Online Learning based Uplink Scheduling in HetNets with Limited Backhaul Capacity

Zhenhua Han*, Haisheng Tan*, Rui Wang‡, Shaojie Tang§, Francis C.M. Lau†
*University of Science and Technology of China, P.R. China, †The University of Hong Kong, Hong Kong
‡The South University of Science and Technology of China, P.R. China, §University of Texas at Dallas, USA

***Abstract*—Heterogeneous cellular networks (HetNets) can significantly improve the spectrum efficiency, where low-power low-complexity base stations (Pico-BSs) are deployed inside the coverage of macro base stations (Macro-BSs). Due to cross-tier interference, joint detection of the uplink signals is widely adopted so that a Pico-BS can either detect the uplink signals locally or forward them to the Macro-BS for processing. The latter can achieve increased throughput at the cost of additional backhaul transmission. However, in existing literature the delay of the backhaul links was often neglected. In this paper, we study the delay-optimal uplink scheduling problem in HetNets with limited backhaul capacity. Local signal detection or joint signal detection is scheduled in a unified delay-optimal framework. Specifically, we first prove that the problem is NP-hard and then formulate it as a Markov Decision Process problem. We propose an efficient and effective algorithm, called `OLIUS`, that can deal with the exponentially growing state and action spaces. Furthermore, `OLIUS` is online learning based which does not require any prior statistical knowledge on user behavior or channel characteristics. We prove the convergence of `OLIUS` and derive an upper bound on its approximation error. Extensive experiments in various scenarios show that our algorithm outperforms existing methods in reducing delay and power consumption.**

I. Introduction

Heterogeneous networks (HetNets) have emerged as a promising network paradigm that can dramatically increase the spectrum efficiency of cellular networks. In HetNets, low-power low-complexity base stations (namely, Pico-BSs or Femto-BSs) are deployed inside the coverage of macro base stations (Macro-BSs), and the macro-BSs and pico-BSs can operate in the same frequency band simultaneously. This cell splitting technology benefits the cellular service in many ways including much reduced path loss and better spatial reuse of the radio spectrum. Fig. 1 shows an example of a HetNet with one macrocell and two picocells.

However, the potential of HetNets has not been fully exploited due to the severe cross-tier interference. Taking the uplink transmission as an example, the signals of the high-power macro-users may interfere with nearby picocells, resulting in interference on the pico-BSs that may even be stronger than the desired signals sent by the pico-users. Joint signal detection is a widely used technique to mitigate such cross-tier interference; it enables the raw signals received at pico-BSs to be forwarded to the macro-BS via the backhaul links (e.g., X2 links in Long Term Evolution (LTE) systems). Essentially, the whole network operates like a virtual MIMO (multiple-input-multiple-output) system and hence the cross-tier interference can be cancelled. Some existing works [1]–[5] studied how to utilize joint detection in HetNets, but they often made an unrealistic assumption that the backhaul links have unlimited capacity. Obviously, the backhaul capacity for joint signal detection is limited in reality because the forwarded signals are raw signals instead of demodulated signals. Although there have been a number of works [6]–[8] which studied the impact of limited backhaul capacity on joint signal detection, they mainly focused on throughout maximization in downlink transmission. With the development of Mobile Edge Computing (MEC) and the Internet of Things (IoT), the requirement for uplink transmission is dramatically increasing. Besides, the demand for high upload capacity, end-to-end delay[1] also plays an important role in the users' experience. For example, in MEC, mobile devices can offload tasks to edge servers which can meet harsh requirements on end-to-end delay (e.g., for real-time applications or online gaming). Moreover, due to the capacity limited backhaul links, joint detection will introduce more queuing delay when forwarding the signals. As existing uplink scheduling algorithms either

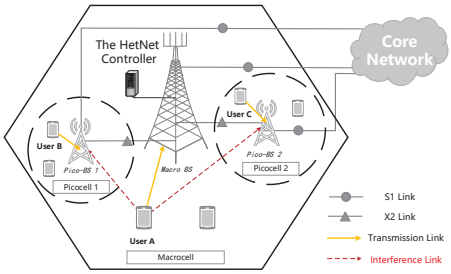


Fig. 1. A heterogeneous network with one macrocell and two picocells. The pico-BSs are connected to the macro-BS via the backhaul links (which is called the X2 interface in LTE systems). After uplink detection, the uplink data is further forwarded to the core network via the data links (called the S1 interface). The control action is determined by a centralized controller in the macro-BS.

Haisheng Tan is the Corresponding Author (email: hstan@ustc.edu.cn).

This work is supported partly by the National Key R&D Program of China 2017YFB1003000, China National Funds for Distinguished Young Scientists No. 61625205, NSFC Grants 61772489, 61502201, 61401192, 61520106007, Key Research Program of Frontier Sciences (CAS) No. QYZDY-SSW-JSC002, NSF ECCS-1247944, NSF CNS 1526638, Hong Kong RGC CRF Grant C7036-15G, and the Fundamental Research Funds for the Central U.

[1]In this paper, we refer to the end-to-end delay as the time from the appearance of a data packet to the time it reaches the core network.

ignore end-to-end delay [9] or neglect joint signal detection [10]–[12], a natural question one may ask is: *Is it possible to design an efficient algorithm for uplink scheduling in HetNets with limited backhaul capacity that minimizes average end-to-end delay?*

In this paper, we give an affirmative answer to the above question by proposing an efficient and effective algorithm, called OLIUS. Instead of using joint detection for all pico-users, we novelly introduce one additional control dimension to decide whether a pico-user's signal should be detected by pico-BS (local detection) or macro-BS (joint detection). When the cross-tier interference is weak, local detection could lead to a smaller delay; when the cross-tier interference is strong, joint detection could have better uplink throughput. Therefore, in the uplink scheduling problem, our algorithm should jointly decide (1) *User Selection*: which users should be selected for uplink transmission; (2) *Power Allocation*: which power level should be picked for each selected user; and (3) *Detection Mode Selection*: whether to use local detection at pico-BSs or to use joint detection at the macro-BS. Our contributions can be summarized as follows.

