, p, and q as shown in Fig. 3. Such a restriction is sufficient when the underlying mobility model is a uniform 1-D random walk where the user moves one step to the left or right with equal probability r_1 and stays in the same location with probability $1 - 2r_1$, in which case we can set $p = q = r_1$ and $p_0 = 2r_1$. This model is also sufficient to approximate the uniform 2-D random walk model, as will be discussed in Section V-B.

For an action of $d'(t) = a(d(t))$, the new service location $h'(t)$ is chosen such that $x = \|h(t) - h'(t)\| = |d(t) - d'(t)|$

[6]We assume that the distance is quantized, as it will be the case with the 2-D model discussed in later sections.

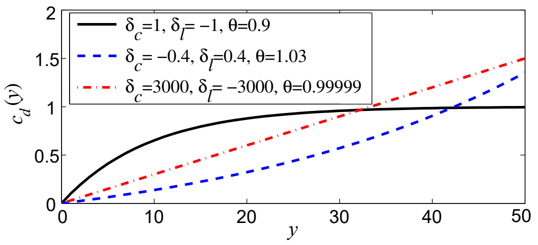


Figure 4. Example of constant-plus-exponential cost function $c(y)$.

and $y = \|u(t) - h'(t)\| = d'(t)$. This means that migration happens along the shortest path that connects $u(t)$ and $h(t)$, and $h'(t)$ is on this shortest path (also note that $d'(t) \leq d(t)$ according to Theorem 1). Such a migration is possible for the 1-D case where $u(t)$, $h(t)$, and $h'(t)$ are all scalar values. It is also possible for the 2-D case if the distance metric is properly defined (see Section V-B). The one-timeslot cost is then $C_a(d(t)) = b(|d(t) - d'(t)|) + c(d'(t))$.

A. Constant-Plus-Exponential Cost Functions

To simplify the analysis later, we define the cost functions $b(x)$ and $c(y)$ in a constant-plus-exponential form:

$$b(x) = \begin{cases} 0, & \text{if } x = 0 \\ \beta_c + \beta_l \mu^x, & \text{if } x > 0 \end{cases} \tag{4}$$

$$c(y) = \begin{cases} 0, & \text{if } y = 0 \\ \delta_c + \delta_l \theta^y, & \text{if } y > 0 \end{cases} \tag{5}$$

where β_c, β_l, δ_c, δ_l, μ, and θ are real-valued parameters.

The cost functions defined above can have different shapes and are thus applicable to many realistic scenarios (see Fig. 4). They can approximate an arbitrary cost function as discussed in Appendix C. For example, they can be defined such that there is a constant non-zero cost whenever the distance is larger than zero. Such a cost definition is applicable in systems where all ESs are connected through a single network hub, and it can also approximate cases where there is a relatively high cost whenever the distance larger than zero. The latter is found from experiments in [29]. The costs can also be defined in a way so that they are (approximately) linearly proportional to the distance, where the distance can be defined as the length of the shortest path between BSs, as in [25], [33]. They also have nice properties allowing us to obtain a closed-form solution to the discounted sum cost, based on which we design an efficient algorithm for finding the optimal policy.

The parameters β_c, β_l, δ_c, δ_l, μ, and θ are selected such that $b(x) \geq 0$, $c(y) \geq 0$, and both $b(x)$ and $c(y)$ are non-decreasing respectively in x and y for $x, y \geq 0$. Explicitly, we have $\mu \geq 0$; $\beta_l \leq 0$ when $\mu \leq 1$; $\beta_l \geq 0$ when $\mu \geq 1$; $\beta_c \geq -\beta_l$; $\theta \geq 0$; $\delta_l \leq 0$ when $\theta \leq 1$; $\delta_l \geq 0$ when $\theta \geq 1$; and $\delta_c \geq -\delta_l$. The definition that $b(0) = c(0) = 0$ is for convenience, because a non-zero cost for $x = 0$ or $y = 0$ can be offset by the values of β_c and δ_c, thus setting $b(0) = c(0) = 0$ does not affect the optimal decision.

With this definition, the values of $\beta_c + \beta_l$ and $\delta_c + \delta_l$ can be regarded as constant terms of the costs, at least such an amount of cost is incurred when $x > 0$ and $y > 0$, respectively. The parameters μ and θ specify the impact of the distance x and y, respectively, to the costs, and their values can be related to the network topology and routing mechanism of the network. The parameters β_l and δ_l further adjust the costs proportionally.

B. Closed-Form Solution to Discounted Sum Cost

1) Problem Formulation with Difference Equations: From (1), we can get the following balance equation on the discounted sum cost for a given policy π:

$$V(d) = C_a(d) + \gamma \sum_{d(1)=a(d)-1}^{a(d)+1} P[a(d), d(1)] \cdot V(d(1)) \quad (6)$$

where we omit the subscript π and write $d(0)$ as d for short (we will also follow this convention in the following).

Theorem 2. *For a given policy π, let $\{n_k : k \geq 0\}$ denote the series of all migration states (such that $a(n_k) \neq n_k$) as specified by policy π, where $0 \leq n_k \leq N$. The discounted sum cost $V(d)$ for $d \in [n_{k-1}, n_k]$ (where we define $n_{-1} \triangleq 0$ for convenience) when following policy π can be expressed as*

$$V(d) = A_k m_1^d + B_k m_2^d + D + \begin{cases} H \cdot \theta^d & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2\theta \neq 0 \\ Hd \cdot \theta^d & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2\theta = 0 \end{cases} \quad (7)$$

where A_k and B_k are constants corresponding to the interval $[n_{k-1}, n_k]$, the coefficients m_1, m_2, D, and H are expressed as

$$m_1 = \frac{1 + \sqrt{1 - 4\phi_1\phi_2}}{2\phi_2}, m_2 = \frac{1 - \sqrt{1 - 4\phi_1\phi_2}}{2\phi_2} \quad (8)$$

$$D = \phi_3 / (1 - \phi_1 - \phi_2) \quad (9)$$

$$H = \begin{cases} \frac{\phi_4}{1 - \frac{\phi_1}{\theta} - \phi_2\theta} & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2\theta \neq 0 \\ \frac{\phi_4}{\frac{\phi_1}{\theta} - \phi_2\theta} & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2\theta = 0 \end{cases} \quad (10)$$

where we define $\phi_1 \triangleq \frac{\gamma q}{1-\gamma(1-p-q)}$, $\phi_2 \triangleq \frac{\gamma p}{1-\gamma(1-p-q)}$, $\phi_3 \triangleq \frac{\delta_c}{1-\gamma(1-p-q)}$, and $\phi_4 \triangleq \frac{\delta_l}{1-\gamma(1-p-q)}$.

Proof. The proof is based on solving a difference equation [34] according to (6), see Appendix D for details. □

We also note that for two different states d_1 and d_2, if policy π has actions $a(d_1) = d_2$ and $a(d_2) = d_2$, then

$$V(d_1) = b(|d_1 - d_2|) + V(d_2). \quad (11)$$

2) Finding the Coefficients: The coefficients A_k and B_k are unknowns in the solution (7) that need to be found using additional constraints. Their values may be different for different k. After A_k and B_k are determined, (7) holds for all $d \in [0, N]$.

We assume $1 - \frac{\phi_1}{\theta} - \phi_2\theta \neq 0$ and $1 - \frac{\phi_2}{\theta} - \phi_1\theta \neq 0$ in the following, the other cases can be derived in a similar way and are omitted for brevity.

Coefficients for interval $[0, n_0]$: We have one constraint from the balance equation (6) for $d = 0$, which is

$$V(0) = \gamma p_0 V(1) + \gamma(1 - p_0)V(0). \quad (12)$$

By substituting (7) into (12), we get

$$A_0(1 - \phi_0 m_1) + B_0(1 - \phi_0 m_2) = D(\phi_0 - 1) + H(\phi_0\theta - 1) \quad (13)$$

where $\phi_0 \triangleq \frac{\gamma p_0}{1-\gamma(1-p_0)}$. We have another constraint by substituting (7) into (11), which gives

$$A_0\left(m_1^{n_0} - m_1^{a(n_0)}\right) + B_0\left(m_2^{n_0} - m_2^{a(n_0)}\right) = \beta_c + \beta_l\mu^{n_0 - a(n_0)} - H\left(\theta^{n_0} - \theta^{a(n_0)}\right). \quad (14)$$

We can find A_0 and B_0 from (13) and (14).

Coefficients for interval $[n_{k-1}, n_k]$: Assume that we have found $V(d)$ for all $d \leq n_{k-1}$. By letting $d = n_{k-1}$ in (7), we have the first constraint given by

$$A_k m_1^{n_{k-1}} + B_k m_2^{n_{k-1}} = V(n_{k-1}) - D - H \cdot \theta^{n_{k-1}}. \quad (15)$$

For the second constraint, we consider two cases. If $a(n_k) \leq n_{k-1}$, then

$$A_k m_1^{n_k} + B_k m_2^{n_k} = \beta_c + \beta_l\mu^{n_k - a(n_k)} + V(a(n_k)) - D - H \cdot \theta^{n_k}. \quad (16)$$

If $n_{k-1} < a(n_k) \leq n_k - 1$, then

$$A_k\left(m_1^{n_k} - m_1^{a(n_k)}\right) + B_k\left(m_2^{n_k} - m_2^{a(n_k)}\right) = \beta_c + \beta_l\mu^{n_k - a(n_k)} - H\left(\theta^{n_k} - \theta^{a(n_k)}\right). \quad (17)$$

The values of A_k and B_k can be solved from (15) together with either (16) or (17).

3) Solution is in Closed-Form: We note that A_0 and B_0 can be expressed in closed-form, and A_k and B_k for all k can also be expressed in closed-form by substituting (7) into (15) and (16) where needed. Thus, (7) is a *closed-form solution* for all $d \in [0, N]$. Numerically, we can find $V(d)$ for all $d \in [0, N]$ in $O(N)$ time.

C. Algorithm for Finding the Optimal Policy

Standard approaches of solving for the optimal policy of an MDP include value iteration and policy iteration [21, Chapter 6]. Value iteration finds the optimal policy from the Bellman's equation (3) iteratively, which may require a large number of iterations before converging to the optimal result. Policy iteration generally requires a smaller number of iterations, because, in each iteration, it finds the exact values of the discounted sum cost $V(d)$ for the policy resulting from the previous iteration, and performs the iteration based on the exact $V(d)$ values. However, in general, the $V(d)$ values are found by solving a system of linear equations, which has a complexity of $O(N^3)$ when using Gaussian-elimination.

We propose a modified policy-iteration approach for finding the optimal policy, which uses the above result instead of Gaussian-elimination to compute $V(d)$, and also only checks for migrating to lower states or not migrating (according to Theorem 1). The algorithm is shown in Algorithm 1, where Lines 4–7 find the values of n_k, Lines 8–17 find the discounted sum cost values, and Lines 18–20 update the optimal policy. The overall complexity for each iteration is $O\left(N^2\right)$

Algorithm 1: Modified policy-iteration algorithm based on difference equations

```
1  Initialize $a(d) \leftarrow 0$ for all $d = 0, 1, 2, ..., N$;
2  Find constants $\phi_0$, $\phi_1$, $\phi_2$, $\phi_3$, $\phi_4$, $m_1$, $m_2$, $D$, and $H$;
3  repeat
4      $k \leftarrow 0$;
5      for $d = 1...N$ do
6          if $a(d) \neq d$ then
7              $n_k \leftarrow d$, $k \leftarrow k + 1$;
8      for all $n_k$ do
9          if $k = 0$ then
10             Solve for $A_0$ and $B_0$ from (13) and (14);
11             Find $V(d)$ with $0 \leq d \leq n_k$ from (7) with $A_0$ and $B_0$ found above;
12         else if $k > 0$ then
13             if $a(n_k) \leq n_{k-1}$ then
14                 Solve for $A_k$ and $B_k$ from (15) and (16);
15             else
16                 Solve for $A_k$ and $B_k$ from (15) and (17);
17             Find $V(d)$ with $n_{k-1} < d \leq n_k$ from (7) with $A_k$ and $B_k$ found above;
18     for $d = 1...N$ do
19         $a_{\text{prev}}(d) \leftarrow a(d)$;
20         $a(d) \leftarrow \arg\min_{a \leq d} \left\{ C_a(d) + \gamma \sum_{j=a-1}^{a+1} P[a, j] \cdot V(j) \right\}$;
21 until $a_{prev}(d) = a(d)$ for all $d$;
22 return $a^*(d) \leftarrow a(d)$ for all $d$;
```

in Algorithm 1, which reduces complexity because standard[7] policy iteration has complexity $O(N^3)$, and the standard value iteration approach does not compute the exact value function in each iteration and generally has long convergence time.

V. Approximate Solution for 2-D Mobility

In this section, we show that the distance-based MDP can be used to find a near-optimal service migration policy, where the user conforms to a uniform 2-D random walk mobility model in an infinite space. This mobility model can be used to approximate real-world mobility traces (see Section VI). We consider a hexagonal cell structure, but the approximation procedure can be extended to other 2-D mobility models (such as Manhattan grid) after modifications. The user is assumed to transition to one of its six neighbors at the beginning of each timeslot with probability r, and stay in the same cell with probability $1 - 6r$.

A. *Offset-Based MDP*

Define the *offset* of the user from the service as a 2-D vector $e(t) = u(t) - h(t)$ (recall that $u(t)$ and $h(t)$ are also 2-D vectors). Due to the space-homogeneity of the mobility model, it is sufficient to model the state of the MDP by $e(t)$ rather than $s(t)$. The distance metric $\|l_1 - l_2\|$ is defined as the minimum number of hops that are needed to reach from cell l_1 to cell l_2 on the hexagon model.

We name the states with the same value of $\|e(t)\|$ as a *ring*, and express the states $\{e(t)\}$ with polar indices (i, j), where the first index i refers to the ring index, and the second index j refers to each of the states within the ring, as shown in Fig. 5. For $e(t) = (i, j)$, we have $\|e(t)\| = i$. If $u(t) = h(t)$ (i.e., the actual user and service locations (cells) are the same), then we have $e(t) = (0, 0)$ and $\|e(t)\| = 0$.

[7]We use the term "standard" here to distinguish with the modified policy iteration mechanism proposed in Algorithm 1.

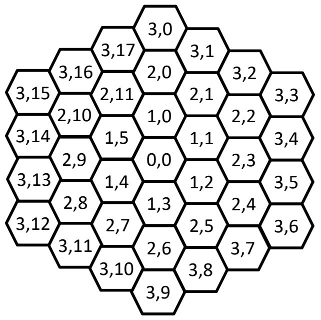


Figure 5. Example of 2-D offset model on hexagon cells, where $N = 3$.

Similarly as in the distance-based MDP, we assume in the 2-D MDP that we always migrate when $\|e(t)\| \geq N$, where N is a design parameter, and we only consider the state space $\{e(t)\}$ with $\|e(t)\| \leq N$. The system operates in the intermediate state $e'(t) = u(t) - h'(t) = a(e(t))$ after taking action $a(e(t))$. The next state $e(t+1)$ is determined probabilistically according to the transition probability $P[e'(t), e(t+1)]$. We have $P[e'(t), e(t+1)] = 1 - 6r$ when $e(t+1) = e'(t)$; $P[e'(t), e(t+1)] = r$ when $e(t+1)$ is a neighbor of $e'(t)$; and $P[e'(t), e(t+1)] = 0$ otherwise. Note that we always have $e(t) - e'(t) = h'(t) - h(t)$, so the one-timeslot cost is $C_a(e(t)) = b(\|e(t) - e'(t)\|) + c(\|e'(t)\|)$.

We note that, even after simplification with the offset model, the 2-D offset-based MDP has a significantly larger number of states compared with the distance-based MDP, because for a distance-based model with N states (excluding state zero), the 2-D offset model has $M = 3N^2 + 3N$ states (excluding state $(0, 0)$). Therefore, we use the distance-based MDP proposed in Section IV to approximate the 2-D offset-based MDP, which significantly reduces the computational time as shown in Section V-D.

B. *Approximation by Distance-based MDP*

In the approximation, the parameters of the distance-based MDP are chosen as $p_0 = 6r$, $p = 2.5r$, and $q = 1.5r$. The intuition of the parameter choice is that, at state $(i'_0, j'_0) = (0, 0)$ in the 2-D MDP, the aggregate probability of transitioning to any state in ring $i_1 = 1$ is $6r$, so we set $p_0 = 6r$; at any other state $(i'_0, j'_0) \neq (0, 0)$, the aggregate probability of transitioning to any state in the higher ring $i_1 = i'_0 + 1$ is either $2r$ or $3r$, and the aggregate probability of transitioning to any state in the lower ring $i_1 = i'_0 - 1$ is either r or $2r$, so we set p and q to the median value of these transition probabilities.