- We formulate the delay-optimal uplink scheduling problem in HetNets as an infinite horizon Markov Decision Process (MDP) problem. We prove that this problem is NP-hard.
- Generally, solving an MDP problem (e.g., value iteration and policy iteration) requires exponential time. We utilize Factored MDP (FMDP) and approximate $\mathbb{Q}$-function to exploit the special structure of the uplink scheduling problem to reduce the time complexity. OLIUS is a low-complexity online learning algorithm which does not require any prior statistical information on user behavior or channel characteristics. We prove the convergence of OLIUS and derive an upper bound on its approximation error.
- We conduct extensive simulations for different scenarios to evaluate the performance of OLIUS, and the results show that OLIUS can achieve significant performance gain over conventional scheduling algorithms in both delay and power consumption.

The remainder of this paper is organized as follows. In Sec. II, we present the system model and formulate the uplink scheduling problem and prove its NP-hardness. In Sec. III, we introduce our algorithm, OLIUS and analyze its convergence and error bounds. In Sec. IV, we present the experiment results. We conclude this paper in Sec. V.

II. System Model

A. Heterogeneous Network Model

We consider the uplink transmission in one macrocell with M picocells deployed within the macrocell's coverage. The pico-BSs are connected with the macro-BS via capacity limited backhaul links. We denote as R_m^β the capacity of the backhaul link from the m-th pico-BS to the macro-BS. Let $\mathcal{M} \triangleq \{1, \ldots, M\}$ and $\mathcal{K}_m \triangleq \{1, \ldots, N_m\} (\forall m \in \mathcal{M})$ denote the set of pico-BSs and the set of users served by the m^{th} pico-BS, respectively, where N_m is the number of users served by the m^{th} pico-BS. Let $\mathcal{K}_0 \triangleq \{1, \ldots, N_0\}$ denote the set of users served by the macro-BS. Without ambiguity, we call the 0^{th} BS and m^{th} $(m = 1, ..., M)$ BS the macro-BS and the m^{th} pico-BS, respectively.

The users and the pico-BSs are equipped with a single antenna, and the macro-BS has L antennas. The time dimension is organized by the scheduling time slots, each of which lasts for τ seconds. Let $h_{m,k}(t)$ denote the channel state information (CSI or channel gain) from the user $k \in \mathcal{K}_m$ to its service pico-BS (i.e., the m^{th} pico-BS); denote as $\mathbf{h}_{0,k}(t)$ the $L \times 1$ CSI vector from the macro-user $k \in \mathcal{K}_0$ to the macro-BS. As the macro-users use much higher power than the pico-users, the macro-users will introduce interference to the pico-BSs. We denote as $h_{0,k}^m$ the interference CSI from the macro-user $k \in \mathcal{K}_0$ to the pico-BS $m \in \mathcal{M}$. We consider the widely used model of block fading channel, where the CSI is quasi-static in each slot and independently and identically distributed (i.i.d.) over

different scheduling time slots. At the beginning of each time-slot, the HetNet controller should select exactly one user for each cell (which includes the pico-cells or and the marco-cell). Thus there are $M + 1$ users selected in each time-slot.

For higher spectrum efficiency, we consider the co-channel heterogeneous network [13], [14], where all cells share a common frequency band with a bandwidth of B Hz. Before each uplink time slot, the centralized controller (as shown in Fig. 1) collects all the system information, and makes a decision on the uplink scheduling. This decision is broadcasted to all the users via their service BSs. The scheduling decision includes which users are selected for uplink transmission, the transmission power and the detection model (i.e., local detection or joint detection). Due to the larger path loss attenuation, macro-users usually use much higher power for uplink transmission than that of the pico-users, and thus they may generate strong interference to the picocells. Because the pico-users use low power for uplink transmission, we assume the interference generated by the pico-users to other BSs can be ignored. As shown in Fig. 1, users A, B and C are chosen for uplink transmission in one time-slot. Because user A belongs to the macrocell, its high power signals will be received by pico-BS 1 and pico-BS 2 as the interference signals to users B and C. Moreover, users B's and C's interference to the other cells can be ignored, because their powers are much lower.

B. Uplink Detection Model

In this paper, we consider two possible paths through which the uplink data could be finally delivered to the core network of the cellular system:[2] (1) the uplink signals are detected locally at the pico-BSs and forwarded to the core network; (2) the uplink signals are first forwarded to the macro-BS for joint detection via capacity limited backhaul links, and then the decoded data is delivered to

[2]In LTE systems, the core network is called the Evolved Packet Core (EPC), and the uplink data should be forwarded to the Serving Gateway of EPC.

the core network from the macro-BS.[3] Note that joint detection needs the pico-BSs forward the raw signals to the macro-BS. As the raw signals without demodulation are in a large volume, they cannot be forwarded instantaneously. Although joint detection helps to increase the throughput of the scheduled users, it introduces more end-to-end delay due to limited backhaul capacity.

Let $S_m^\alpha(t) \in \mathcal{K}_m (\forall m \in \mathcal{M} \cup \{0\})$ denote the scheduled user in the m^{th} BSs (macro-BS or pico-BS) for uplink transmission in the t^{th} time-slot. Let $S_m^\beta(t) \in \{0,1\}$ $(\forall m \in \mathcal{M})$ denote the decision on whether or not to use joint detection. For instance, $S_m^\beta(t)$ equals 1 if joint detection is used, and 0 if local detection is used. Let $\mathbf{P} \triangleq diag(P_0, P_1, \ldots, P_M)$ denote the diagonal matrix of the transmission power where P_m denotes the transmission power of the scheduled user in the m^{th} BS. We assume $P_m \in \mathbb{P}_m$ where $\mathbb{P}_m$ is the discrete power levels that the users in the m^{th} BS can use.

Let $(\cdot)^T$ denote the transpose of a matrix or a vector. Let $\mathbf{h}_{mu}^{pico} = (h_{0,S_0^\alpha}^1, \ldots, h_{0,S_0^\alpha}^M)^T$ denote the vector of the interference CSI. Let $\mathbf{h}_{pu}^{pico} = diag(h_{1,S_1^\alpha}, \ldots, h_{M,S_M^\alpha})$ denote the diagonal matrix of CSI between the scheduled pico-users and their service pico-BSs. Denote the aggregated CSI matrix from all the scheduled users to all the BSs in the t^{th} time-slot as $\mathbf{H}(t)$; then it can be written as:

$$\mathbf{H}(t) = \left(\begin{array}{c|c} \mathbf{h}_{0,S_0^\alpha} & \mathbf{0} \\ \hline \mathbf{h}_{mu}^{pico} & \mathbf{h}_{pu}^{pico} \end{array} \right),$$

We call $\mathbf{H}(t)$ the global CSI at time-slot t.

Denote as $R_m(t)$ the data rate of the scheduled uplink user in the m^{th} BS. Formally, we derive this data rate in the following lemma.

Lemma 1. *The data rate of the scheduled uplink user in the m^{th} BS is*

$$R_m(t) \triangleq \begin{cases} \tau B \log_2 \left(1 + \frac{\xi}{\|\mathbf{w}_{m+1}^\dagger\|^2}\right), & \text{if } m = 0 \\ & \text{or } S_m^\beta = 1; \\ \tau B \log_2 \left(1 + \frac{\xi |h_{m,S_m^\alpha} P_m|^2}{1 + |h_{0,S_0^\alpha}^m P_0|^2}\right), & \text{if } m \neq 0, S_m^\beta = 0, \end{cases} \quad (1)$$

where $\mathbf{w}_{m+1}^\dagger$ is a parameter vector derived by a Zero Forcing (ZF) filter and ξ is the decoding efficiency, which can be used to model both the coded and uncoded systems.

Proof. Please refer to Appendix A in the full version [15]. □

It can be observed that if

the local detection is used (when $S_m^\beta = 0$), the data rate at a picocell suffers from the interference brought by the macro-user (the term $h_{0,S_0^\alpha}^m P_0$ in the second case of Eqn. (1)). Although joint detection can cancel the interference (as shown in Eqn. (1)) when $S_m^\beta = 1$), the limited backhaul capacity will introduce additional queuing delay, which is formally established in the following section.

[3]We assume that the data receiving gateway in the core network is able to reorder the packets received from the macro-BS and pico-BSs for each user.

C. Queuing Model

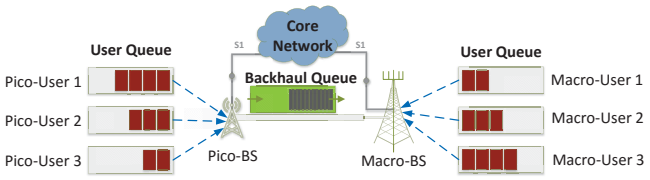


Fig. 2. The queuing model in HetNets. Each uplink user has its own data queue for uplink transmission, while all users in the same cell share a common backhaul queue.

Fig. 2 shows the queuing model in a HetNet cell. Let $Q_{m,k}^\alpha(t)$ denote the number of packets in the uplink buffer, and N_p denote the packet size in the number of bits. The pico-BS can send the raw received signals to the macro-BS via the backhaul links for joint detection. Because the signals have not been detected at the pico-BSs, the amount of the raw signals for joint detection can be very large (in terms of number of bits) so that the limited capacity of the backhaul links will introduce additional delay in the uplink transmission. Thus, we use a queue $Q_m^\beta(t)$ to denote the number of bits in the backhaul queue of the m^{th} pico-BS. Let $\mathbf{Q}_m(t) \triangleq (Q_m^\beta, Q_{m,1}^\alpha, \ldots, Q_{m,N_m}^\alpha)$ denote the joint queue state of the m^{th} pico-BS (which includes the backhaul queue and the users' data queues). Similarly, let $\mathbf{Q}_0(t) \triangleq (Q_{0,1}^\alpha, \ldots, Q_{0,N_0}^\alpha)$ denote the joint queue state of the macro-BS, which contains the data queue states of all the macro-users. Therefore, we can denote as $\mathbf{Q} \triangleq (\mathbf{Q}_0, \ldots \mathbf{Q}_M)$ the global joint queue state of the entire network. The cardinality of the global joint queue state $\mathbf{Q}$ is $I_Q = (N_Q^\alpha + 1)^{\sum_{m=0}^M N_m} \times (N_Q^\beta + 1)^M$, where N_Q^α and N_Q^β are the maximum buffer sizes of one uplink user and one backhaul queue, respectively.

Denoting $A_{m,k}(t)$ as the new arrival packets at the end of a time-slot for the user $k \in \mathcal{K}_m$, we make the following assumption on the data arrivals.