To find the optimal policy for the 2-D MDP, we first find the optimal policy for the distance-based MDP with the parameters defined above. Then, we map the optimal policy from the distance-based MDP to a policy for the 2-D MDP. To explain this mapping, we note that, in the 2-D hexagon offset model, there always exists at least one shortest path from any state (i, j) to an arbitrary state in ring i', the length of this shortest path is $|i - i'|$, and each ring between i and i' is traversed once on the shortest path. For example, one shortest path from state $(3, 2)$ to ring $i' = 1$ is $\{(3, 2), (2, 1), (1, 0)\}$. When the system is in state (i, j) and the optimal action from the distance-based MDP is $a^*(i) = i'$, we perform migration on the shortest path from (i, j) to ring i'. If there exist multiple shortest paths, one path is arbitrarily chosen. For example, if $a(3) = 2$ in the distance-based MDP, then we have either $a(3, 2) = (2, 1)$ or $a(3, 2) = (2, 2)$ in the 2-D MDP. With this mapping, the one-timeslot cost $C_a(d(t))$ for the distance-based MDP and the one-timeslot cost $C_a(e(t))$ for the 2-D MDP are the same, because the migration distances in the distance-based MDP and 2-D MDP are the same (thus same migration cost) and all states in the same ring $i' = \|e'(t)\| = d'(t)$ have the same transmission cost $c(\|e'(t)\|) = c(d'(t))$.

C. Bound on Approximation Error

Error arises from the approximation because the transition probabilities in the distance-based MDP are not exactly the same as that in the 2-D MDP (there is at most a difference of $0.5r$). In this subsection, we study the difference in the discounted sum costs when using the policy obtained from the distance-based MDP and the true optimal policy for the 2-D MDP. The result is summarized as Theorem 3.

Theorem 3. *Let $V^*_{\text{dist}}(e)$ denote the discounted sum cost when using the policy that is optimal for the distance-based MDP, and let $V^*(e)$ denote the discounted sum cost when using true optimal policy of the 2-D MDP, then we have $V^*_{\text{dist}}(e) - V^*(e) \leq \frac{\gamma r \kappa}{1-\gamma}$ for all e, where $\kappa \triangleq \max_x \{b(x+2) - b(x)\}$.*

Proof. (Outline) The proof is completed in three steps. First, we modify the states of the 2-D MDP in such a way that the aggregate transition probability from any state $(i'_0, j'_0) \neq (0, 0)$ to ring $i_1 = i'_0 + 1$ (correspondingly, $i_1 = i'_0 - 1$) is $2.5r$ (correspondingly, $1.5r$). We assume that we use a given policy on both the original and modified 2-D MDPs, and show a bound on the difference in the discounted sum costs for these two MDPs. In the second step, we show that the modified 2-D MDP is equivalent to the distance-based MDP. This can be intuitively explained by the reason that the modified 2-D MDP has the same transition probabilities as the distance-based MDP when only considering the ring index i, and also, the one-timeslot cost $C_a(e(t))$ only depends on $\|e(t) - a(e(t))\|$ and $\|a(e(t))\|$, both of which can be determined from the ring indices of $e(t)$ and $a(e(t))$. The third step uses the fact that the optimal policy for the distance-based MDP cannot bring higher discounted sum cost for the distance-based MDP (and hence the modified 2-D MDP) than any other policy. By utilizing the error bound found in the first step twice, we prove the result. For details of the proof, see Appendix E. □

The error bound is a constant value when all the related parameters are given. It increases with γ. However, the absolute value of the discounted sum cost also increases with γ, so the relative error can remain low.

D. Numerical Evaluation

The error bound derived in Section V-C is a worst-case upper bound of the error. In this subsection, we evaluate the performance of the proposed approximation method numerically, and focus on the average performance of the approximation.

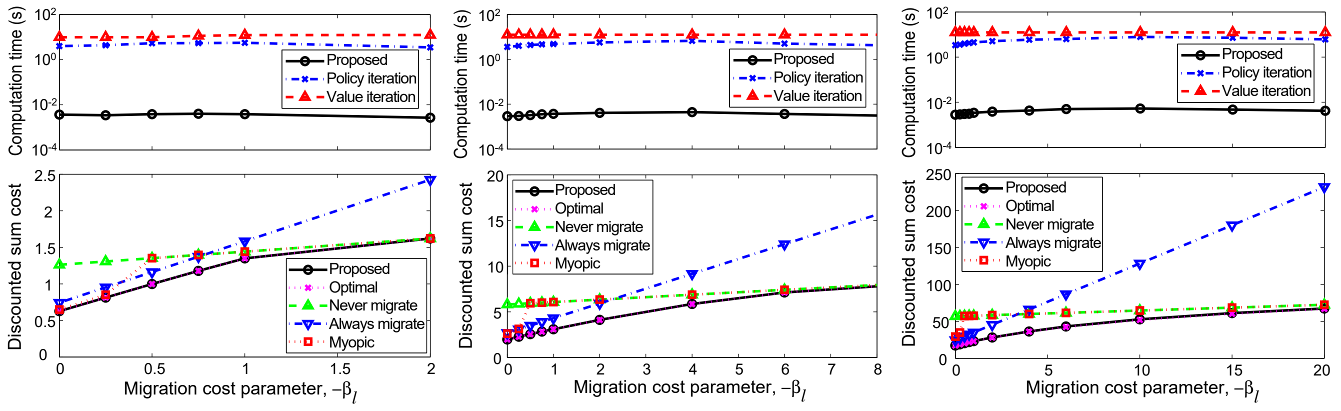
We consider 2-D random walk mobility with randomly chosen parameter r. The maximum user-service distance is set as $N = 10$. The transmission cost function parameters are selected as $\theta = 0.8$, $\delta_c = 1$, and $\delta_l = -1$. With these parameters, we have $\delta_c + \delta_l = 0$, which means that there is no constant portion in the cost function. For the migration cost, we choose $\mu = 0.8$ and fix $\beta_c + \beta_l = 1$ to represent a constant server processing cost for migration. The parameter $\beta_l \leq 0$ takes different values in the simulations, to represent different sizes of data to be migrated.

The simulations are performed in MATLAB on a computer with Intel Core i7-2600 CPU, 8GB memory, and 64-bit Windows 7. We study the computation time (i.e., the time used to run the algorithm) and the discounted sum cost of the proposed approach that is based on approximating the original 2-D MDP with the distance-based MDP. For the computation time comparison, standard value and policy iteration approaches [21, Chapter 6] are used to solve the original 2-D MDP. The discounted sum cost from the proposed approach is compared with the costs from alternative policies, including the *true optimal* policy from standard policy iteration on the 2-D model, the *never-migrate* policy which never migrates except when at states in ring $i \geq N$ (in which case the service is migrated to the current location of the user), the *always-migrate* policy which always migrates to the current user location when the user and service are at different locations, and the *myopic* policy that chooses actions to minimize the one-slot cost. Note that the always-migrate policy is suggested in [29], [30], and a myopic policy that makes migration decisions based on instantaneous quality of service observations is proposed in [31].

The simulations are run with 50 different random seeds, and the overall results are shown in Fig. 6 with different values of the discount factor γ.

1) Reduction in Computation Time: Fig. 6 shows that the computation time of the proposed method is only about 0.1% of that of standard value or policy iteration. This time reduction is explained as follows. As discussed in Section V-A, for a distance-based MDP with N states (excluding state zero), the 2-D MDP has $M = 3N^2 + 3N$ states (excluding state $(0, 0)$). When we ignore the complexity of matrix inversion in the policy iteration procedure, the standard value and policy iteration approaches on the 2-D MDP have a complexity of $O(M^2)$ in each iteration, because the optimal action needs to be found for each state, which requires enumerating through all the states and all possible actions for each state (similarly as in Lines 18–20 of Algorithm 1). In the simulations, $N = 10$, so we have $M = 330$. Recall that the complexity of Algorithm 1 used in the proposed approach is $O(N^2)$, so the ratio of the computational complexities of different approaches can be approximated by $\frac{M^2}{N^2} \approx 10^3$. Therefore, the standard value and policy iteration consume about 10^3 times more computation time compared to the proposed approach. Also note that the computation time of our proposed approach is only a few milliseconds (see Fig. 6), whereas the MDP solution approach used in [7] can easily take several seconds to minutes to obtain the solution for a similar sized system.

2) Near-Optimal Cost: We can also see from Fig. 6 that the proposed method yields a discounted sum cost that is very close to the optimal cost. The results also provide several



(a) (b) (c)

Figure 6. Simulation result with 2-D random walk: (a) $\gamma = 0.5$, (b) $\gamma = 0.9$, (c) $\gamma = 0.99$.

insights into the performance of the baseline policies. Specifically, the cost of the always-migrate policy approximates the optimal cost when $|\beta_l|$ is small, and the cost of the never-migrate policy approximates the optimal cost when $|\beta_l|$ is large. This is because, when $|\beta_l|$ is small, the migration cost is relatively small, and migration can be beneficial for most cases; when $|\beta_l|$ is large, the migration cost is large, and it is better not to migrate in most cases. The myopic policy is the same as the never-migrate policy when $|\beta_l| \geq 0.5$, because $b(x)$ and $c(y)$ are both concave according to the simulation settings and we always have $b(x) \geq c(y)$ when $|\beta_l| \geq 0.5$, where we recall that the myopic policy does not consider the future impact of actions. There is an intersection of the costs from never-migrate and always-migrate policies. When $|\beta_l|$ takes values that are larger than the value at the intersection point, the optimal cost is close to the cost from the never-migrate policy when γ is small, and the gap becomes larger with a larger γ. The reason is that the benefit of migration becomes more significant when we look farther ahead into the future. We also note that the cost of never-migrate policy slightly increases as $|\beta_l|$ increases, because the never-migrate policy also occasionally migrates when the user-service distance greater than or equal to N (see earlier definition).

VI. Application to Real-World Scenarios

In this section, we discuss how the aforementioned approaches can be applied to service migration in the real world, where *multiple users and services co-exist* in the cloud system, and the transition probability parameter r is estimated based on an interval of recent observations before the current time. We note that in practical scenarios, ESs may *not* be deployed at every BS, and each ES may have a *capacity limit* that restricts the number of services it can host. Theoretically, it is still possible to formulate the service migration problem with these additional constraints as an MDP. However, the resulting MDP will have a significantly larger state space than our current model, and it is far more difficult to solve or approximate this new MDP. While we leave the theoretical analysis of this new MDP as future work, we propose a heuristic approach in this section to handle these additional constraints. The proposed approach is largely based on the results and intuitions obtained in previous sections. The 2-D MDP approximation approach proposed in Section V-B is used as a subroutine in the proposed scheme, and the distance-based MDP resulting from the approximation is solved using Algorithm 1.

A. Mapping between Real-World and MDP-Model

The mapping between the real-world and the MDP model is discussed as follows.

MEC Controller: We assume that there exists a control entity which we refer to as the *MEC controller*. The MEC controller does not need to be a separate cloud entity. Rather, it can be a service running at one of the ESs.

Base Stations: Each BS (which may or may not have an ES attached to it) is assumed to have basic capability of keeping records on arriving and departing users, and performing simple monitoring and computational operations.

Timeslots: The physical time length corresponding to a slot in the MDP is a pre-specified parameter, which is a constant for ease of presentation. This parameter can be regarded as a protocol parameter, and it is *not* necessary for all BSs to precisely synchronize on individual timeslots.

Transition Probability: The transition probability parameter r is estimated from the sample paths of multiple users, using the procedure described in Section VI-B below. We define a window length $T_w \geq 1$ (represented as the number of timeslots), which specifies the amount of timeslots to look back to estimate the parameter r. We consider the case where r is the same across the whole geographical area, which is a reasonable assumption when different locations within the geographical area under consideration have similarities (for example, they all belong to an urban area). More sophisticated cases can be studied in the future.

Distance: The discrete distance in the MDP model can be measured using metrics related to the displacement of the user, such as the number of hops between different BSs or quantized Euclidean distance. The simulations in Section VI-E show that both metrics are good for the proposed method to work well in practical scenarios.

Cost Parameters: The cost parameters β_c, β_l, μ, δ_c, δ_l, and θ are selected based on the actual application scenario and MEC system characteristics, and their values may vary with the background traffic load of the network and ESs.

Discount Factor: The discount factor γ balances the trade-off between short-term and long-term costs. For example, if we set $\gamma = 0$, the algorithm minimizes the instantaneous cost only, without considering the future impact. This is essentially the same as the myopic policy. If we set $\gamma \approx 1.0$, then the algorithm aims at minimizing the long-term average cost. However, the instantaneous cost may be high in some timeslots in this case. The choice of γ in practice depends on the acceptable level of fluctuation in the instantaneous cost, the importance of low average cost, and the time duration that the user accesses the service. In general, a larger γ reduces the long-term average cost but the instantaneous cost in some slots may be higher. Also, if a user only requires the service for a short time, then there is no need to consider the cost for the long-term future. For ease of presentation, we set γ as a constant value. In practice, the value of γ can be different for different users or services.

Policy Update Interval: A policy update interval T_u is defined (represented as the number of timeslots), at which a new migration policy is computed by the MEC controller.

B. Overall Procedure

The data collection, estimation, and service placement procedure is described below.

1) At the beginning of each slot, the following is performed:

a) Each BS obtains the identities of its associated users. Based on this information, the BS computes the number of users that have left the cell (compared to the beginning of the previous timeslot) and the total number of users that are currently in the cell. This information is saved for each timeslot for the duration of T_w and will be used in step 2b.

b) The MEC controller collects information on currently active services on each ES, computes the new placement of services according to the procedure described in Section VI-C, and sends the resulting placement instructions to each ES. The placements of all services are updated based on these instructions.

2) At every interval T_u, the following is performed:

a) The MEC controller sends a request to all BSs to collect the current statistics.

b) After receiving the request, each BS n computes the empirical probability of users moving outside of the cell:

$$f_n = \frac{1}{T_w} \sum_{\tau=t-T_w}^{t-1} \frac{m'_n(\tau)}{m_n(\tau)} \tag{18}$$

where the total number of users that are associated to BS n in slot τ is $m_n(\tau)$, among which $m'_n(\tau)$ users have disconnected from BS n at the end of slot τ and these users are associated to a different BS in slot $\tau + 1$; and t denotes the current timeslot index. These empirical probabilities are sent together with other monitored information, such as the current load of the network and ES (if the BS has an ES attached to it), to the MEC controller.

c) After the controller receives responses from all BSs, it performs the following:

i) Compute the transmission and migration cost parameters β_c, β_l, μ, δ_c, δ_l, and θ based on the measurements at BSs.

ii) Compute the average of empirical probabilities f_n by

$$\bar{f} = \frac{1}{N_{\text{BS}}} \sum_{n \in \mathcal{N}_{\text{BS}}} f_n \tag{19}$$

where $\mathcal{N}_{\text{BS}}$ is the set of BSs and $N_{\text{BS}} = |\mathcal{N}_{\text{BS}}|$ is the total number of BSs. Then, estimate the parameter r by

$$\hat{r} = \bar{f}/6. \tag{20}$$

iii) In the distance-based MDP, set $p_0 = 6\hat{r}$, $p = 2.5\hat{r}$, and $q = 1.5\hat{r}$ (as discussed in Section V-B), compute and save the optimal distance-based policy from Algorithm 1. Also save the estimated cost parameters and the optimal discounted sum costs $V^*(d)$ for all d for later use.

Remark: In the procedure presented above, we have assumed that $m_n(\tau) \neq 0$ for all n and τ. This is only for ease of presentation. When there exist some n and τ such that $m_n(\tau) = 0$, we can simply ignore those terms (set the corresponding terms to zero) in the sums of (18) and (19), and set the values of T_w and N_{BS} to the actual number of terms that are summed up in (18) and (19), respectively.

In essence, the above procedure recomputes $\hat{r}$ and other model parameters at an interval of T_u timeslots, using measurements obtained in the previous T_w slots. This allows the MDP model and the algorithm to adapt to the most recent characteristics of the system, which may dynamically change over time due to network and user dynamics.

C. Service Placement Update

At the start of every timeslot, service placement is updated and migration is performed when needed (step 1b in Section VI-B).

The policy found from the MDP model specifies which cell to migrate to when the system is in a particular state $(u(t), h(t))$. However, we may not be able to apply the policy directly, because not every BS has an ES attached to it and each ES has a capacity limit. We may need to make some modifications to the service placement specified by the policy, so that the practical constraints are not violated. The MDP model also does not specify where to place the service if it was not present in the system before. In the following, we present a method to determine the service placement with these practical considerations, which is guided by the optimal policy obtained from the MDP model and at the same time satisfies the practical constraints.

The algorithm first ignores the ES capacity limit and repeat the process in steps (I) and (II) below for every service. Then, it incorporates the capacity limit, and reassign the locations of some services (in step (III)) that were previously assigned to an ES whose capacity is exceeded.

(I) Initial Service Placement: When the service was not running in the system before (i.e., it is being initialized), the service is placed onto an ES that has the smallest distance to the user. The intuition for this rule is that the initialization cost and the cost of further operation is usually small with such a placement.