Assumption 1 (Packet Arrival Process)**.** *We assume that the arrival process $A_{m,k}(t)$ is i.i.d. over the time-slots and is according to a general distribution with an average arrival rate of $\lambda_{m,k}$, i.e., $\mathbb{E}[A_{m,k}(t)] = \lambda_{m,k}$. This assumption is widely adopted by works on cellular networks, such as [16].*

Hence for the k^{th} user in the m^{th} BS, the queue dynamics can be written as

$$Q_{m,k}^\alpha(t+1) = \min\{[Q_{m,k}^\alpha(t) - \lfloor \tau R_{m,k}(t)/N_p \rfloor]^+ + A_{m,k}(t), N_Q^\alpha\}, \quad (2)$$

where $[x]^+ = \max\{x, 0\}$ and $R_{m,k}(t)$ is the data rate of the k^{th} user in the m^{th} BS. Similarly, the dynamics of a backhaul queue for the m^{th} pico-BS is given by

$$Q_m^\beta(t+1) = \min\{[Q_m^\beta(t) - \tau R_m^\beta]^+ + S_m^\beta(t) \cdot f_\beta(\tau B), N_Q^\beta\},$$

where R_m^β is the data rate of the backhaul queue (i.e., the capacity of the backhaul link) in the m^{th} pico-BS. Note that the signal in the backhaul queue is not detected, and

thus its size $f_\beta(\tau B)$ only depends on the time duration, the sampling rate, the bandwidth of the frequency band and the signal compression algorithm used. The information rate $R_{m,k}(t)$ would not affect the size of the signals going into the backhaul queue. For example, if no signal compression is used, $f_\beta(\tau B) = 2\tau B N_{bit}$ according to the Nyquist sampling theorem [17], which is independent of the data rate or packet size. Here, N_{bit} is the number of bits used to represent one sampled signal value.

D. Problem Definition

At time-slot t, the decision of the m^{th} pico-BS is denoted as $\mathbf{S}_m(t) \triangleq \{S^\alpha_m(t), S^\beta_m(t), P_m(t)\}$, which contains the selected user, the detection mode and the transmission power. Similarly, the decision of the macro-BS is denoted as $\mathbf{S}_0(t) \triangleq \{S^\alpha_0(t), P_0(t)\}$. Here, we define the discounted accumulated cost of the user $k \in \mathcal{K}_m$ as follows

$$\overline{T}_{m,k} \triangleq \lim_{T\to\infty} \sup \mathbb{E}\Big[\sum_{t=0}^{T-1} \gamma^t f(Q^\alpha_{m,k}(t), Q^\beta_m(t), \mathbf{S}_m(t))\Big], \quad (3)$$

where $f(Q^\alpha_{m,k}(t), Q^\beta_m(t), \mathbf{S}_m(t))$ is the instantaneous cost function of this user and γ $(0 \le \gamma < 1)$ is a constant called the discount factor to indicate the importance of future cost. The instantaneous cost function $f(Q^\alpha_{m,k}(t), Q^\beta_m(t), \mathbf{S}_m(t))$ can be defined in various ways. According to the Little's law [18], $\overline{T}_{m,k}$ is related to the average end-to-end delay[4] of the k^{th} user in the m^{th} cell, and it can be estimated approximately with

$$f(Q^\alpha_{m,k}(t), Q^\beta_m(t), \mathbf{S}_m(t)) = \frac{Q^\alpha_{m,k}(t)}{\lambda_{m,k}} + \mathbf{I}[S^\alpha_m(t) = k] \cdot \mathbf{I}[S^\beta_m(t) = 1] \cdot \frac{Q^\beta_m(t)}{R^\beta_m}, \quad (4)$$

where $\mathbf{I}[x]$ is the indicator function which equals 1 if the condition x holds, or 0 otherwise.

At the beginning of each time slot, the HetNet controller at the macro-BS should collect all the system state information and then make a decision on the link scheduling. We define the system control policy for the HetNet controller as follows.

Definition 1 (Stationary Scheduling Policy)**.** *A stationary scheduling control policy $\Omega : \mathbf{H}, \mathbf{Q} \to \mathbf{S}$ is a mapping from the global CSI $\mathbf{H}$ and the global joint queue state $\mathbf{Q}$ to a control action $\mathbf{S} = \{\mathbf{S}_m : \forall m \in \mathcal{M} \cup \{0\}\}$.*

In this paper, we seek to find the optimal control policy Ω^* to minimize the total cost of all the uplink users. Specifically, we define the problem as follows:

[4]The delay from the packet's arrival at the user's buffer to the time instance when it arrives at the core network.

Problem 1. *(Delay-Optimal Uplink Scheduling Problem) For some positive constants $\boldsymbol{\beta} = \{\beta_{m,k} : m \in \mathcal{M} \cup \{0\}, k \in \mathcal{K}_m\}$, find a stationary control policy such that*

$$\min_{\Omega} \quad \sum_{m,k} \eta_{m,k} \overline{T}_{m,k}(\Omega) = \lim_{T\to\infty} \sup \mathbb{E}\Big[\sum_{t=0}^{T-1} \gamma^t g(\mathbf{Q}(t), \mathbf{S}(t))|\Omega\Big], \quad (5)$$

where $g(\mathbf{Q}(t), \mathbf{S}(t)) = \sum_{m,k} \eta_{m,k} f(Q^\alpha_{m,k}(t), Q^\beta_m(t), \mathbf{S}_m(t))$ is the weighted per time-slot (instantaneous) cost. The constant $\eta_{m,k}$ denotes the relative importance of the uplink user $k \in \mathcal{K}_m$. For a given set of β, the optimal solution of this problem is a Pareto optimal to the multi-objective optimization problem to minimize the cost of each uplink user, i.e., $\min_\Omega \overline{T_{m,k}}(\Omega), \forall m, k$. Thus Ω^* is a Pareto optimal control.