(II) Dynamic Service Migration: When the service has been initialized earlier and is currently running in the system, a decision on whether and where to migrate the service is made. Without loss of generality, assume that the current timeslot is $t = 0$. We would like to find an action a that is the solution to the following problem:

$$\min_a C_a(d) + \gamma \sum_{d(1)=a(d)-1}^{a(d)+1} P[a(d), d(1)] \cdot V^*(d(1)) \tag{21}$$

s.t. there exists an ES such that the user-service distance is $a(d)$ after migration

where $V^*(d)$ stands for the optimal discounted sum cost found from step 2(c)iii in Section VI-B. We note that (21) is a one-step value iteration following the balance equation (6). Intuitively, it means that assuming the optimal actions are taken in future slots, find the action for the current slot that incurs the lowest discounted sum cost (including both immediate and future cost). When all BSs have ESs attached to them, the solution a to problem (21) is the same as the optimal action of the MDP-model found from step 2(c)iii in Section VI-B. However, the optimal a from (21) may be different from the optimal action from the MDP model when some BSs do not have ESs attached. The resulting distance-based migration action can be mapped to a migration action on 2-D space using the procedure in Section V-B.

(III) Reassign Service Location if ES's Capacity Exceeded: The placement decisions in steps (I) and (II) above do not consider the capacity limit of each ES, so it is possible that we find the ES capacity is exceeded after following the steps in the above sections. When this happens, we start with an arbitrary ES (denoted by i_0) whose capacity constraint is violated. We rank all services in this ES according to the objective function in (21), and start to reassign the service location with the highest objective function value. This service is placed on an ES that still has capacity for hosting it, where the placement decision is also made according to (21) but only the subset of ESs that are still capable of hosting this service are considered. The above process is repeated until the number of services hosted at ES i_0 becomes within its capacity limit. Then, this whole process is repeated for other ESs with exceeded capacity.

Remark: We note that service relocation does not really occur in the system. It is only an intermediate step in the algorithm for finding new service locations. We use this two-step approach involving temporary placement and relocation instead of an alternative one-step approach that checks for ES capacity when performing the placement/migration in steps (I) and (II), because with such a two-step approach, we can leave the low-cost services within the ES and move high-cost services to an alternative location.

D. Performance Analysis

1) Estimation of Parameter r: As introduced in Section V, at every timeslot, each user randomly moves to one of its neighboring cells with probability r and stays in the same cell with probability $1 - 6r$. In the real world, the parameter r is unknown a priori and needs to be estimated based on observations of user movement. Equations (18)–(20) in Section VI-B serve for this estimation purpose, and the resulting $\hat{r}$ is an estimator of r. In the following, we analyze some statistical properties of $\hat{r}$ and discuss the rationale for using such an estimation approach[8].

We note that the mobility model presented in Section V is for an infinite 2-D space with an infinite number of cells. In reality, the number of cells is finite. We assume in our analysis that each user stays in the same cell with probability $1 - 6r$ ($r \leq \frac{1}{6}$) and moves out of its current cell with probability $6r$, no matter whether the cell is at the boundary (such as cells in the outer ring $i = 3$ in Fig. 5) or not. When a cell is at the boundary, its transition probability to each of its neighboring cells is larger than r, because it has less than six neighbors. For example, in Fig. 5, a user in cell $(3, 0)$ moves to *each* of its neighboring cells (including $(3, 17), (2, 0), (3, 1)$) with probability $2r$, and the total probability of moving out of the cell is still $6r$. We also assume that the mobility patterns of different users are independent of each other.

Theorem 4. *Assume that each user follows 2-D random walk (defined above) with parameter r, then $\hat{r}$ is an unbiased estimator of r, i.e., $\mathbb{E}\{\hat{r}\} = r$.*

Proof. See Appendix F. □

The fact that $\hat{r}$ is an unbiased estimator of r justifies our estimation approach, which intuitively means that the long-term average of the estimated value $\hat{r}$ should not be too far away from the true value of r.

We analyze the variance of the estimator next. Such analysis is not very easy due to the dependency among different random variables. To make the analysis theoretically tractable, we introduce the following assumption.

Assumption 1. *We assume that $m'_n(\tau)$ is independent of $m_n(\tilde{\tau})$, $m_{\tilde{n}}(\tau)$, $m_{\tilde{n}}(\tilde{\tau})$, $m'_n(\tilde{\tau})$, $m'_{\tilde{n}}(\tau)$, and $m'_{\tilde{n}}(\tilde{\tau})$ (where $\tilde{n} \neq n$ and $\tilde{\tau} \neq \tau$) when $m_n(\tau)$ is given.*

Essentially, this assumption says that $m'_n(\tau)$ is only dependent on $m_n(\tau)$. In practice when the number of users is large, this assumption is close to reality, because the locations of different users are independent of each other, and also because of the Markovian property which says that future locations of a user only depends on its present location (and independent of all past locations when the present location is given).

Theorem 5. *Assume that Assumption 1 is satisfied and each user follows 2-D random walk (defined at the beginning of Section VI-D1) with parameter r. The variance of estimator $\hat{r}$ is upper bounded by $\mathrm{Var}\{\hat{r}\} \leq \frac{1}{144 N_{BS} T_w}$.*

Proof. See Appendix G. □

We see from Theorem 5 that the upper bound of variance is inversely proportional to T_w. This is an intuitive and also very favorable property, which says that when users follow an ideal random walk mobility model with parameter r, we can

[8]In the analysis, we still assume that $m_n(\tau) \neq 0$ for all n and τ as in Section VI-B. For cases with $m_n(\tau) = 0$, we can make a similar substitution as described in the remark in Section VI-B.

estimate the value of r as accurate as possible if we have a sufficient amount of samples.

Different from many estimation problems where it is costly (requiring human participation, data communication, etc.) to obtain samples, it is not too difficult to collect samples in our case, because each BS can save user records at every timeslot, and we can easily adjust the number of samples (proportional to T_w) by changing the amount of timeslots to search back in the record. Therefore, unlike many other estimation problems where the goal is to minimize the variance of estimator under a given sample size, we do not target this goal here. A much more important issue in our problem is that the ideal random walk mobility model may not hold in practice. The parameter estimation procedure in Section VI-B considers possible model violations, and its rationale is explained below.

In the estimation procedure, we first focus on a particular cell n and compute the empirical probability of users moving out of that cell in (18), by treating each timeslot with equal weight. Then, we take the average of such empirical probabilities of all cells in (19). We note that the number of users in different cells and timeslots may be imbalanced. Thus, in (18), the empirical probabilities for different cells and slots may be computed with different number of users (samples), i.e., different values of $m_n(\tau)$.

Suppose we use an alternative approach that first computes the total number of users in all cells and all slots (i.e., $\sum_{n\in\mathcal{N}_{\text{BS}}}\sum_{\tau=t-T_w}^{t-1} m_n(\tau)$) and the aggregated total number of users leaving their current cells at the end of slots (i.e., $\sum_{n\in\mathcal{N}_{\text{BS}}}\sum_{\tau=t-T_w}^{t-1} m'_n(\tau)$). Then, this approach estimates r by the overall empirical probability of users moving out of their current cells (i.e., $\frac{\sum_{n\in\mathcal{N}_{\text{BS}}}\sum_{\tau=t-T_w}^{t-1} m'_n(\tau)}{6\cdot\sum_{n\in\mathcal{N}_{\text{BS}}}\sum_{\tau=t-T_w}^{t-1} m_n(\tau)}$). Intuitively, this alternative estimator may bring a lower variance compared to our estimator $\hat{r}$, because it treats all the samples as a whole.

However, the uniform random walk mobility model *may not hold* precisely in practice; even if it holds, the parameter *r may be time-varying or different in different geographical areas*. Thus, we compute the empirical probability for each cell in each slot first, so that different cells and slots are equally weighted in the estimation, although some cells may have more users than others in certain slots. This is for the consideration of fairness among different cells, so that the performance at different cells remains similar. It is also to avoid statistics at a single slot dominating the overall result. Since r may be time-varying in practice, it is possible that single-slot statistics do not represent the overall case.

2) Cost Difference Due to Additional Constraints: Due to the constraints on ES existence and capacity, the procedure in Section VI-C may have to choose an action $a(d)$ that is different from the optimal action $a^*(d)$ of the distance-based MDP. The following theorem captures the effect of such non-optimal choices, from the MDP's point of view.

Theorem 6. *Assume that the actual action* $a(d) = [a^*(d)+k]_0^{N-1}$ *with probability* α_k*, and there exists* $K \geq 0$ *such that* $\alpha_k = 0$ *for all* $|k| > K$*. Let* $\tilde{V}(d)$ *and* $V^*(d)$ *denote the discounted sum costs of the distance-based MDP, when following the actual and optimal actions, respectively. We have* $\tilde{V}(d) - V^*(d) \leq \frac{\gamma\kappa'}{1-\gamma}$ *for all* d*, where* $\kappa' \triangleq \max_x\{b(x+K+2)-b(x)\}$ *and* $[x]_v^w$ *denotes* $\min\{\max\{x,v\},w\}$.

Proof. See Appendix H. □

In the above theorem, we assume that the actual actions are random for the ease of analysis. The probabilities α_k (and K) depend on the additional constraints on ES existence/capacity and the load of the system. Intuitively, K becomes large when the system is heavily loaded or when only a few of the BSs have ESs attached to them, because the system may need to choose an action that is far away from the optimal action in this case. Since the migration cost $b(x)$ is non-decreasing with x, a larger K can only give a worse (or equal, but not better) error bound, as one would intuitively expect.

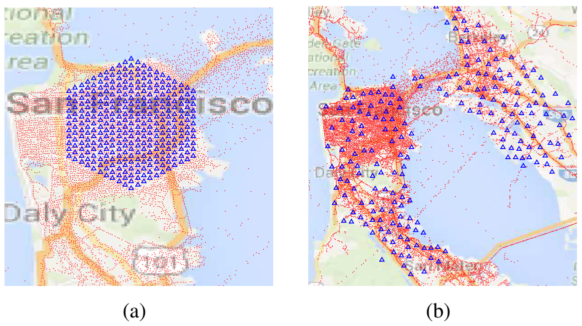


Figure 7. BS and taxi locations: (a) Hexagonal BS placement, (b) Real BS placement. The blue triangles indicate the BS location and the red dots indicate possible taxi locations. There appears to be a small amount of erroneous/inaccurate data for taxi locations but the majority of them are correct. Map data: Google.

E. Trace-Driven Simulation

We perform simulation with real-world mobility traces of 536 taxis in San Francisco, collected on May 31, 2008 [35], [36], where different number of taxis are operating (i.e., active) at different time of the day. Each active taxi is modeled as a user that requests a service that is independent of the services of the other taxis. Two different ways of BS placement are considered, as shown in Fig. 7. The first is a *hexagonal* BS placement with 500 m cell separation and $N_{\text{BS}} = 331$ cells (BSs) in total in the central area of San Francisco. The second placement uses *real* cell tower locations obtained from www.antennasearch.com on a larger geographical area. To limit the total number of BSs in the simulation, the cell tower locations in the original dataset are down-selected so that the distance between two BSs is at least $1,000$ m. In both cases, each taxi connects to its nearest BS measured by Euclidean distance. This modeling of user mobility and BS locations is similar to other recent work including [7], [27], [31], [37]. For the hexagonal BS placement, the distance of the MDP model is computed as the length of the shortest path between two different BSs. For the real BS placement, the distance is computed as the Euclidean distance between BSs divided by $1,000$ m and rounded up to the next integer.

We set the physical time length for each timeslot as 60 s. The parameters for data collection and policy update are set as $T_u = 1$ slot, and $T_w = 60$ slots. We choose $N = 10$ and $\gamma = 0.9$. From Theorem 5, we know that for the hexagonal

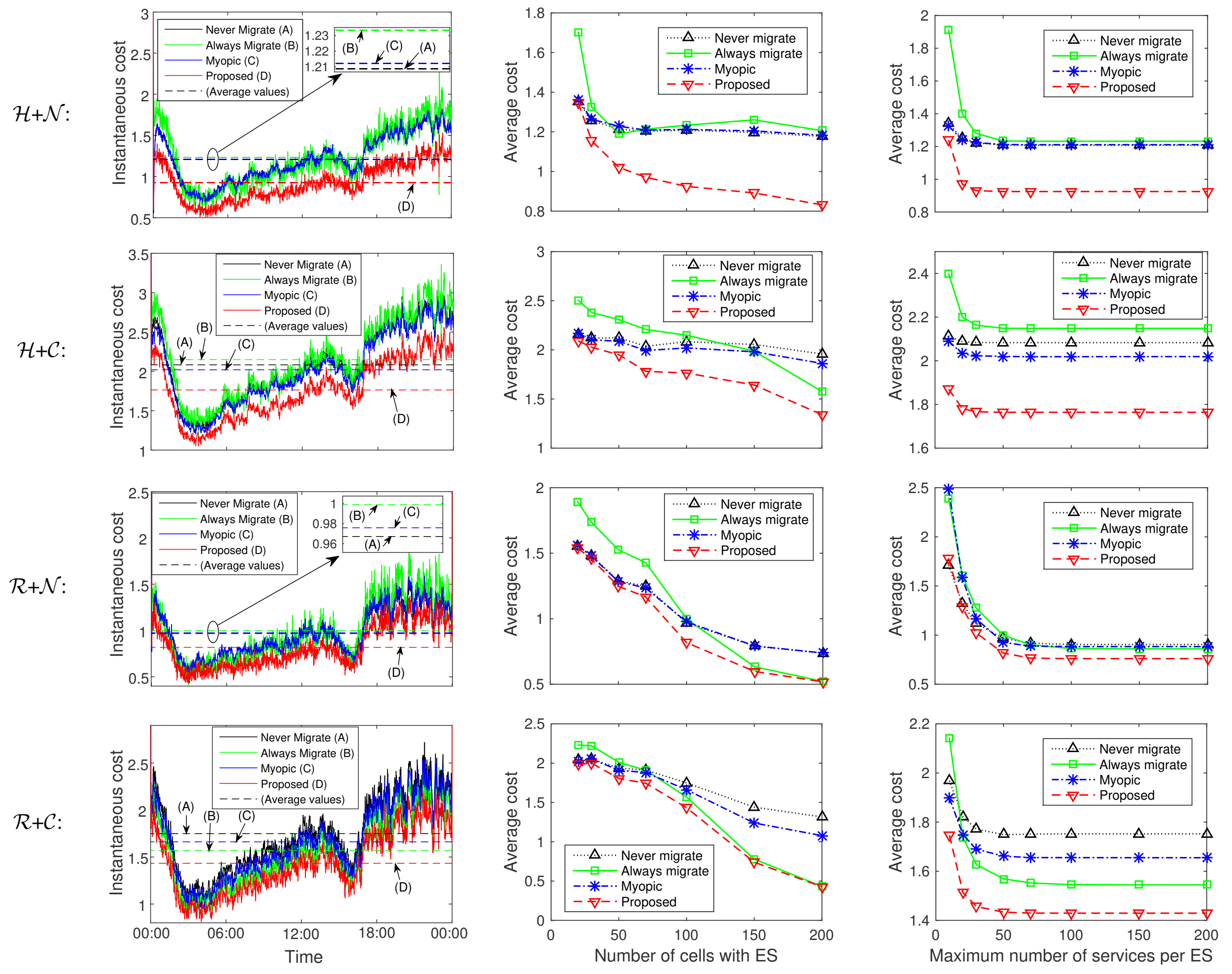


Figure 8. Instantaneous and average costs over a day in trace-driven simulation, where $R_t = R_p = 1.5$, $\mathcal{H}$ and $\mathcal{R}$ respectively stand for $\mathcal{H}$exagonal and $\mathcal{R}$eal BS placement, $\mathcal{N}$ and $\mathcal{C}$ respectively stand for $\mathcal{N}$on-constant and $\mathcal{C}$onstant costs. In some plots, an enlarged plot of the circled area is shown. The arrows annotated with (A), (B), (C), and (D) point to the average values over the whole day of the corresponding policy.

BS placement, the standard deviation of estimator $\hat{r}$ (for an ideal random walk mobility model) is upper-bounded by $\sqrt{\frac{1}{144 N_{\mathrm{BS}} T_w}} = 0.00059$, which is reasonably small. Unless otherwise specified, there are 100 BSs that have ESs attached to them and each ES can host at most 50 services. The BSs with ESs are evenly distributed among all BSs. Note that the locations of taxis in the dataset are unevenly distributed and the density of taxis can be very high in some areas at certain times of the day.

Similar to Section V-D, we compare the performance of the proposed method with baseline policies including always/never-migrate and myopic. To cope with the nature of the policies, the objective function in (21) is modified accordingly for these baseline policies. The objective in (21) is defined as the user-service distance for the always- or never-migrate policies (recall that migration also happens with never-migrate policy when the user-service distance is N or larger), and it is defined as the one-timeslot cost for the myopic policy.

Cost definition: The cost is defined as related to both the distance and the system load, where the latter may introduce additional delay due to queueing. We assume that the system load is proportional to the number of taxis in operation, and define parameters $R_t, R_p > 1$ representing the availability of two types of resources. Then, we define $G_t \triangleq 1 \Big/ \left(1 - \frac{m_{\mathrm{cur}}}{R_t m_{\max}}\right)$ and $G_p \triangleq 1 \Big/ \left(1 - \frac{m_{\mathrm{cur}}}{R_p m_{\max}}\right)$, where m_{cur} denotes the number of taxis in operation at the time when the optimal policy is being computed, and $m_{\max}$ denotes the maximum number of taxis that may simultaneously operate at any time instant in the considered dataset. The above expressions for G_t and G_p have the same form as the average queueing delay expression [38], [39].