***Remark* 1** (The discounted cost)**.** *In this work, we adopt the discounted accumulated cost function in Eqn. (3). The other widely adopted criterion is the long-term average cost without the discount factor [19], such as*

$$\lim_{T\to\infty} \sup \frac{1}{T} \mathbb{E}\Big[\sum_{t=0}^{T-1} g(\mathbf{Q}(t), \mathbf{S}(t))|\Omega\Big]. \quad (6)$$

However, this average cost criterion is insensitive to the instantaneous cost due to its arbitrary far tail cost. For example, two control policies providing costs $10 + 0 + 0 + 0 + 0 + \cdots$ and $0 + 0 + 0 + 0 + 0 + \cdots$ are equally good for the average cost criterion, but we know the latter is better with the discounted accumulated cost criterion. Moreover, it has been shown that the optimal control policy for the discounted accumulated cost is the same as that for the average cost criterion when the discount factor is sufficiently close to 1 [20]–[24]. Therefore, with a properly chosen discount factor γ, the discounted accumulated cost criterion can generate the control policy with good average cost and remain sensitive to the instantaneous cost.

Theorem 1. *The delay-optimal uplink scheduling problem in HetNets is NP-hard.*

Proof. Please refer to Appendix B in the full version [15]. □

III. OLIUS: Onine Learning based Uplink Scheduling Algorihtm in HetNets

In this section, we propose our algorithm, called `OLIUS` (the Online LearnIng based Uplink Scheduling algorithm). Here is a sketch of `OLIUS`:

- **Section III-A**: We first derive the optimal condition (i.e., the Bellman's equation) for solving Problem 1. However, solving the Bellman's equation with general approaches suffers from the curse of dimensionality.
- **Section III-B1**: In order to avoid the curse of dimensionality in the system state space, we define the approximate $\mathbb{Q}$-function only on some specific states, so that the space grows polynomially with the number of users and picocells.

- **Section III-B2**: As the control action space also grows exponentially, we adopt Factored MDP to exploit the system structure, which can obtain the optimal control action in polynomial time.
- **Section III-C**: Because we assume no prior knowledge on the statistical information of the system (such as the packet arrival rate and channel characteristics), we propose an online learning based algorithm to learn the approximate $\mathbb{Q}$-function to estimate the necessary information.

A. The Bellman's Equation

The Bellman's equation [25] is the sufficient condition for the optimal control policy Ω^* to solve the MDP problem in Problem 1, which is given by

$$\mathbb{Q}^*(\mathbf{Q},\mathbf{S}) = \mathbb{E}_{\mathbf{H}}\Big\{ g(\mathbf{Q},\mathbf{S}) + \gamma \sum_{\mathbf{Q}'} Pr\{\mathbf{Q}'|\mathbf{Q},\mathbf{S}\} \min_{\mathbf{S}'} \mathbb{Q}^*(\mathbf{Q}',\mathbf{S}') \Big| \mathbf{S} \Big\}, \quad (7)$$

where $\mathbb{Q}^*(\mathbf{Q},\mathbf{S})$ is called the "$\mathbb{Q}$-function" which equals the expected discounted accumulated cost of the state $\mathbf{Q}$ by taking the control action $\mathbf{S}$ at the first time-slot and following the optimal control policy in future time-slots. Thus the optimal control policy $\Omega^*(\mathbf{Q})$ can be obtained by $\Omega^*(\mathbf{Q}) = \text{argmin}_{\mathbf{S}}\{\mathbb{Q}^*(\mathbf{Q},\mathbf{S})\}$.

Note that the optimal $\mathbb{Q}$-function (i.e., $\mathbb{Q}^*$) is a function of the global user states and actions, whose space grows exponentially with the number of uplink users and picocells. General approaches, such as value iteration [26] or policy iteration [27], to solve the Bellman's equation in Eqn. (7) need exponential time. Thus, we aim to find an efficient algorithm to get an approximate solution.

B. Approximate $\mathbb{Q}$-function & Factored MDP

Finding the optimal $\mathbb{Q}$-function in Eqn. (7) is actually computationally prohibitive due to the following two reasons. (1) Since the number of possible system states $\mathbf{Q}$ is huge, the number of $\mathbb{Q}$-function values to be calculated in Eqn. (7) is also huge. (2) Due to the minimization at $\Omega^*(\mathbf{Q}) = \text{argmin}_{\mathbf{S}}\{\mathbb{Q}^*(\mathbf{Q},\mathbf{S})\}$, we have to check all the possible control action $\mathbf{S}$ for each system state $\mathbf{Q}$. However, the number of control actions grows exponentially with respect to the number of cells. In this paper, we propose a low-complexity approximate solution that uses the techniques of approximate $\mathbb{Q}$-function and Factored MDP. The former is good at dealing with the large state space and the latter can efficiently exploit the action structure to derive the control policy optimally based on the approximate $\mathbb{Q}$-function.