We set $\mu = \theta = 0.8$ and consider two different cost definitions. The first is *non-constant cost* with parameters defined as $\beta_c = G_p + G_t$, $\beta_l = -G_t$, $\delta_c = G_t$, and $\delta_l = -G_t$. With such a definition, according to (4) and (5), we have that whenever the distance $x, y > 0$, the migration cost $b(x) > \beta_c + \beta_l = G_p$ representing a non-zero processing cost whenever the distance is not zero, and the transmission cost

$c(y) > \delta_c + \delta_l = 0$. The costs $b(x)$ and $c(y)$ increase with x and y respectively, at a rate related to G_t, to represent the cost due to network communication (incurred in both migration and user-ES data transmission) which is related to the distance. The second type of cost we consider is *constant cost*, where we set $\beta_c = G_p$, $\delta_c = G_t$, and $\beta_l = \delta_l = 0$. With this definition, the migration cost is G_p and the transmission cost is G_t whenever the distance is larger than zero.

Results: The results are shown in Fig. 8, which includes the instantaneous costs over the day (i.e., the $C_a(s(t))$ values, averaged over all users that are active in that slot), and the average costs over the day with different number of cells with ES and different capacities per ES. We see that proposed approach gives lower costs than other approaches in almost all cases. The fluctuation in the instantaneous cost is due to different system load (number of active taxis) over the day. This also implies that it is necessary to compute the optimal policy in real-time, based on recent observations of the system condition. The average cost of the proposed approach becomes closer to that of never-migrate and myopic policies when the number of ES or the capacity per ES is small, because it is hardly possible to migrate the service to a better location due to the constrained resource availability in this case. Additional simulation results are presented in Appendix I, showing that the proposed approach outperforms the other approaches also with different values of R_t, R_p.

VII. Discussions

Some assumptions have been made in the paper to make the problem theoretically tractable and also for the ease of presentation. In this section, we justify these assumptions from a practical point of view and discuss possible extensions.

Cost Functions: To ease our discussion, we have limited our attention to transmission and migration costs in this paper. This can be extended to include more sophisticated cost models. For example, the transmission cost can be extended to include the computational cost of hosting the service at an ES, by adding a constant value to the transmission cost expression. As in Section VI-E, the cost values can also be time-varying and related to the background system load. Furthermore, the cost definition can be practically regarded as the average cost over multiple locations, which means that when seen from a single location, the monotonicity of cost values with distances does not need to apply. This makes the proposed approach less restrictive in terms of practical applicability.

We also note that it is generally possible to formulate an MDP with additional dimensions in cost modeling, such as one that includes the state of the network, load at each specific ES, state of the service to avoid service interruption when in critical state, etc. However, this requires a significantly larger state space compared to our formulation in this paper, as we need to include those network/ES/service states in the state space of the MDP. There is a tradeoff between the complexity of solving the problem and accuracy of cost modeling. Such issues can be studied in the future, where we envision similar approximation techniques as in this paper can be used to approximately solve the resulting MDP.

Single/Multiple Users: As pointed out in Section III, although we have focused on a single user in our problem modeling, practical cases involving multiple users running independent services can be considered by setting cost functions related to the background traffic generated by other users, as in Section VI. For more complicated cases such as multiple users sharing the same service, or where the placement of different services is strongly coupled and reflected in the cost value, we can formulate the problem as an MDP with larger state space (similarly to the generalized cost model discussed above).

Uniform Random Walk: The uniform random walk mobility model is used as a modeling assumption, which not only simplifies the theoretical analysis, but also makes the practical implementation of the proposed method fairly simple in the sense that only the empirical probability of users moving outside of the cell needs to be recorded (see Section VI-B). This model can capture the average mobility of a large number of users. The simulation results in Section VI-E confirm that this model provides good performance, even though individual users do not necessarily follow a uniform random walk.

Centralized/Distributed Control: We have focused on a centralized control mechanism in this paper for the ease of presentation. However, many parts of the proposed approach can be performed in a distributed manner. For example, in step 1b in Section VI-B, the service placement decision can be made among a smaller group of ESs if the controller sends the results from step 2(c)iii to these ESs. In particular, the temporary service placement decision in steps (I) and (II) in Section VI-C can be made locally on each ES (provided that it knows the locations of other ESs). The capacity violation check in step (III) in Section VI-C can also be performed locally. If some ESs are in excess of capacity, service relocation can be performed using a few control message exchange between ESs, where the number of necessary messages is proportional to the number of services to relocate. Relocation would rarely occur if the system load is not very high. The computation of average empirical probability in step 2(c)ii in Section VI-B can also be distributed in the sense that a subset of ESs compute local averages, which are subsequently sent to the MEC controller that computes the global average.

ES-BS Co-location: For ease of presentation, we have assumed that ESs are co-located with BSs. However, our proposed approach is not restricted to such cases and can easily incorporate scenarios where ESs are not co-located with BSs as long as the costs are geographically dependent.

VIII. Conclusions

In this paper, we have studied service migration in MEC. The problem is formulated as an MDP, but its state space can be arbitrarily large. To make the problem tractable, we have reduced the general problem into an MDP that only considers a meaningful parameter, namely the distance between the user and service locations. The distance-based MDP has several structural properties that allow us to develop an efficient algorithm to find its optimal policy. We have then shown that the distance-based MDP is a good approximation to scenarios where the users move in a 2-D space, which is confirmed by analytical and numerical evaluations. After that, we have presented a method of applying the theoretical results to a practical setting, which is evaluated by simulations with real-world data traces of taxis and cell towers in San Francisco.

The results in this paper provide an efficient solution to service migration in MEC. Further, we envision that our theoretical approach can be extended to a range of other MDP problems that share similar properties. The highlights of our approaches include: a closed-form solution to the discounted sum cost of a particular class MDPs, which can be used to simplify the procedure of finding the optimal policy; a method to approximate an MDP (in a particular class) with one that has smaller state space, where the approximation error can be shown analytically; and a method to collect statistics from the real-world to serve as parameters of the MDP.

Acknowledgment

Contribution of S. Wang is partly related to his previous affiliation with Imperial College London. Contributions of R. Urgaonkar, M. Zafer, and T. He are related to their previous affiliation with IBM Research.

References

[1] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *Proc. of IFIP Networking 2015*, May 2015.

[2] S. Wang, "Dynamic service placement in mobile micro-clouds," Ph.D. dissertation, Imperial College London, 2015.

[3] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[4] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, Jan. 2017.

[5] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[6] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. PP, no. 99, 2017.

[7] W. Zhang, J. Chen, Y. Zhang, and D. Raychaudhuri, "Towards efficient edge cloud augmentation for virtual reality mmogs," in *Proc. of ACM/IEEE Symposium on Edge Computing (SEC)*, Oct. 2017.

[8] M. Satyanarayanan, R. Schuster, M. Ebling, G. Fettweis, H. Flinck, K. Joshi, and K. Sabnani, "An open ecosystem for mobile-cloud convergence," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 63–70, March 2015.

[9] "Smarter wireless networks," *IBM Whitepaper No. WSW14201USEN*, Feb. 2013. [Online]. Available: www.ibm.com/services/multimedia/Smarter_wireless_networks.pdf

[10] M. Satyanarayanan, G. Lewis, E. Morris, S. Simanta, J. Boleng, and K. Ha, "The role of cloudlets in hostile environments," *IEEE Pervasive Computing*, vol. 12, no. 4, pp. 40–49, Oct. 2013.

[11] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Live service migration in mobile edge clouds," *IEEE Wireless Communications*, vol. PP, no. 99, pp. 2–9, 2017.

[12] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, 2012, pp. 13–16.

[13] T. Taleb and A. Ksentini, "Follow me cloud: interworking federated clouds and distributed mobile networks," *IEEE Network*, vol. 27, no. 5, pp. 12–19, Sept. 2013.

[14] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1002–1016, Apr. 2017.

[15] M. Selimi, L. Cerdà-Alabern, M. Sánchez-Artigas, F. Freitag, and L. Veiga, "Practical service placement approach for microservices architecture," in *Proc. of CCGrid*, May 2017, pp. 401–410.

[16] Z. Becvar, J. Plachy, and P. Mach, "Path selection using handover in mobile networks with cloud-enabled small cells," in *IEEE PIMRC*, Sept. 2014, pp. 1480–1485.

[17] T. Taleb and A. Ksentini, "An analytical model for follow me cloud," in *Proc. of IEEE GLOBECOM 2013*, Dec. 2013.

[18] A. Ksentini, T. Taleb, and M. Chen, "A Markov decision process-based service migration procedure for follow me cloud," in *Proc. of IEEE ICC 2014*, June 2014.

[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. of IEEE MILCOM 2014*, Oct. 2014.

[20] "Mobile-edge computing – introductory technical white paper," Sept. 2014. [Online]. Available: https://portal.etsi.org/portals/0/tbpages/mec/docs/mobile-edge_computing_-_introductory_technical_white_paper_v1%2018-09-14.pdf

[21] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2009, vol. 414.

[22] M. Bienkowski, A. Feldmann, J. Grassler, G. Schaffrath, and S. Schmid, "The wide-area virtual service migration problem: A competitive analysis approach," *IEEE/ACM Trans. on Networking*, vol. 22, no. 1, pp. 165–178, Feb. 2014.

[23] U. Mandal, M. Habib, S. Zhang, B. Mukherjee, and M. Tornatore, "Greening the cloud using renewable-energy-aware service migration," *IEEE Network*, vol. 27, no. 6, pp. 36–43, Nov. 2013.

[24] J. Plachy, Z. Becvar, and E. C. Strinati, "Dynamic resource allocation exploiting mobility prediction in mobile edge computing," in *IEEE PIMRC*, Sept. 2016.

[25] A. Ceselli, M. Premoli, and S. Secci, "Mobile edge cloud network design optimization," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, Jun. 2017.

[26] I.-H. Hou, T. Zhao, S. Wang, and K. Chan, "Asymptotically optimal algorithm for online reconfiguration of edge-clouds," in *ACM MobiHoc 2016*, 2016.

[27] L. Wang, L. Jiao, J. Li, and M. Muhlhauser, "Online resource allocation for arbitrary user mobility in distributed edge clouds," in *2017 IEEE ICDCS*, Jun. 2017, pp. 1281–1290.

[28] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, 2015.

[29] K. Ha, Y. Abe, Z. Chen, W. Hu, B. Amos, P. Pillai, and M. Satyanarayanan, "Adaptive VM handoff across cloudlets," Technical Report CMU-CS-15-113, CMU School of Computer Science, Tech. Rep., 2015.

[30] L. Ma, S. Yi, and Q. Li, "Efficient service handoff across edge servers via docker container migration," in *Proc. of ACM/IEEE Symposium on Edge Computing (SEC)*, Oct. 2017.

[31] E. Saurez, K. Hong, D. Lillethun, U. Ramachandran, and B. Ottenwälder, "Incremental deployment and migration of geo-distributed situation awareness applications in the fog," in *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems*, ser. DEBS '16, 2016, pp. 258–269.

[32] 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 Communications Surveys Tutorials*, vol. 16, no. 1, pp. 64–91, 2014.

[33] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient algorithms for capacitated cloudlet placements," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 10, pp. 2866–2880, Oct. 2016.

[34] S. Elaydi, *An Introduction to Difference Equations, Third Edition*. Springer, 2005.

[35] M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser, "A parsimonious model of mobile partitioned networks with clustering," in *Proc. of COMSNETS*, Jan. 2009.

[36] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser, "CRAWDAD data set epfl/mobility (v. 2009-02-24)," Downloaded from http://crawdad.org/epfl/mobility/, Feb. 2009.

[37] F. Zhang, C. Xu, Y. Zhang, K. K. Ramakrishnan, S. Mukherjee, R. Yates, and T. Nguyen, "Edgebuffer: Caching and prefetching content at the edge in the mobilityfirst future internet architecture," in *IEEE WoWMoM*, Jun. 2015.

[38] L. Kleinrock, *Queuing Systems, Volume II: Computer Applications*. Hoboken, NJ: John Wiley and Sons, 1976.

[39] J. Li and H. Kameda, "Load balancing problems for multiclass jobs in distributed/parallel computer systems," *IEEE Transactions on Computers*, vol. 47, no. 3, pp. 322–332, Mar. 1998.

Appendix A
Proof of Theorem 1

Suppose that we are given a service migration policy π such that the service can be migrated to a location that is farther away from the user, i.e., $\|u-h'\| > \|u-h\|$. We will show that, for an arbitrary sample path of the user locations $u(t)$, we can find a (possibly history dependent) policy ψ that does not migrate to locations farther away from the user in any timeslot, which performs not worse than policy π.

For an arbitrary sample path of user locations $u(t)$, denote the timeslot t_0 as the *first* timeslot (starting from $t=0$) in which the service is migrated to somewhere farther away from the user when following policy π. The initial state at timeslot t_0 is denoted by $s(t_0)=(u(t_0),h(t_0))$. When following π, the state shifts from $s(t_0)$ to $s'_\pi(t_0)=(u(t_0),h'_\pi(t_0))$, where $\|u(t_0)-h'_\pi(t_0)\| > \|u(t_0)-h(t_0)\|$. The subsequent states in this case are denoted by $s_\pi(t)=(u(t),h_\pi(t))$ for $t>t_0$.

Now, we define a policy ψ such that the following conditions are satisfied for the given sample path of user locations $u(t)$:

- The migration actions in timeslots $t<t_0$ are the same when following either ψ or π.
- The policy ψ specifies that there is no migration within the timeslots $t\in[t_0,t_m-1]$, where $t_m>t_0$ is a timeslot index that is defined later.
- The policy ψ is defined such that, at timeslot t_m, the service is migrated to $h'_\pi(t_m)$, where $h'_\pi(t_m)$ is the service location (after possible migration at timeslot t_m) when following π.
- For timeslots $t>t_m$, the migration actions for policies ψ and π are the same.

For $t>t_0$, the states when following ψ are denoted by $s_\psi(t)=(u(t),h_\psi(t))$.

The timeslot t_m is defined as the *first* timeslot after t_0 such that the following condition is satisfied:

1) $\|u(t_m)-h_\psi(t_m)\| > \|u(t_m)-h'_\pi(t_m)\|$, i.e., the transmission cost (before migration at timeslot t_m) when following ψ is larger than the transmission cost (after possible migration at timeslot t_m) when following π.

Accordingly, within the interval $[t_0+1,t_m-1]$, the transmission cost when following ψ is always less than or equal to the transmission cost when following π, and there is no migration when following ψ. Therefore, for timeslots $t\in[t_0,t_m-1]$, policy π cannot bring lower cost than policy ψ.

In timeslot t_m, we can always choose a migration action for ψ where the migration cost is smaller than or equal to the sum of the migration costs of π within $[t_0,t_m]$. The reason is that ψ can migrate (in timeslot t_m) following the same migration path as π within $[t_0,t_m]$.

It follows that, for timeslots within $[t_0,t_m]$, policy π cannot perform better than policy ψ, and both policies have the same costs for timeslots within $[0,t_0-1]$ and $[t_m+1,\infty)$. The above procedure can be repeated so that all the migration actions to a location farther away from the user can be removed without increasing the overall cost, thus we can redefine ψ to be a policy that removes all such migrations.

We note that the policy ψ can be constructed based on policy π without prior knowledge of the user's future locations. For any policy π, the policy ψ is a policy that does not migrate whenever π migrates to a location farther away from the user (corresponding to timeslot t_0). Then, it migrates to $h'_\pi(t_m)$ when condition 1 is satisfied (this is the timeslot t_m in the above discussion).

The policy ψ is a history dependent policy, because its actions depend on the past actions of the underlying policy π. From [21, Chapter 6], we know that history dependent policies cannot outperform Markovian policies for our problem, assuming that the action spaces of both policies are identical for every possible state. Therefore, there exists a Markovian policy that does not migrate to a location farther away from the user, which does not perform worse than π. Noting that the optimal policy found from (2) and (3) are Markovian policies, we have proved the theorem.

Appendix B
Proof of Corollary 1

The proof follows the same procedure as the proof of Theorem 1. For any given policy π that migrates to locations other than the user location, we show that there exists a policy ψ that does not perform such migration, which performs not worse than the original policy π. The difference is that t_0 is defined as the first timeslot such that $u(t_0)\neq h'_\pi(t_0)$, and t_m is defined as the first timeslot after t_0 such that $u(t_m)=h'_\pi(t_m)$. Due to the strict inequality relationship of the cost functions given in the corollary, and because $0<\gamma<1$, we can conclude that π is not optimal.

Appendix C
Approximating a General Cost Function with Constant-Plus-Exponential Cost Function

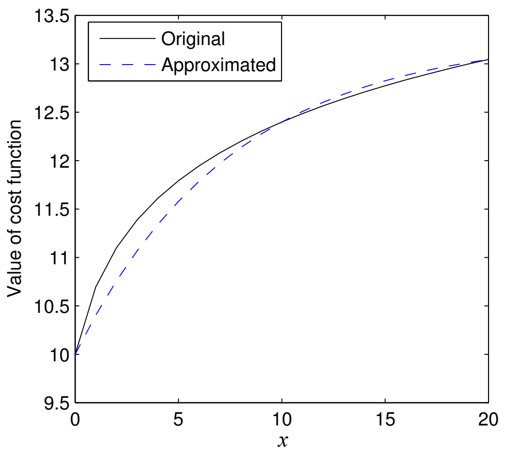
We only focus on the migration cost function $b(x)$, because $b(x)$ and $c(y)$ have the same form. For a given cost function $f(x)$, where $x\geq 0$, we would like to approximate it with the constant-plus-exponential cost function given by $b(x)=\beta_c+\beta_l\mu^x$. Note that although we force $b(0)=0$ by definition, we relax that restriction here and only consider the smooth part of the cost function for simplicity. We assume that this smoothness can be extended to $x=0$, so that $f(x)$ remains smooth at $x=0$. Under this definition, $f(x)$ may be non-zero at $x=0$.

Because $b(x)$ includes both a constant term and an exponential term, we cannot obtain an exact analytical expression to minimize a commonly used error function, such as the mean squared error. We can, however, solve for the parameters that minimize the error cost function numerically. To reduce the computational complexity compared to a fully numerical solution, we propose an approximate solution in this section.

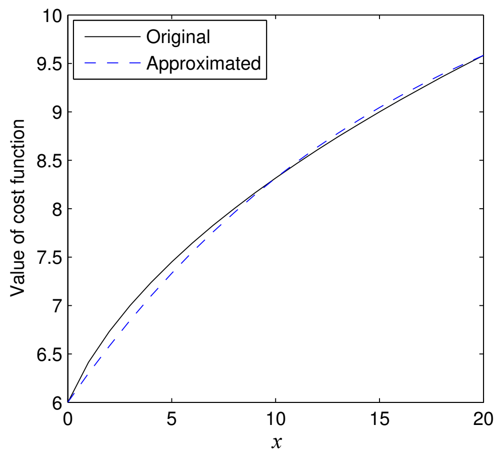
We are given an integer w that may be chosen according to practical considerations, and we solve for the parameters β_c, β_l, and μ according to the following system of equations:

$$\beta_c+\beta_l=f(0) \tag{A.1}$$

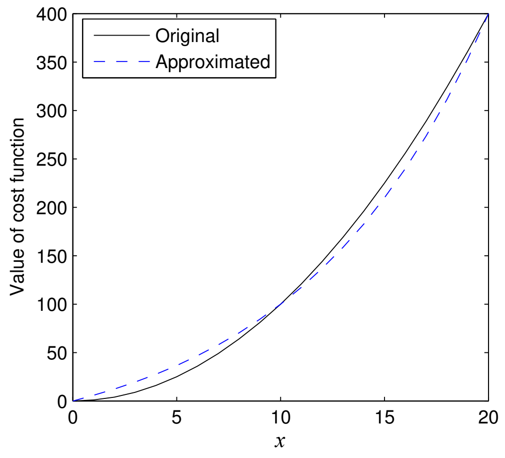
$$\beta_c+\beta_l\mu^w=f(w) \tag{A.2}$$



(a)



(b)



(c)

Figure A.1. Examples of approximating a general cost function with exponential cost function: (a) $f(x) = \ln(x+1) + 10$, (b) $f(x) = \sqrt{x+1} + 5$, (c) $f(x) = x^2$.

$$\beta_c + \beta_l \mu^{2w} = f(2w) \tag{A.3}$$

Subtracting (A.1) respectively from (A.2) and (A.3) and dividing the two results gives

$$\frac{\mu^{2w} - 1}{\mu^w - 1} = \frac{f(2w) - f(0)}{f(w) - f(0)} \tag{A.4}$$

subject to $\mu^w \neq 1$. We can then solve μ^w from (A.4) which gives

$$\mu^w = \frac{R \pm \sqrt{R^2 - 4(R-1)}}{2} \tag{A.5}$$

where $R \triangleq \frac{f(2w)-f(0)}{f(w)-f(0)} \geq 1$. It follows that we always have $R^2 - 4(R-1) \geq 0$ and $\mu^w \geq 0$. To guarantee that $\mu^w \neq 1$, we set $\mu^w = 1 \pm \epsilon_0$ if $|\mu^w - 1| < \epsilon_0$, where ϵ_0 is a small number and the sign is the same as the sign of $\mu^w - 1$. Then, we have

$$\mu = \left(\frac{R \pm \sqrt{R^2 - 4(R-1)}}{2} \right)^{\frac{1}{w}} \tag{A.6}$$

From which we can solve

$$\beta_c = \frac{f(0)\mu^w - f(w)}{\mu^w - 1} \tag{A.7}$$

$$\beta_l = \frac{f(w) - f(0)}{\mu^w - 1} \tag{A.8}$$

According to (A.6), we have two possible values of μ, and correspondingly two possible sets of values of β_c and β_l. To determine which is better, we compute the sum squared error $\sum_{x=0}^{2w} \left(f(x) - (\beta_c + \beta_l \mu^x) \right)^2$ for each parameter set, and choose the parameter set that produces the smaller sum squared error.

Some examples of approximation results are shown in Fig. A.1.

Appendix D
Proof of Theorem 2

Note that (6) is a difference equation [34]. Because we only migrate at states $\{n_k\}$, we have $a(d) = d$ for $d \in (n_{k-1}, n_k)$ $(\forall k)$. From (6), for $d \in (n_{k-1}, n_k)$, we have

$$V(d) = \delta_c + \delta_l \theta^d + \gamma \sum_{d_1 = d-1}^{d+1} P[d, d_1] \cdot V(d_1) \, . \tag{A.9}$$

We can rewrite (A.9) as

$$V(d) = \phi_1 V(d-1) + \phi_2 V(d+1) + \phi_3 + \phi_4 \theta^d . \tag{A.10}$$

This difference function has characteristic roots as expressed in (8). When $0 < \gamma < 1$, we always have $m_1 \neq m_2$. Fixing an index k, for $d \in (n_{k-1}, n_k)$, the homogeneous equation of (A.10) has general solution

$$V_h(d) = A_k m_1^d + B_k m_2^d \, . \tag{A.11}$$

To solve the non-homogeneous equation (A.10), we try a particular solution in the form of

$$V_p(d) = \begin{cases} D + H \cdot \theta^d & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2 \theta \neq 0 \\ D + Hd \cdot \theta^d & \text{if } 1 - \frac{\phi_1}{\theta} - \phi_2 \theta = 0 \end{cases} \tag{A.12}$$

where D and H are constant coefficients. By substituting (A.12) into (A.10), we get (9) and (10).

Because the expression in (A.10) is related to $d-1$ and $d+1$, the result also holds for the closed interval. Therefore, $V(d)$ can be expressed as (7) for $d \in [n_{k-1}, n_k]$ $(\forall k)$.

APPENDIX E
PROOF OF THEOREM 3

The proof is completed in three steps. First, we modify the states of the 2-D MDP in such a way that the aggregate transition probability from any state $(i'_0, j'_0) \neq (0,0)$ to ring $i_1 = i'_0 + 1$ (correspondingly, $i_1 = i'_0 - 1$) is $2.5r$ (correspondingly, $1.5r$). We assume that we use a given policy on both the original and modified 2-D MDPs, and show a bound on the difference in the discounted sum costs for these two MDPs. In the second step, we show that the modified 2-D MDP is equivalent to the distance-based MDP. This can be intuitively explained by the reason that the modified 2-D MDP has the same transition probabilities as the distance-based MDP when only considering the ring index i, and also, the one-timeslot cost $C_a(e(t))$ only depends on $\|e(t) - a(e(t))\|$ and $\|a(e(t))\|$, both of which can be determined from the ring indices of $e(t)$ and $a(e(t))$. The third step uses the fact that the optimal policy for the distance-based MDP cannot bring a higher discounted sum cost for the distance-based MDP (and hence the modified 2-D MDP) than any other policy. By utilizing the error bound found in the first step twice, we prove the result.

The detailed proof is given as follows.

Recall that among the neighbors (i', j') of a cell (i,j) in the 2-D offset-based MDP $\{e(t)\}$, when $i > 0$, we have two (or, correspondingly, one) cells with $i' = i - 1$, and two (or, correspondingly, three) cells with $i' = i + 1$. To "even out" the different number of neighboring cells, we define a new (modified) MDP $\{g(t)\}$ for the 2-D offset-based model, where the states are connected as in the original 2-D MDP, but the transition probabilities are different, as shown in Fig. A.2.

In the modified MDP $\{g(t)\}$, the transition probabilities starting from state $(0,0)$ to each of its neighboring cells have the same value r. For all the other states $(i,j) \neq (0,0)$, they are defined as follows:

- The transition probability to each of its neighbors with the same ring index i is r.
- If state (i,j) has two (correspondingly, one) neighbors in the lower ring $i-1$, then the transition probability to each of its neighbors in the lower ring is $\frac{1.5}{2}r$ (correspondingly, $1.5r$).
- if state (i,j) has two (correspondingly, three) neighbors in the higher ring $i+1$, then the transition probability to each of its neighbors in the higher ring is $\frac{2.5}{2}r$ (correspondingly, $\frac{2.5}{3}r$).

We denote the discounted sum cost from the original MDP $\{e(t)\}$ by $V(i,j)$, and denote that from the modified MDP $\{g(t)\}$ by $U(i,j)$, where (i,j) stands for the initial state in the discounted sum cost definition.

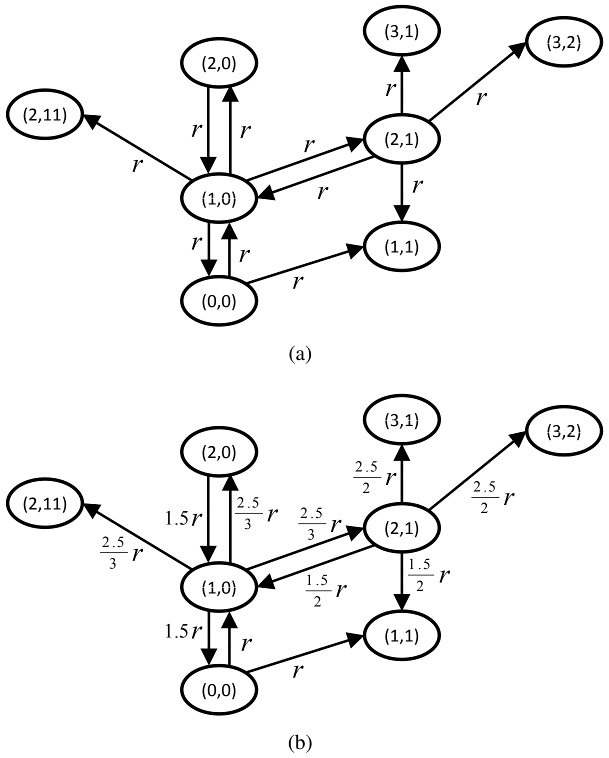


Figure A.2. Illustration of original and modified 2-D MDPs, only some exemplar states and transition probabilities are shown: (a) original, (b) modified.

Part I – Upper bound on the difference between $V(i,j)$ and $U(i,j)$ for a given policy π

Assume we have the same policy π for the original and modified MDPs. Then, the balance equation of $V(i,j)$ is

$$V(i,j) = C_a(i,j) + \gamma \left((1-6r)\, V(a(i,j)) + r \sum_{(i',j') \in \mathcal{N}(a(i,j))} V(i',j') \right) \tag{A.13}$$

where $a(i,j)$ is the new state after possible migration at state (i,j), and $\mathcal{N}(a(i,j))$ is the set of states that are neighbors of state $a(i,j)$.

For $U(i,j)$, we have

$$U(i,j) = C_a(i,j) + \gamma \Bigg((1-6r)\, U(a(i,j)) + \sum_{(i',j') \in \mathcal{N}(a(i,j))} P[g'(t) = a(i,j), g(t+1) = (i',j')] \cdot U(i',j') \Bigg) \tag{A.14}$$

where $P[g'(t) = a(i,j), g(t+1) = (i',j')]$ is the transition probability of the modified MDP $\{g(t)\}$ as specified earlier.

In the following, let $i_{a(i,j)}$ denote the ring index of $a(i,j)$. We define sets

$$\mathcal{N}^{-}(a(i,j)) = \left\{ (i',j') \in \mathcal{N}(a(i,j)) : i' = i_{a(i,j)} - 1 \right\}$$
$$\mathcal{N}^{+}(a(i,j)) = \left\{ (i',j') \in \mathcal{N}(a(i,j)) : i' = i_{a(i,j)} + 1 \right\}$$

to represent the neighboring states of $a(i,j)$ that are respectively in the lower and higher rings. We use $|\cdot|$ to denote the number of elements in a set.

Assume $|U(i,j) - V(i,j)| \leq \epsilon$ for all i and j, and the value

$$\begin{aligned}
&|U(i,j)-V(i,j)| \\
&=\gamma\Bigg|\left(1-6r\right)\left(U(a(i,j))-V(a(i,j))\right) \\
&\quad+\sum_{(i',j')\in\mathcal{N}(a(i,j))}P[g'(t)=a(i,j),g(t+1)=(i',j')]\cdot\left(U(i',j')-V(i',j')\right) \\
&\quad\pm 0.5\lambda_{i_{a(i,j)}}r\left(\frac{\sum_{(i',j')\in\mathcal{N}^+(a(i,j))}V(i',j')}{\left|\mathcal{N}^+\left(a(i,j)\right)\right|}-\frac{\sum_{(i',j')\in\mathcal{N}^-(a(i,j))}V(i',j')}{\left|\mathcal{N}^-\left(a(i,j)\right)\right|}\right)\Bigg| && \text{(A.15)} \\
&\le\gamma\epsilon+0.5\gamma r\left|\frac{\sum_{(i',j')\in\mathcal{N}^+(a(i,j))}V(i',j')}{\left|\mathcal{N}^+\left(a(i,j)\right)\right|}-\frac{\sum_{(i',j')\in\mathcal{N}^-(a(i,j))}V(i',j')}{\left|\mathcal{N}^-\left(a(i,j)\right)\right|}\right| && \text{(A.16)} \\
&\le\gamma\epsilon+0.5\gamma r\max_{i,j,j':(i+1,j)\in\mathcal{N}_2(i-1,j')}\left|\left(V(i+1,j)-V(i-1,j')\right)\right| && \text{(A.17)}
\end{aligned}$$

---

of ϵ is unknown for now. We subtract (A.13) from (A.14), and then take the absolute value, yielding (A.15), (A.16), and (A.17) which are explained below, where the set $\mathcal{N}_2\left(i,j\right)$ is the set of states that are two-hop neighbors of state (i,j), the variable $\lambda_{i_{a(i,j)}}=0$ when $i_{a(i,j)}=0$, and $\lambda_{i_{a(i,j)}}=1$ when $i_{a(i,j)}>0$.

The first two terms of (A.15) subtract the discounted sum cost of the original MDP $\{e(t)\}$ from that of the modified MDP $\{g(t)\}$, by assuming that both chains have the *same* transition probabilities specified by the modified MDP. The difference in their transition proabilities is captured by the last term of (A.15). There is no difference in the transition probabilities when $i_{a(i,j)}=0$, thus $\lambda_{i_{a(i,j)}}=0$ when $i_{a(i,j)}=0$.

In the following, we consider $i_{a(i,j)}>0$ and further explain the last term of (A.15). We first note that there is difference in the transition probabilities only when moving to the lower or higher ring:

- The *sum* probability of moving to the lower ring in $\{g(t)\}$ is by $0.5r$ smaller (or, correspondingly, greater) than that in $\{e(t)\}$.
- The *sum* probability of moving to the higher ring in $\{g(t)\}$ is by $0.5r$ greater (or, correspondingly, smaller) than that in $\{e(t)\}$.

Therefore, the transition probablity difference for each neighboring state in the lower (or higher) ring is $\pm 0.5r$ divided by the number of neighbors in the lower (or higher) ring. Also note that the probablity difference for lower and higher rings have opposite signs. This explains the last term of (A.15), which captures the difference in the transition probabilities and its impact on the discounted sum costs.

The inequality in (A.16) is from the triangle inequality. We note that the subtraction in the last term of (A.15) only occurs on $V(i',j')$ values that are two-hop neighbors, so we have the inequality in (A.17) by replacing the value with the maximum.

From (A.17), we can obtain a balance equation for the upper bound of $|U(i,j)-V(i,j)|$, which is

$$\epsilon=\gamma\epsilon+0.5\gamma r\max_{i,j,j':(i+1,j)\in\mathcal{N}_2(i-1,j')}\left|\left(V(i+1,j)-V(i-1,j')\right)\right| \tag{A.18}$$

Because $0<\gamma<1$, the value of $V(i,j)$ converges after a number of iterations according to (A.13) [21, Chapter 6], so $|(V(i+1,j)-V(i-1,j))|$ also converges, and the value of ϵ can be solved by

$$\epsilon_V=\frac{\gamma r\max_{i,j,j':(i+1,j)\in\mathcal{N}_2(i-1,j')}\left|\left(V(i+1,j)-V(i-1,j')\right)\right|}{2(1-\gamma)} \tag{A.19}$$

Note that the above argument also applies when interchanging V and U, which means that an alternative upper bound of the cost is

$$\epsilon_U=\frac{\gamma r\max_{i,j,j':(i+1,j)\in\mathcal{N}_2(i-1,j')}\left|\left(U(i+1,j)-U(i-1,j')\right)\right|}{2(1-\gamma)} \tag{A.20}$$

The upper bound can also be expressed as $\epsilon=\min\left\{\epsilon_V,\epsilon_U\right\}$, but either ϵ_V or ϵ_U may have the smaller value.