1) Approximate $\mathbb{Q}$-function : To deal with the huge state space in the calculation of Bellman's equation (7), we propose to approximate the $\mathbb{Q}$-function by the summation of "per-user $\mathbb{Q}$-function" on some specific "reference states". Specifically, we define $\mathbf{Q}_I$ as the reference states, which is given as

$$\mathbf{Q}_I \triangleq \{\mathbf{T}_{m,k} \circ \mathbf{Q} \mid \forall\ \mathbf{Q}, m, k\}, \quad (8)$$

where the operator $\mathbf{T}_{m,k} \circ \mathbf{Q}$ is defined as

$$\mathbf{T}_{m,k} \circ \mathbf{Q} \triangleq (\mathbf{Q}_0 = \mathbf{0}, \ldots, \mathbf{Q}_{m-1} = \mathbf{0}, \mathbf{Q}_m = (Q_m^{\beta}, \underbrace{0,\ldots,0}_{k-1}, Q_{m,k}^{\alpha}, \underbrace{0,\ldots 0}_{N_m-k}), \mathbf{Q}_{m+1} = \mathbf{0}, \ldots, \mathbf{Q}_M = \mathbf{0}). \quad (9)$$

Note that $\mathbf{T}_{m,k} \circ \mathbf{Q}$ picks up the queue information of the k^{th} user in the m^{th} cell only. Then we can apply the following approximation on the $\mathbb{Q}$-function:

$$\mathbb{Q}(\mathbf{Q},\mathbf{S}) \approx \sum_{m,k} \mathbb{Q}_{m,k}(\varphi_{m,k}), \quad (10)$$

where $\varphi_{m,k} \triangleq (\mathbf{T}_{m,k} \circ \mathbf{Q}, \mathbf{S}_m, \mathbf{S}_0)$. The notation $\mathbb{Q}_{m,k}$ is called the per-user $\mathbb{Q}$-function, and it satisfies the Bellman's equation in the reference states as follows.

$$\mathbb{Q}_{m,k}(\varphi_{m,k}) = \mathbb{E}_{\mathbf{H}}\Big[\beta_{m,k} \cdot f_{m,k}(\mathbf{T}_{m,k} \circ \mathbf{Q}, \mathbf{S}_m) + \gamma \sum_{\mathbf{Q}'} Pr\{\mathbf{Q}'|\varphi_{m,k},\mathbf{H}\} \min_{\mathbf{S}'=(\mathbf{S}_0',\mathbf{S}_1',\ldots,\mathbf{S}_{m'})} \sum_{m,k} \mathbb{Q}_{m,k}(\varphi'_{m,k}) \Big| \varphi_{m,k} \Big], \quad \forall m,k. \quad (11)$$

where $\varphi'_{m,k} = (\mathbf{Q}', \mathbf{S}'_m, \mathbf{S}'_0)$ is the next local state-action.

Because the per-user $\mathbb{Q}$-function $\mathbb{Q}_{m,k}(\varphi_{m,k})$ is only defined when $\mathbf{Q} \in \mathbf{Q}_I$ with the action of the m^{th} picocell and the macrocell, we can avoid calculating the $\mathbb{Q}$-function for all possible state-action pairs. In Section III-C, we propose the $\mathbb{Q}$-Learning algorithm to obtain the approximate per-user $\mathbb{Q}$-function. Our theoretical analysis in Theorem 3 shows the error bound on the performance with the learned per-user $\mathbb{Q}$-function. Although Eqn. (11) solves the large state space problem, its computation complexity is still prohibitively high due to the minimization operation in $\min_{\mathbf{S}}\{\sum_{m,k} \mathbb{Q}_{m,k}(\varphi'_{m,k})\}$ (the space of the control action $\mathbf{S}$ grows exponentially with respective to the number of cells). In the following, we introduce the Factored MDP (FMDP) that can optimally deal with the large control action space in polynomial time.

2) Factored MDP: The FMDP model was first proposed by Boutilier et. al. [28] who used a two-time slices Dynamic Bayesian Network (DBN) to represent the relations between the states and actions. In our problem, the state transition of the users in the macrocell only depends on the control actions of the macro-BS, i.e., $\mathbf{S}_0(t)$. However, the state transition of the users in a picocell depends not only on the control actions of the pico-BS (i.e., $\mathbf{S}_m(t)$) but also the control actions of the macro-BS (i.e., $\mathbf{S}_0(t)$). With the standard approach of FMDP, we can decompose the state transition probability as follows:

$$\begin{aligned} &\prod_{k=0}^{M} Pr[\mathbf{Q}_k(t+1)|\mathbf{Q}_k(t),\mathbf{S}(t),\mathbf{H}(t)] \\ &= \prod_{k=1}^{N_0} Pr[Q_{0,k}^{\alpha}(t+1)|Q_{0,k}^{\alpha}(t),\mathbf{S}_0(t),\mathbf{H}(t)] \times \\ &\quad \prod_{m,k} Pr[Q_{m,k}^{\alpha}(t+1)|Q_{m,k}^{\alpha}(t),\mathbf{S}_m(t),\mathbf{S}_0(t),\mathbf{H}(t)] \\ &\quad \times Pr[Q_m^{\beta}(t+1)|Q_m^{\beta}(t),\mathbf{S}_m(t)]. \end{aligned} \quad (12)$$

According to Eqn. (12), the users in the m^{th} picocell are only affected by the control actions of the macrocell (i.e., $\mathbf{S}_0$) and their service picocell (i.e., $\mathbf{S}_m$). Therefore, the minimization operation in Eqn. (11) can be written as

$$\min_{\mathbf{S}=(\mathbf{S}_0,\mathbf{S}_1,\ldots\mathbf{S}_m)}\{\sum_{m,k}\mathbb{Q}_{m,k}(\varphi_{m,k})\} \tag{13}$$
$$=\min_{\mathbf{S}_0}\Big\{\underbrace{\sum_{k\in\mathcal{K}_0}\mathbb{Q}_{0,k}(\varphi_{0,k})}_{\text{The Macrocell}}+\sum_{m\in\mathcal{M}}\underbrace{\Big(\min_{\mathbf{S}_m}\sum_{k\in\mathcal{K}_m}\{\mathbb{Q}_{m,k}(\varphi_{m,k})\}\Big)}_{\text{The } m^{th} \text{ picocell}}\Big\}.$$

With the above decomposition, the computation complexity is reduced from $O(\prod_{m=0}^{M}(|\mathbf{S}_m|\cdot\sum_{m\mathcal{M}}|\mathcal{K}_m|))$ to $O(|\mathbf{S}_0|\times\sum_{m\mathcal{M}}|\mathbf{S}_m||\mathcal{K}_m|)$, which is a polynomial time complexity.