Part II – Optimal policy for the modified 2-D MDP $\{g(t)\}$ is equivalent to the optimal policy for the distance-based MDP $\{d(t)\}$

We note that the optimal policy of an MDP can be found from value iteration [21, Chapter 6]. For the modified MDP $\{g(t)\}$, we initialize with a never migrate policy, which gives the initial value function $U_0(i,j)=c\left(i\right)$, satisfying $U_0(i,j)=U_0(i,j')$ for all i and $j\neq j'$.

Suppose $U_n(i,j)=U_n(i,j')$ for all i and $j\neq j'$. In each iteration, we use the following equation to obtain the new value function and the corresponding actions for each i and j:

$$U_{n+1}(i,j)=\min_a\Bigg\{C_a(i,j)+\gamma\sum_{i'}\sum_{j'}P[g'(t)=a(i,j),g(t+1)=(i',j')]\cdot U_n(i',j')\Bigg\} \tag{A.21}$$

From the hexagon model in Fig. 5, we can see that for ring indices i and i', where $i'<i$, we can always reach from state (i,j) to a state in ring i' with $i-i'$ hops, regardless of the index j. In the $(n+1)$th iteration, if it is optimal to migrate from state (i,j) to a state with ring index i', then the migration destination must be $i-i'$ hops away from origin state, because $U_n(i',j)=U_n(i',j')$ for all i' and $j\neq j'$, it cannot be beneficial to migrate to somewhere farther away.

Further, if it is optimal to migrate at a state (i, j) to a state in ring i', it must be optimal to migrate at states (i, j) for all j to a state (which may not be the same state) in ring i', bringing the same cost, i.e. $U_{n+1}(i, j) = U_{n+1}(i, j')$ for $j \neq j'$. This is due to the symmetry of cost functions (in the sense that $U_n(i, j) = U_n(i, j')$ for all i and $j \neq j'$) and symmetry of transition probabilities (in the sense that the sum probability of reaching ring $i-1$ from any state in ring i is the same, and the sum probability of reaching ring $i+1$ from any state in ring i is also the same). Similarly, if it is optimal not to migrate at a state (i, j), then it is optimal not to migrate at states (i, j) for all j, which also brings $U_{n+1}(i, j) = U_{n+1}(i, j')$ for any $j \neq j'$.

Because the value iteration converges to the optimal policy and its corresponding cost as $n \to \infty$, for the optimal policy of the modified MDP $\{g(t)\}$, we have the same discounted sum cost for states in the same ring, i.e. $U^*(i, j) = U^*(i, j')$ for all i and $j \neq j'$. Meanwhile, for a given i, the optimal actions $a^*(i, j)$ and $a^*(i, j')$ for any $j \neq j'$ have the same ring index $i_{a^*(i,j)}$.

Since the optimal actions $a^*(i, j)$ only depend on the ring index i and the ring index of $a^*(i, j)$ does not change with j, the optimal policy for $\{g(t)\}$ can be directly mapped to a policy for the distance-based MDP $\{d(t)\}$. A policy for $\{d(t)\}$ can also be mapped to a policy for $\{g(t)\}$ by considering the shortest path between different states in $\{g(t)\}$, as discussed in Section V-B. This implies that there is a one-to-one mapping between the optimal policy for $\{g(t)\}$ and a policy for $\{d(t)\}$, because the optimal policy for $\{g(t)\}$ also only migrates along the shortest path between states.

We now show that the policy for $\{d(t)\}$ obtained from the optimal policy for $\{g(t)\}$ is optimal for $\{d(t)\}$. To find the optimal policy for $\{d(t)\}$, we can perform value iteration according to the following update equation:

$$U_{n+1}(i) = \min_a \Bigg\{ C_a(i) + \gamma \sum_{i'} \left(\sum_{j'} P[g(t) = a(i), g(t+1) = (i', j')] \right) U_n(i') \Bigg\} \tag{A.22}$$

The difference between (A.21) and (A.22) is that (A.22) does not distinguish the actions and value functions with different j indices. Recall that for the modified 2-D MDP $\{g(t)\}$, we have $U_n(i, j) = U_n(i, j')$ for all n, i and $j \neq j'$, so the index j can be natually removed from the value functions. Further, if it is optimal to migrate at state (i, j) to a state in ring i', it must be optimal to migrate at states (i, j) for all j to a state (which may not be the same state) in ring i'. The migration cost for different j are the same because they all follow the shortest path from state (i, j) to ring i'. If it is optimal not to migrate at state (i, j), then it is optimal not to migrate at states (i, j) for all j. Therefore, we can also remove the j index associated with the actions, without affecting the value function. It follows that the optimal policy for $\{g(t)\}$ is equivalent to the optimal policy for $\{d(t)\}$, both bringing the same value functions (discounted sum costs).

Part III – Error bound for distance-based approximation

By now, we have shown the upper bound on the discounted sum cost difference between the original and modified 2-D MDPs $\{e(t)\}$ and $\{g(t)\}$, when both MDPs use the same policy. We have also shown that the optimal policy for the modified 2-D MDP $\{g(t)\}$ is equivalent to the optimal policy for the distance-based MDP $\{d(t)\}$. Note that the true optimal cost is obtained by solving for the optimal policy for the original 2-D MDP $\{e(t)\}$, and the approximate optimal cost is obtained by applying the optimal policy for $\{d(t)\}$ to $\{e(t)\}$. In the following, we consider the upper bound on the difference between the true and approximate optimal discounted sum costs.

We start with a (true) optimal policy π^*_{true} for $\{e(t)\}$, denote the discounted sum costs from this policy as $V_{\pi^*_{\text{true}}}(i, j)$. When using the same policy on $\{g(t)\}$, the difference between the costs $U_{\pi^*_{\text{true}}}(i, j)$ and $V_{\pi^*_{\text{true}}}(i, j)$ satisfies the upper bound given in (A.19), i.e.

$$U_{\pi^*_{\text{true}}}(i, j) - V_{\pi^*_{\text{true}}}(i, j) \leq \epsilon_{V_{\pi^*_{\text{true}}}} \tag{A.23}$$

Since $V_{\pi^*_{\text{true}}}(i, j)$ are the optimal costs, we have

$$\max_{i,j,j':(i+1,j)\in\mathcal{N}_2(i-1,j')} \left| \left(V_{\pi^*_{\text{true}}}(i+1, j) - V_{\pi^*_{\text{true}}}(i-1, j') \right) \right| \leq \max_x \left\{ b(x+2) - b(x) \right\} \tag{A.24}$$

because, otherwise, there exists at least one pair of states $(i+1, j)$ and $(i-1, j')$ for which it is beneficial to migrate from state $(i+1, j)$ to state $(i-1, j')$, according to a 2-D extension of (11). Note that further migration may occur at state $(i-1, j')$, thus we take the maximum over x in the expression. The cost after performing such migration is upper bounded by (A.24), which contradicts with the fact that π^*_{true} is optimal.

Define

$$\epsilon_c = \frac{\gamma r \max_x \left\{ b(x+2) - b(x) \right\}}{2(1-\gamma)} \tag{A.25}$$

From (A.19), (A.23) and (A.24), we have

$$U_{\pi^*_{\text{true}}}(i, j) - V_{\pi^*_{\text{true}}}(i, j) \leq \epsilon_c \tag{A.26}$$

According to the equivalence of $\{g(t)\}$ and $\{d(t)\}$, we know that the optimal policy π^*_{appr} of $\{d(t)\}$ is also optimal for $\{g(t)\}$. Hence, we have

$$U_{\pi^*_{\text{appr}}}(i, j) \leq U_{\pi^*_{\text{true}}}(i, j) \tag{A.27}$$

because the cost from the optimal policy cannot be higher than the cost from any other policy.

When using the policy π^*_{appr} on $\{e(t)\}$, we get costs $V_{\pi^*_{\text{appr}}}(i, j)$. From (A.20), and because $U_{\pi^*_{\text{appr}}}(i, j)$ is the optimal cost for $\{g(t)\}$, we have

$$V_{\pi^*_{\text{appr}}}(i, j) - U_{\pi^*_{\text{appr}}}(i, j) \leq \epsilon_{U_{\pi^*_{\text{appr}}}} \leq \epsilon_c \tag{A.28}$$

From (A.26), (A.27), and (A.28), we get

$$V_{\pi^*_{\text{appr}}}(i, j) - V_{\pi^*_{\text{true}}}(i, j) \leq 2\epsilon_c \tag{A.29}$$

which completes the proof.

Appendix F
Proof of Theorem 4

We note that in (18), $m_n(\tau)$ and $m'_n(\tau)$ are both random variables respectively representing the total number of users associated to BS (located in cell) n in slot τ and the number of users that have moved out of cell n at the end of slot τ. The values of $m_n(\tau)$ and $m'_n(\tau)$ are random due to the randomness of user mobility.

According to the mobility model, each user moves out of its current cell at the end of a timeslot with probability $6r$, where we recall that $r \leq \frac{1}{6}$ by definition. Hence, under the condition that $m_n(\tau) = k$, $m'_n(\tau)$ follows the binomial distribution with parameters k and $6r$. Thus, the conditional probability

$$\Pr\left\{m'_n(\tau) = k' | m_n(\tau) = k\right\} = \left(\begin{array}{c} k \\ k' \end{array}\right) (6r)^{k'} (1-6r)^{k-k'} \tag{A.30}$$

for all $0 \leq k' \leq k$.

From the expression of the mean of binomially distributed random variables, we know that the conditional expectation

$$\mathbb{E}\left\{m'_n(\tau) | m_n(\tau)\right\} = 6r \cdot m_n(\tau). \tag{A.31}$$

Combining (18)–(20), we have

$$\hat{r} = \frac{1}{6N_{\mathrm{BS}}T_w} \sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \frac{m'_n(\tau)}{m_n(\tau)}. \tag{A.32}$$

We then have

$$\begin{aligned}
\mathbb{E}\{\hat{r}\} &= \mathbb{E}\left\{\frac{1}{6N_{\mathrm{BS}}T_w} \sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \frac{m'_n(\tau)}{m_n(\tau)}\right\} \\
&= \frac{1}{6N_{\mathrm{BS}}T_w} \sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \mathbb{E}\left\{\frac{\mathbb{E}\left\{m'_n(\tau)|m_n(\tau)\right\}}{m_n(\tau)}\right\} \\
&= \frac{1}{6N_{\mathrm{BS}}T_w} \sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \mathbb{E}\left\{\frac{6r \cdot m_n(\tau)}{m_n(\tau)}\right\} \\
&= \frac{r}{N_{\mathrm{BS}}T_w} N_{\mathrm{BS}}T_w \\
&= r
\end{aligned}$$

where the second equality follows from the law of iterated expectations.

Appendix G
Proof of Theorem 5

We define $\mathcal{M} \triangleq \{m_n(\tau) : \forall n \in \mathcal{N}_{\mathrm{BS}}, \tau \in [t-T_w, t-1]\}$ as the set of number of users at all BSs and all timeslots considered in the estimation. We first introduce the following lemma.

Lemma 1. *Assume that Assumption 1 is satisfied and each user follows 2-D random walk (defined at the beginning of Section VI-D1) with parameter r. The variance of estimator $\hat{r}$, under the condition that $\mathcal{M}$ is given, is*

$$\mathrm{Var}\{\hat{r}|\mathcal{M}\} = \frac{r(1-6r)}{6N_{BS}^2T_w^2} \left(\sum_{n\in\mathcal{N}_{BS}} \sum_{\tau=t-T_w}^{t-1} \frac{1}{m_n(\tau)}\right) \tag{A.33}$$

where the conditional variance $\mathrm{Var}\{\hat{r}|\mathcal{M}\}$ *is defined as*

$$\mathrm{Var}\{\hat{r}|\mathcal{M}\} \triangleq \mathbb{E}\left\{\hat{r}^2|\mathcal{M}\right\} - \left(\mathbb{E}\left\{\hat{r}|\mathcal{M}\right\}\right)^2. \tag{A.34}$$

Proof. Taking the conditional expectation on both sides of (A.32), we have

$$\begin{aligned}
\mathbb{E}\{\hat{r}|\mathcal{M}\} &= \mathbb{E}\left\{\frac{1}{6N_{\mathrm{BS}}T_w} \sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \frac{m'_n(\tau)}{m_n(\tau)}\middle|\mathcal{M}\right\} \\
&= \frac{1}{6N_{\mathrm{BS}}T_w} \sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \frac{\mathbb{E}\left\{m'_n(\tau)|m_n(\tau)\right\}}{m_n(\tau)} \\
&= \frac{1}{6N_{\mathrm{BS}}T_w} \sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \frac{6r \cdot m_n(\tau)}{m_n(\tau)} \\
&= \frac{r}{N_{\mathrm{BS}}T_w} N_{\mathrm{BS}}T_w \\
&= r
\end{aligned} \tag{A.35}$$

where the second equality is because $m'_n(\tau)$ is independent of $m_n(\tilde{\tau})$, $m_{\tilde{n}}(\tau)$, and $m_{\tilde{n}}(\tilde{\tau})$ (where $\tilde{n} \neq n$ and $\tilde{\tau} \neq \tau$) when $m_n(\tau)$ is given, according to Assumption 1. We thus have $\left(\mathbb{E}\{\hat{r}|\mathcal{M}\}\right)^2 = r^2$.

We focus on evaluating $\mathbb{E}\left\{\hat{r}^2|\mathcal{M}\right\}$ in the following. From (A.32), we have

$$\hat{r}^2 = \frac{1}{36N_{\mathrm{BS}}^2T_w^2} \left(\sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \frac{m'_n(\tau)}{m_n(\tau)}\right)^2 \tag{A.36}$$

$$= \frac{1}{36N_{\mathrm{BS}}^2T_w^2} \left(\sum_{n\in\mathcal{N}_{\mathrm{BS}}} \sum_{\tau=t-T_w}^{t-1} \frac{(m'_n(\tau))^2}{(m_n(\tau))^2} + \sum_{\substack{n_1,n_2\in\mathcal{N}_{\mathrm{BS}};\\ \tau_1,\tau_2\in[t-T_w,t-1];\\ n_1\neq n_2 \text{ and/or } \tau_1\neq\tau_2}} \frac{m'_{n_1}(\tau_1)}{m_{n_1}(\tau_1)} \cdot \frac{m'_{n_2}(\tau_2)}{m_{n_2}(\tau_2)}\right). \tag{A.37}$$

We now consider the two parts in (A.37). From the proof of Theorem 4, we know that $m'_n(\tau)$ follows the binomial distribution when $m_n(\tau)$ is given. Further, when $m_n(\tau)$ is given, $m'_n(\tau)$ is independent of $m_n(\tilde{\tau})$, $m_{\tilde{n}}(\tau)$, and $m_{\tilde{n}}(\tilde{\tau})$ (where $\tilde{n} \neq n$ and $\tilde{\tau} \neq \tau$), according to Assumption 1. Thus, we have

$$\begin{aligned}
\mathbb{E}\left\{\frac{(m'_n(\tau))^2}{(m_n(\tau))^2}\middle|\mathcal{M}\right\} &= \frac{\mathbb{E}\left\{(m'_n(\tau))^2|m_n(\tau)\right\}}{(m_n(\tau))^2} \\
&= \frac{m_n(\tau)\cdot 6r\cdot(1-6r) + 36r^2(m_n(\tau))^2}{(m_n(\tau))^2} \\
&= \frac{6r\cdot(1-6r)}{m_n(\tau)} + 36r^2
\end{aligned} \tag{A.38}$$

where the second equality is a known result for binomially distributed random variables. We also have

$$\begin{aligned}
&\mathbb{E}\left\{\frac{m'_{n_1}(\tau_1)}{m_{n_1}(\tau_1)} \cdot \frac{m'_{n_2}(\tau_2)}{m_{n_2}(\tau_2)}\middle|\mathcal{M}\right\} \\
&= \mathbb{E}\left\{\frac{m'_{n_1}(\tau_1)}{m_{n_1}(\tau_1)}\middle|m_{n_1}(\tau_1)\right\} \cdot \mathbb{E}\left\{\frac{m'_{n_2}(\tau_2)}{m_{n_2}(\tau_2)}\middle|m_{n_2}(\tau_2)\right\}
\end{aligned}$$

$$= \frac{\mathbb{E}\left\{m'_{n_1}(\tau_1)|m_{n_1}(\tau_1)\right\}}{m_{n_1}(\tau_1)} \cdot \frac{\mathbb{E}\left\{m'_{n_2}(\tau_2)|m_{n_2}(\tau_2)\right\}}{m_{n_2}(\tau_2)}$$
$$= 36r^2 \quad \text{(A.39)}$$

for $n_1 \neq n_2$ and/or $\tau_1 \neq \tau_2$, where the first equality follows from the fact that $m'_{n_1}(\tau_1)$ and $m'_{n_2}(\tau_2)$ are independent when $m_{n_1}(\tau_1)$ and $m_{n_2}(\tau_2)$ are given (according to Assumption 1), the last equality follows from (A.31).

We now take the conditional expectation on both sides of (A.37), and substitute corresponding terms with (A.38) and (A.39). This yields

$$\mathbb{E}\left\{\hat{r}^2|\mathcal{M}\right\}$$
$$= \frac{1}{36N_{\mathrm{BS}}^2T_w^2}\left(\sum_{n\in\mathcal{N}_{\mathrm{BS}}}\sum_{\tau=t-T_w}^{t-1}\mathbb{E}\left\{\frac{(m'_n(\tau))^2}{(m_n(\tau))^2}\middle|\mathcal{M}\right\}\right.$$
$$\left. + \sum_{\substack{n_1,n_2\in\mathcal{N}_{\mathrm{BS}};\\ \tau_1,\tau_2\in[t-T_w,t-1];\\ n_1\neq n_2 \text{ and/or } \tau_1\neq\tau_2}}\mathbb{E}\left\{\frac{m'_{n_1}(\tau_1)}{m_{n_1}(\tau_1)}\cdot\frac{m'_{n_2}(\tau_2)}{m_{n_2}(\tau_2)}\middle|\mathcal{M}\right\}\right)$$
$$= \frac{1}{36N_{\mathrm{BS}}^2T_w^2}\left(\sum_{n\in\mathcal{N}_{\mathrm{BS}}}\sum_{\tau=t-T_w}^{t-1}\left(\frac{6r\cdot(1-6r)}{m_n(\tau)}+36r^2\right)\right. \quad \text{(A.40)}$$
$$\left. + \sum_{\substack{n_1,n_2\in\mathcal{N}_{\mathrm{BS}};\\ \tau_1,\tau_2\in[t-T_w,t-1];\\ n_1\neq n_2 \text{ and/or } \tau_1\neq\tau_2}} 36r^2\right)$$
$$= \frac{1}{36N_{\mathrm{BS}}^2T_w^2}\left(6r\cdot(1-6r)\cdot\left(\sum_{n\in\mathcal{N}_{\mathrm{BS}}}\sum_{\tau=t-T_w}^{t-1}\frac{1}{m_n(\tau)}\right)\right. \quad \text{(A.41)}$$
$$\left. + N_{\mathrm{BS}}^2T_w^2\cdot 36r^2\right)$$
$$= \frac{r(1-6r)}{6N_{\mathrm{BS}}^2T_w^2}\left(\sum_{n\in\mathcal{N}_{\mathrm{BS}}}\sum_{\tau=t-T_w}^{t-1}\frac{1}{m_n(\tau)}\right)+r^2. \quad \text{(A.42)}$$

Subtracting $(\mathbb{E}\left\{\hat{r}|\mathcal{M}\right\})^2 = r^2$ from the above yields the result. □

Lemma 1 gives the conditional variance of estimator $\hat{r}$ when the values of $m_n(\tau)$ are given. It is not straightforward to remove the condition, because it is hard to find the stationary distribution of user locations in a hexagonal 2-D mobility model with a finite number of cells. We note that the set of $m_n(\tau)$ values represents the set of samples in our estimation problem. In standard estimation problems, the sample size is usually deterministic, while it is random in our problem due to random user locations. This causes the difficulty in finding the unconditional variance of our estimator.

However, Lemma 1 is important because it gives us a sense on how large the gap between $\hat{r}$ and r is, provided that each user precisely follows the random walk mobility model. It also enables us to proof Theorem 5.

Proof. (**Theorem 5**) As discussed in Section VI-B, we assume that $m_n(\tau) \neq 0$ for all n and τ. Thus, we always have $\frac{1}{m_n(\tau)} \leq 1$ (since $m_n(\tau)$ is a positive integer) and

$$\sum_{n\in\mathcal{N}_{\mathrm{BS}}}\sum_{\tau=t-T_w}^{t-1}\frac{1}{m_n(\tau)} \leq N_{\mathrm{BS}}T_w. \quad \text{(A.43)}$$

We also have the following bound:

$$r(1-6r) \leq \frac{1}{24} \quad \text{(A.44)}$$

for any $r \in [0, \frac{1}{6}]$. The law of total variance gives

$$\mathrm{Var}\{\hat{r}\} = \mathbb{E}\left\{\mathrm{Var}\{\hat{r}|\mathcal{M}\}\right\} + \mathrm{Var}\{\mathbb{E}\left\{\hat{r}|\mathcal{M}\right\}\}$$

According to (A.35), $\mathbb{E}\left\{\hat{r}|\mathcal{M}\right\} = r$ which is a constant, thus $\mathrm{Var}\{\mathbb{E}\left\{\hat{r}|\mathcal{M}\right\}\} = 0$. Therefore, we have

$$\mathrm{Var}\{\hat{r}\} = \mathbb{E}\left\{\mathrm{Var}\{\hat{r}|\mathcal{M}\}\right\} \leq \max_{\mathcal{M}}\mathrm{Var}\{\hat{r}|\mathcal{M}\} \leq \frac{1}{144N_{\mathrm{BS}}T_w}$$

where the last inequality follows from substituting (A.43) and (A.44) into (A.33). □

Appendix H
Proof of Theorem 6

Recall that for the original distance-based MDP $\{d(t)\}$ and its discounted sum cost, we have the following according to (6):

$$V^*(d) = C_{a^*}(d) + \gamma\sum_{d(t+1)=0}^{N}P\left[d'(t), d(t+1)\right]V^*(d(t+1)) \quad \text{(A.45)}$$

where $d'(t) = a^*(d)$. The transition probability $P\left[d'(t), d(t+1)\right]$ is defined according to Fig. 3. We always have $P\left[d'(t), d(t+1)\right] = 0$ when $|d(t+1) - d'(t)| > 1$.

We note that the distance-based MDP following the actual (randomized) action $a(d)$ is equivalent to an MDP following the optimal action $a^*(d)$ with modified transition probabilities. Let $\{\xi(t)\}$ denote such a modified MDP. There is a one-to-one mapping between the states in $\{\xi(t)\}$ and the states in $\{d(t)\}$, but the transition probabilities of these two MDPs are different. For convenience of comparison later, we use $\{d(t)\}$ to denote the states of $\{\xi(t)\}$ and use $P_\xi[\cdot,\cdot]$ to denote the transition probability of the MDP $\{\xi(t)\}$. We have

$$P_\xi\left[d'(t), d(t+1)\right]$$
$$= \begin{cases} (1-p_0)\sum_{k\leq\Delta_t}\alpha_k + q\alpha_{\Delta_t+1}, & \text{if } d(t+1)=0\\ p_0\sum_{k\leq\Delta_t-1}\alpha_k + \nu\alpha_{\Delta_t} + q\alpha_{\Delta_t+1}, & \text{if } d(t+1)=1\\ p\alpha_{\Delta_t-1} + \nu\alpha_{\Delta_t} + q\alpha_{\Delta_t+1}, & \text{if } d(t+1)\in[2,N-3]\\ p\alpha_{\Delta_t-1} + \nu\alpha_{\Delta_t} + q\sum_{k\geq\Delta_t+1}\alpha_k, & \text{if } d(t+1)=N-2\\ p\alpha_{\Delta_t-1} + \nu\sum_{k\geq\Delta_t}\alpha_k, & \text{if } d(t+1)=N-1\\ p\alpha_{\Delta_t-1}, & \text{if } d(t+1)=N \end{cases} \quad \text{(A.46)}$$

where $\nu \triangleq 1-p-q$ and $\Delta_t \triangleq d(t+1) - d'(t)$. This transition probability expression is from the definition of α_k and the original distance-based MDP definition in Fig. 3. The above expression holds for $N \geq 5$; for smaller values of N, similar expressions can be derived and we omit the details.

By definition, $\tilde{V}(d)$ is equal to the discounted sum cost of this modified MDP $\{\xi(t)\}$. We have the following balance equation:

$$\tilde{V}(d) = C_{a^*}(d) + \gamma \sum_{d(t+1)=0}^{N} P_\xi \left[d'(t), d(t+1)\right] \tilde{V}(d(t+1)). \tag{A.47}$$

Similar to the first part of the proof of Theorem 3, we assume that $\left|\tilde{V}(d) - V^*(d)\right| \leq \epsilon$ for all d and the value of ϵ will be determined later. Let $\Psi(d(t+1)) \triangleq P_\xi\left[d'(t), d(t+1)\right] - P\left[d'(t), d(t+1)\right]$. We have

$$\begin{aligned}
&\left|\tilde{V}(d) - V^*(d)\right| \\
&= \gamma \left| \sum_{d(t+1)=0}^{N} P_\xi\left[d'(t), d(t+1)\right] \left(\tilde{V}(d(t+1)) - V^*(d(t+1))\right) \right. \\
&\qquad \left. + \sum_{d(t+1)=0}^{N} \Psi(d(t+1)) V^*(d(t+1)) \right| \\
&\leq \gamma\epsilon + \gamma \left| \sum_{d(t+1)=0}^{N} \Psi(d(t+1)) V^*(d(t+1)) \right|
\end{aligned} \tag{A.48}$$

We consider the second term in (A.48) in the following. We first note that

$$\left|V^*\left(\tilde{d}\right) - V^*(d)\right| \leq \max_x \left\{ b\left(x + \left|\tilde{d} - d\right|\right) - b(x) \right\} \tag{A.49}$$

for any d and $\tilde{d} > d$, because otherwise, there exist some $\tilde{d}$ and d, for which (A.49) does not hold, where one can choose a new action that migrates from $\tilde{d}$ to d. When using this new action, (A.49) is satisfied (further migration may occur at state d, thus we take the maximum over x in the expression) and the discounted sum cost $V^*(d)$ is reduced. This contradicts with the Bellman's equation in (3) since $V^*(d)$ is for the optimal policy.

We also note that

$$\begin{aligned}
&\sum_{d(t+1)=0}^{N} \Psi(d(t+1)) \\
&= \sum_{d(t+1)=0}^{N} P_\xi\left[d'(t), d(t+1)\right] - \sum_{d(t+1)=0}^{N} P\left[d'(t), d(t+1)\right] \\
&= 1 - 1 = 0.
\end{aligned}$$

Therefore, the sum of all positive $\Psi(d(t+1))$ is equal to the sum of all negative $\Psi(d(t+1))$ in (A.48).

Because $\alpha_k = 0$ for all $|k| > K$, according to the definition of $P_\xi\left[d'(t), d(t+1)\right]$ in (A.46), we have $P_\xi\left[d'(t), d(t+1)\right] = 0$ when $\Delta_t + 1 < -K \leq 0$ or $\Delta_t - 1 > K \geq 0$. It follows that $P_\xi\left[d'(t), d(t+1)\right]$ can be non-zero only when $|\Delta_t| \leq K + 1$. For the original distance-based MDP, its transition probability $P\left[d'(t), d(t+1)\right]$ can be non-zero only when $|\Delta_t| \leq 1$ according to the definition. It follows that $\Psi(d(t+1))$ can only be negative when $|\Delta_t| \leq 1$.

Therefore, for any d and $\tilde{d}$ such that $\Psi(d(t+1)) > 0$ and $\Psi(\tilde{d}(t+1)) < 0$, we must have

$$\left|d(t+1) - \tilde{d}(t+1)\right| \leq K + 2. \tag{A.50}$$

Expanding the sum in (A.48) into separate positive and negative terms, we have

$$\begin{aligned}
&\left| \sum_{d(t+1)=0}^{N} \Psi(d(t+1)) V^*(d(t+1)) \right| \\
&= \left| \sum_{d(t+1):\Psi(d(t+1))>0} \Psi(d(t+1)) V^*(d(t+1)) \right. \\
&\quad \left. - \sum_{\tilde{d}(t+1):\Psi\left(\tilde{d}(t+1)\right)<0} \left|\Psi\left(\tilde{d}(t+1)\right)\right| \cdot V\left(\tilde{d}(t+1)\right) \right| \\
&\leq \max_x \left\{ b\left(x + K + 2\right) - b(x) \right\}
\end{aligned} \tag{A.51}$$

where the last inequality is from (A.49), (A.50), and the fact that

$$\begin{aligned}
&\sum_{d(t+1):\Psi(d(t+1))>0} \Psi(d(t+1)) \\
&\quad = \sum_{\tilde{d}(t+1):\Psi\left(\tilde{d}(t+1)\right)<0} \left|\Psi\left(\tilde{d}(t+1)\right)\right| \leq 1.
\end{aligned}$$

Substituting (A.51) into (A.48), we get

$$\left|\tilde{V}(d) - V^*(d)\right| \leq \gamma\epsilon + \gamma \max_x \left\{ b\left(x + K + 2\right) - b(x) \right\}$$

Because $0 < \gamma < 1$, $V^*(d)$ and $\tilde{V}(d)$ both converge to some fixed value (for each d) when iterating according to (A.45) and (A.47), respectively [21, Chapter 6]. Hence, $\left|\tilde{V}(d) - V^*(d)\right|$ also converges to some fixed value (for each d). We can therefore solve ϵ as

$$\epsilon = \frac{\gamma \max_x \left\{ b\left(x + K + 2\right) - b(x) \right\}}{1 - \gamma}$$

which completes the proof.

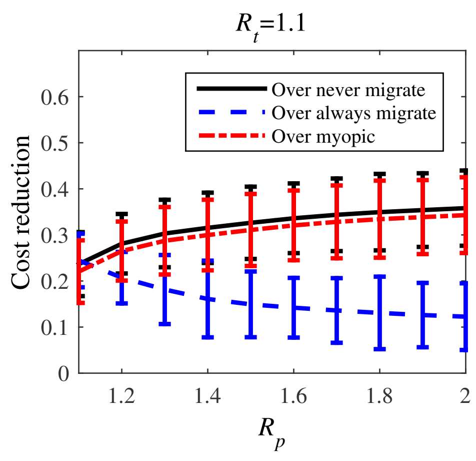
APPENDIX I
ADDITIONAL SIMULATION RESULTS WITH REAL-WORLD TRACES

To consider the overall performance under different parameter settings, we denote the cost of the proposed method as C and the cost of the baseline method under comparison as C_0, and we define the *cost reduction* as $(C_0 - C)/C_0$. Figs. A.3–A.6 show the cost reductions (averaged over the entire day) under different parameter settings. A positive cost reduction indicates that the proposed method performs better than the baseline, and the reverse is true for a negative cost reduction.

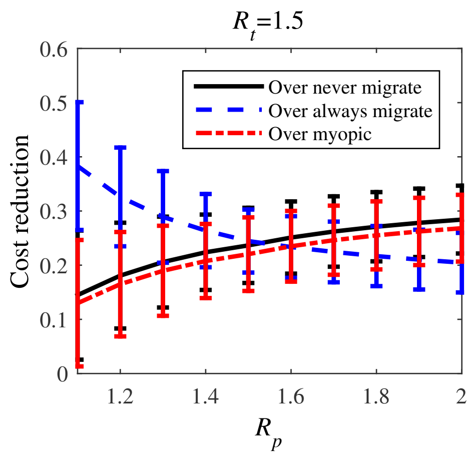
We can see that under the default number of ESs and capacity limit at each ES, the proposed approach is beneficial with average cost reductions of up to 44% compared to the never/always-migrate or myopic policies.

We also see that the cost reductions compared to never-migrate and myopic policies become small in the case where either the number of ESs is small or the capacity of each ES is low. In this case, it is hardly possible to migrate the service to a better location because the action space for migration is very small. Therefore, the proposed approach gives a similar cost as baseline policies, but still outperforms them as indicated by a positive cost reduction on average. The cost reduction compared to the always-migrate policy becomes slightly higher in the case of small action space, because the always-migrate policy always incurs migration cost even though the benefit of such migration is not obvious.

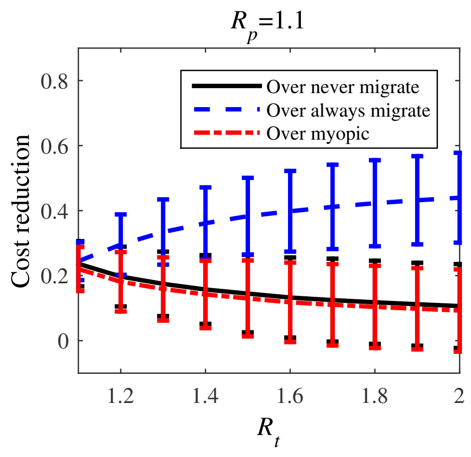
It is also interesting to see that while the average performance of the proposed approach is the best in almost all cases, there exist some instances in which the error bar (showing the standard deviation of the cost reductions computed on instantaneous costs) in Figs. A.3–A.6 goes below zero, indicating that the instantaneous cost of the proposed approach may be higher than some baseline approaches in some instances. This is possible because the proposed approach aims at minimizing the discounted sum cost defined in (1) and we use $\gamma = 0.9$ in the simulations. Thus, it may not minimize the instantaneous costs in all timeslots. The myopic policy, on the other hand, aims at minimizing the instantaneous cost. We note that setting $\gamma = 0$ makes our algorithm the same as the myopic policy. Hence, our algorithm is also capable of minimizing the instantaneous cost if the application desires so. However, in many practical scenarios, one would like to achieve a balance between the instantaneous cost and the long-term average cost, so that the service performance is reasonably good both instantaneously and on average. The discount factor γ acts as a control knob that adjusts this balance.