C. The Proposed Algorithm: OLIUS

In the above section, we adopt the approximate $\mathbb{Q}$-function and the factored MDP to simplify the original Problem 1. So far, the only unsolved problem is how to obtain the per-user $\mathbb{Q}$-function. In this section, we first propose the solution framework to determine the control actions, and then elaborate on the reinforcement learning based algorithm that can adaptively learn the value of the per-user $\mathbb{Q}$-function.

Algorithm 1: OLIUS

1 $t=0$;
2 $\mathbb{Q}_{m,k}(\varphi_{m,k})=0,\quad\forall m,k,\varphi_{m,k}$;
3 **for** $t=0,1,2,\ldots$ **do**
4 Determine the control action according to Eqn. (13) ;
5 **if** $\mathbb{Q}$-*function converged* = *False* **then**
6 Update $\mathbb{Q}$-function according to Eqn. (14) ;
7 **else**
8 Apply LSPI to adjust the $\mathbb{Q}$-function ;

1) Solution Framework: Algorithm 1 elaborates the solution framework. In each time-slot, the control action is determined by Eqn. (13) (Line 4). Afterwards, the approximate $\mathbb{Q}$-function is updated according to the Discounted $\mathbb{Q}$-Learning Algorithm in Section III-C2 (Line 6). After the convergence of the approximate $\mathbb{Q}$-function, the Least Square Policy Iteration (LSPI) in Algorithm 2 is applied to adjust the approximate $\mathbb{Q}$-function for policy improvement (Line 8).

Although the approximate $\mathbb{Q}$-function can be calculated by some offline methods, such as value iteration or policy iteration [29], these methods require prior statistical knowledge about the channel state and user behavior which are typically unknown in practice. Therefore, in the rest of this section, we adopt a reinforcement learning based algorithm called Discounted Cost $\mathbb{Q}$-Learning algorithm which can adaptively calculate the value of the approximate $\mathbb{Q}$-function by learning the statistical information online.

2) Discounted Cost $\mathbb{Q}$-Learning Algorithm with LSPI : As stated in Remark 1, instead of considering the average cost over an infinite time-horizon, we consider the discounted accumulated cost over an infinite time-horizon with a large discount factor close to 1 (e.g. $\gamma=0.99$).

At time-slot t, when the current system state is a reference state (i.e., $\mathbf{Q}(t)\in\mathbf{Q}_I$), the updating rule for the approximate $\mathbb{Q}$-function under the discounted cost model is:

$$\begin{aligned}\mathbb{Q}^{t+1}_{m,k}(\varphi_{m,k}(t))=(1-\epsilon_{l_{m,k}(\varphi_{m,k}(t))})\mathbb{Q}^{t}_{m,k}(\varphi_{m,k}(t))+\\ \epsilon_{l_{m,k}(\varphi_{m,k}(t))}\Big[\eta_{m,k}f_{m,k}(\varphi_{m,k})\\ +\gamma\mathbb{Q}^{t}_{m,k}(\varphi_{m,k}(t+1))\Big].\end{aligned} \tag{14}$$

Eqn. (14) is to adaptively adjust the approximate per-user $\mathbb{Q}$-function according to the instantaneous cost (i.e. $\eta_{m,k}f_{m,k}(\varphi_{m,k})$) and the current observed discounted accumulated cost (i.e., $\gamma\mathbb{Q}^{t}_{m,k}(\varphi_{m,k}(t+1))$. In Theorem 2, we prove the approximate per-user $\mathbb{Q}$-function will converge with the update in Eqn. (14) in finite steps. After the convergence, an algorithm called Least Square Policy Iteration (LSPI) [30], [31] is applied to the approximate $\mathbb{Q}$-function for policy improvement, which is elaborated in Algorithm 2.

Algorithm 2: Least-Squares Policy Iteration (LSPI)

Input:
N_{user}: The number of users;
$\mathbb{Q}_{m,k}$: The per-user $\mathbb{Q}$-function for all users;
D: A list of samples $(\mathbf{Q},\mathbf{S},g(\mathbf{Q},\mathbf{S}),\mathbf{Q}')$
δ: The convergence condition of weights;
Output: $\widehat{\mathbb{Q}}_{m,k}$: The improved $\mathbb{Q}$-function
1 $\omega^0_{m,k}\leftarrow 1,\forall m,k$; $\omega^0\leftarrow\{\omega^0_{m,k}\}_{m,k}$; $i\leftarrow 0$;
2 $\phi(\mathbf{Q},\mathbf{S})\triangleq\{\mathbb{Q}_{m,k}(\varphi_{m,k})\}_{m,k}$;
3 **repeat**
4 $\widetilde{\mathbf{A}}\leftarrow\mathbf{0}$; //$(N_{user}\times N_{user})$ matrix;
5 $\widetilde{b}\leftarrow\mathbf{0}$; //$(N_{user}\times 1)$ vector;
6 **foreach** *sample* $(\mathbf{Q},\mathbf{S},g(\mathbf{Q},\mathbf{S}),\mathbf{Q}')$ *in* D **do**
7 Calculate $\mathbf{S}'$ by Eqn. (16) ;
8 $\widetilde{\mathbf{A}}\leftarrow\widetilde{\mathbf{A}}+\phi(\mathbf{Q},\mathbf{S})(\phi(\mathbf{Q},\mathbf{S})-\gamma\phi(\mathbf{Q}',\mathbf{S}'))$;
9 $\widetilde{b}\leftarrow\widetilde{b}+\phi(\mathbf{Q},\mathbf{S})g(\mathbf{Q},\mathbf{S})$;
10 $\omega^{i+1}\leftarrow\widetilde{\mathbf{A}}^{-1}\widetilde{b}$;
11 $i\leftarrow i+1$;
12 **until** $\|\omega^{i+1}-\omega^{i}\|<\delta$;;
13 **return** $\widehat{\mathbb{Q}}_{m,k}\leftarrow\omega^{i}_{m,k}\cdot\mathbb{Q}_{m,k},\quad\forall m,k$;