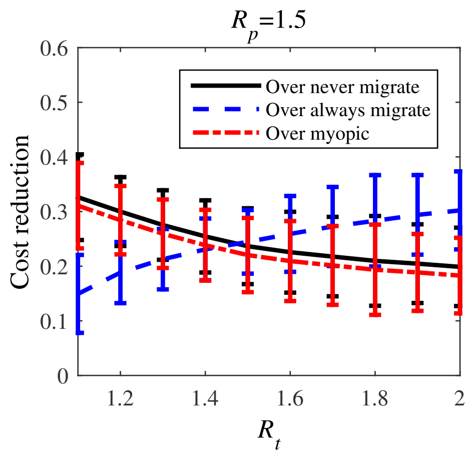
(a)



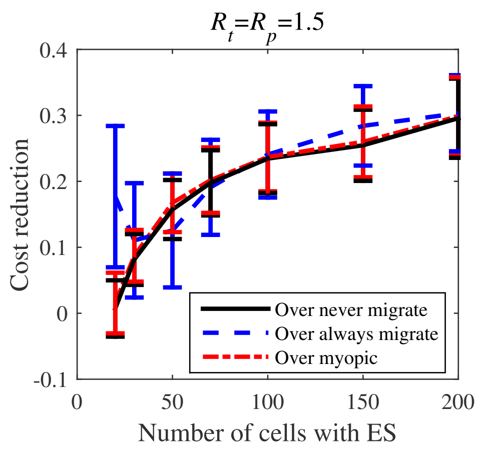
(b)



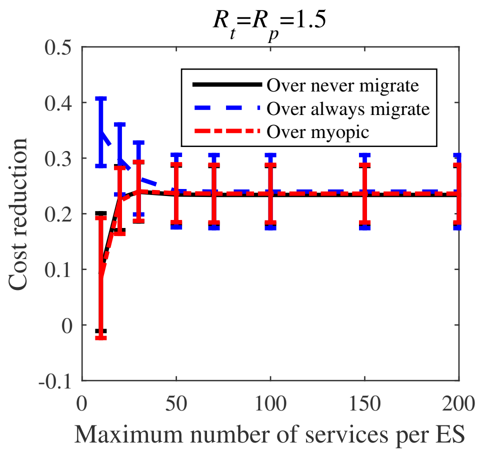
(c)



(d)

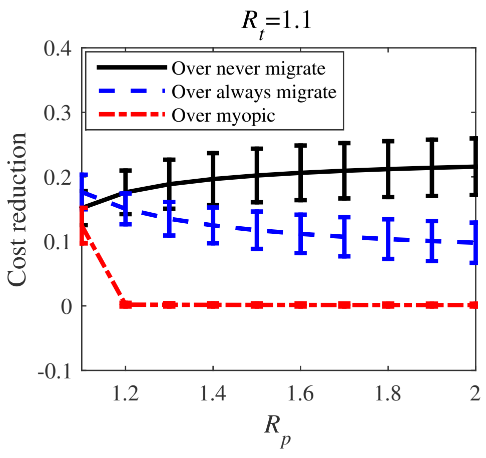


(e)

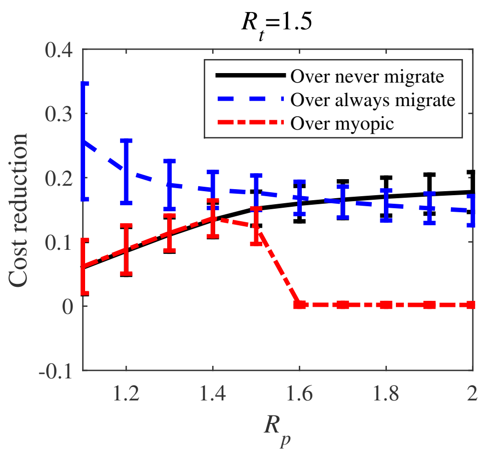


(f)

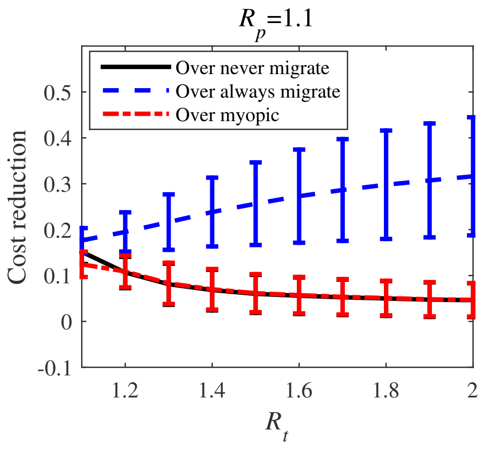
Figure A.3. (Hexagon, non-constant cost) Cost reduction (averaged over the entire day) compared to alternative policies in trace-driven simulation, the error bars denote the standard deviation (where we regard the cost reduction of instantaneous cost at different time of the day as samples): (a)–(b) cost reduction vs. different R_t, (c)–(d) cost reduction vs. different R_p, (e) cost reduction vs. different number of cells with ES, (f) cost reduction vs. different capacity limit of each ES (expressed as the maximum number of services allowed per ES).



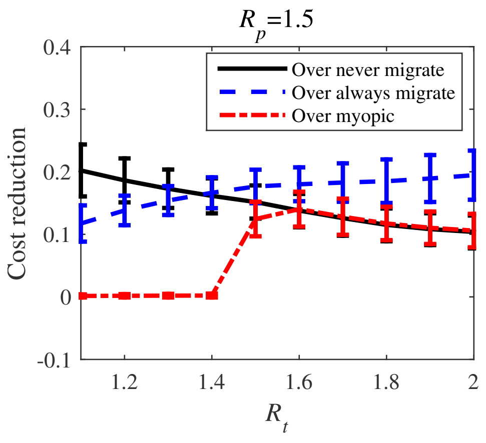
(a)



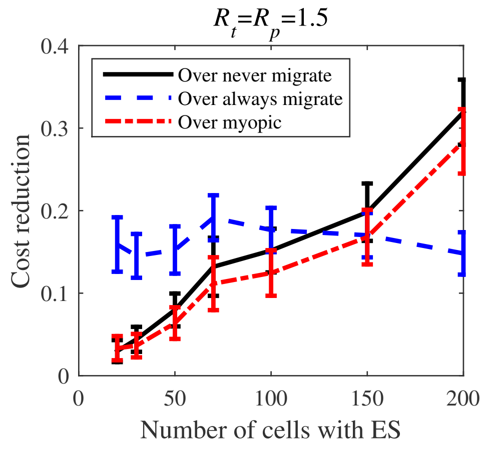
(b)



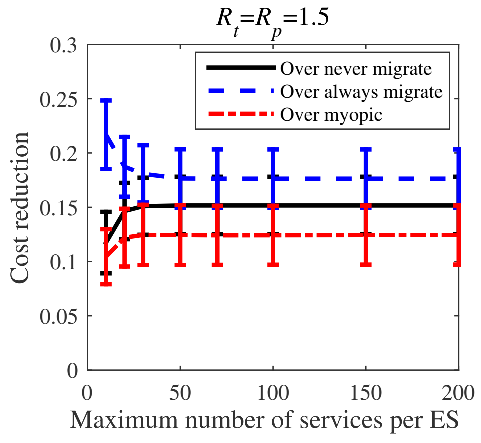
(c)



(d)



(e)



(f)

Figure A.4. (Hexagon, constant cost) Cost reduction (averaged over the entire day) compared to alternative policies in trace-driven simulation, the error bars denote the standard deviation (where we regard the cost reduction of instantaneous cost at different time of the day as samples): (a)–(b) cost reduction vs. different R_t, (c)–(d) cost reduction vs. different R_p, (e) cost reduction vs. different number of cells with ES, (f) cost reduction vs. different capacity limit of each ES (expressed as the maximum number of services allowed per ES).

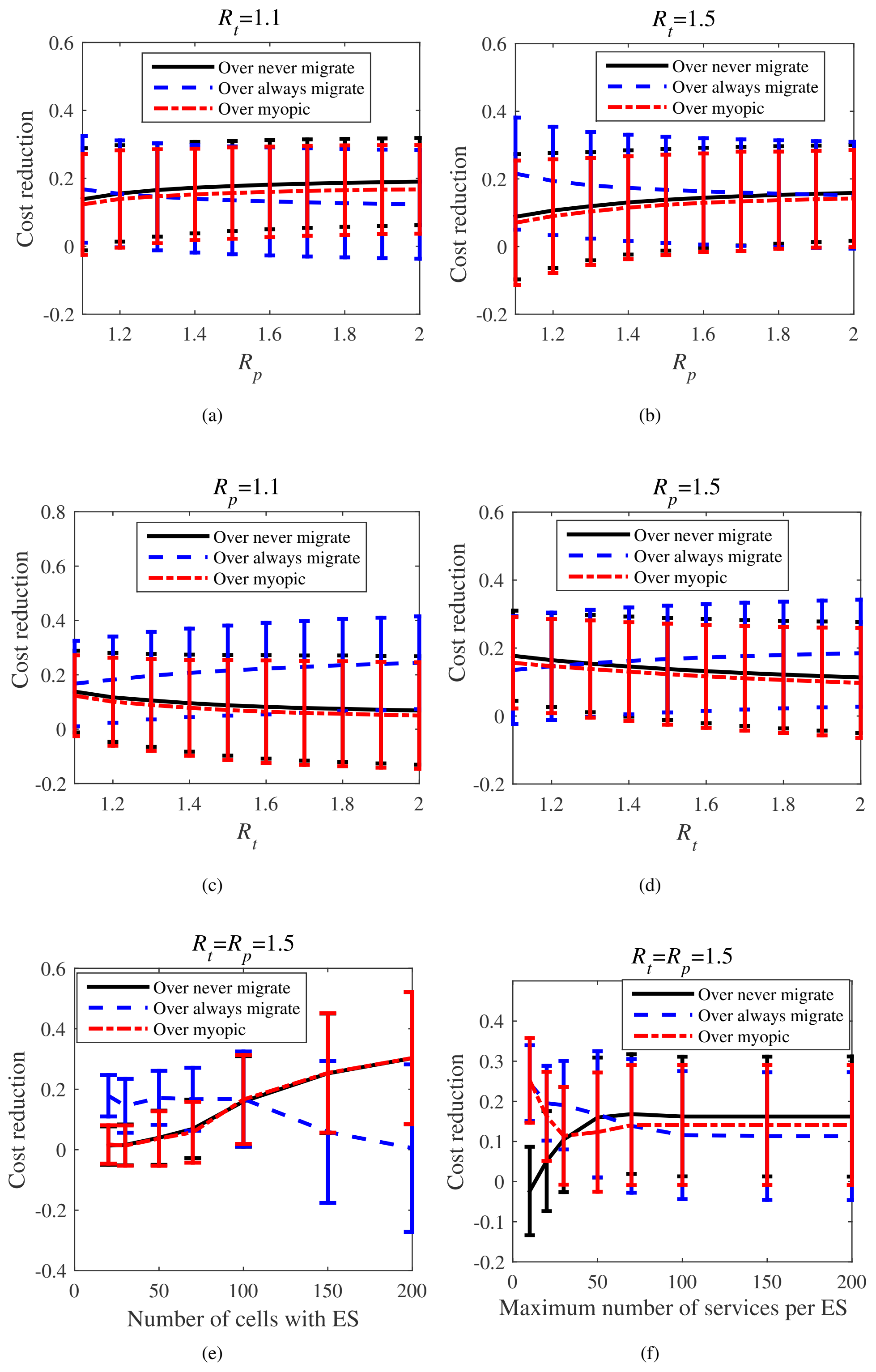
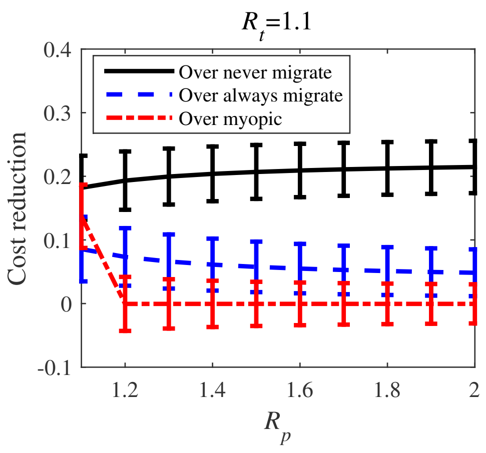
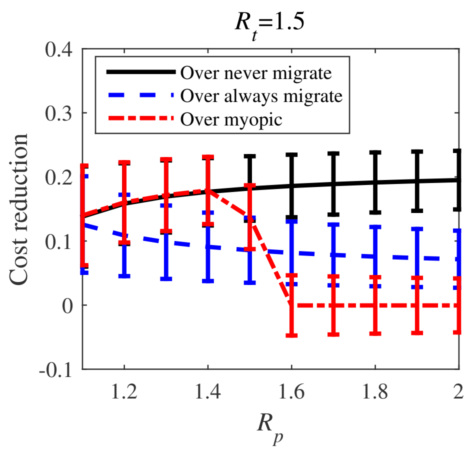


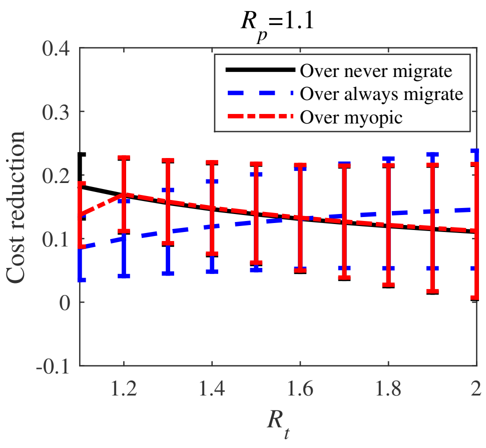
Figure A.5. (Real, non-constant cost) Cost reduction (averaged over the entire day) compared to alternative policies in trace-driven simulation, the error bars denote the standard deviation (where we regard the cost reduction of instantaneous cost at different time of the day as samples): (a)–(b) cost reduction vs. different R_t, (c)–(d) cost reduction vs. different R_p, (e) cost reduction vs. different number of cells with ES, (f) cost reduction vs. different capacity limit of each ES (expressed as the maximum number of services allowed per ES).



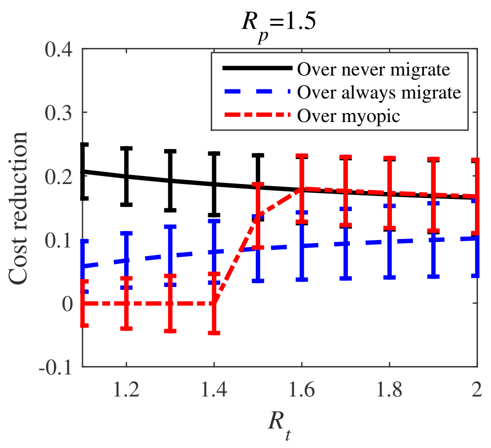
(a)



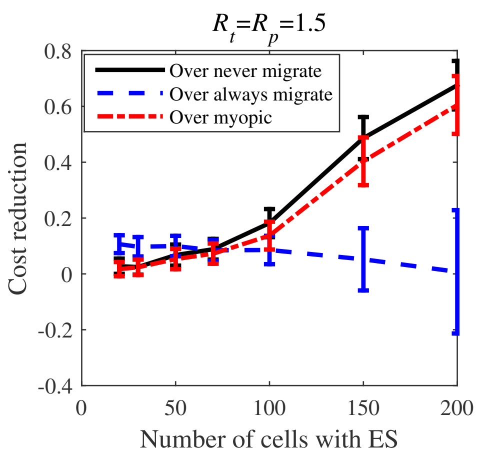
(b)



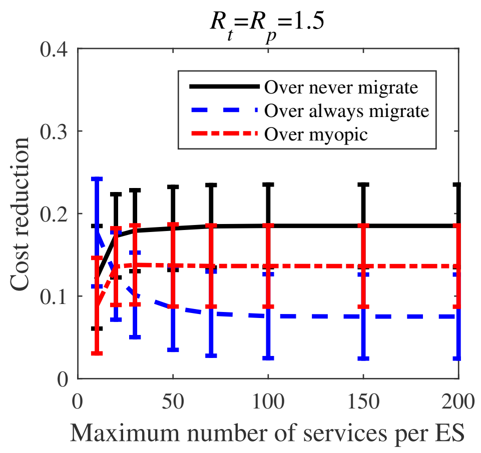
(c)



(d)



(e)



(f)

Figure A.6. (Real, constant cost) Cost reduction (averaged over the entire day) compared to alternative policies in trace-driven simulation, the error bars denote the standard deviation (where we regard the cost reduction of instantaneous cost at different time of the day as samples): (a)–(b) cost reduction vs. different R_t, (c)–(d) cost reduction vs. different R_p, (e) cost reduction vs. different number of cells with ES, (f) cost reduction vs. different capacity limit of each ES (expressed as the maximum number of services allowed per ES).