In order to explain how LSPI works, we should define the Bellman error on the $\mathbb{Q}$-function as follows:

$$BellmanErr(\mathbb{Q})\triangleq\|\mathcal{T}\mathbf{V}-\mathbf{V}\|_\infty, \tag{15}$$

where $\mathbf{V}(\mathbf{Q})\triangleq\min_{\mathbf{S}}\mathbb{Q}(\mathbf{Q},\mathbf{S})$ and $\mathcal{T}$ is the operator which is equal to the R.H.S. of Eqn. (11) with the state $\mathbf{Q}$ and the action $\text{argmin}_{\mathbf{S}}\,\mathbb{Q}(\mathbf{Q},\mathbf{S})$. The Bellman error should be 0 if the control policy is optimal, i.e., $BellmanErr(\mathbb{Q})=0$ if $\Omega=\Omega^*$. However, the approximate per-user $\mathbb{Q}$-function

learned by Eqn. (14) is not optimal, it is hard to obtain the Bellman error of 0 with the approximate $\mathbb{Q}$-function. Nonetheless, we can adjust the approximate $\mathbb{Q}$-function to reduce the Bellman error. Therefore, after the convergence of the approximate $\mathbb{Q}$-function with the learning update in Eqn. (14), the Least-Squares Policy Iteration (LSPI) in Algorithm 2 would be executed to adjust the approximate $\mathbb{Q}$-function by $\widehat{\mathbb{Q}}_{m,k} = \omega^i_{m,k}\mathbb{Q}_{m,k}$, where $\widehat{\mathbb{Q}}_{m,k}$ is the adjusted approximate $\mathbb{Q}$-function and $\omega^i_{m,k}$ is the adjust factor calculated by LSPI.

Before the execution of Algorithm 2, a set of samples is collected by observing the system state $\mathbf{Q}$, the control action $\mathbf{S}$, the instantaneous cost $g(\mathbf{Q},\mathbf{S})$ and the new system state $\mathbf{Q}'$ in the next time-slot. In Algorithm 2, Line 1 to Line 2 are to initialize the weights $\omega^0_{m,k} = 1$ for all users. Line 4 to Line 11 are to iteratively calculate the weights to reduce the Bellman error in Eqn. (15). Note that, in Line 7, the action $\mathbf{S}'$ for the state $\mathbf{Q}'$ is calculated by minimizing the following equation, which is similar to Eqn. (13) but the approximate per-user $\mathbb{Q}$-function is replaced by $\omega^i_{m,k}\mathbb{Q}_{m,k}$, i.e.

$$\min_{\mathbf{S}} \mathbb{Q}(\mathbf{Q},\mathbf{S}) = \min_{\mathbf{S}_0}\Big(\sum_{k\in\mathcal{K}_0} \omega^i_{0,k}\mathbb{Q}_{0,k}(\varphi_{0,k}) + \sum_{m\in\mathcal{M}} \min_{\mathbf{S}_m} \sum_{k\in\mathcal{K}_m} \omega^i_{m,k}\mathbb{Q}_{m,k}(\varphi_{m,k})\Big). \quad (16)$$

After the convergence of the weights, the adjusted approximate $\mathbb{Q}$-function $\widehat{\mathbb{Q}}_{m,k}$ is returned to replace the original approximate $\mathbb{Q}$-function $\mathbb{Q}_{m,k}$ in future time-slots (Line 13).

***Remark* 2** (*The Time Complexity of* `OLIUS`)**.** We can divide the algorithm `OLIUS` into two phases: (1) the online phase (i.e., the online learning with Eqn. (14)) and (2) the offline phase (i.e., adjusting the $\mathbb{Q}$-function with LSPI in Algorithm 2. In the online phase, the control action is first decided by Eqn. (13) with $O(|\mathbf{S}_0| \times \sum_{m\mathcal{M}}$

—$\mathrm{S}_m||\mathcal{K}_m|)$ time complexity. Then, the controller updates the $\mathbb{Q}$-function according to Eqn. (14) with $O(1)$ time complexity. Thus the online phase has a low complexity and can be further accelerated with parallel computing. After the convergence of the $\mathbb{Q}$-function, `OLIUS` needs to adjust the $\mathbb{Q}$-function using LSPI. Although LSPI is an iterative algorithm with high time complexity ($O(|D||\mathbf{S}_0| \times \sum_{m\mathcal{M}} |\mathbf{S}_m||\mathcal{K}_m|)$ per iteration), it can be executed offline in parallel with the online phase. After LSPI completes, we can replace the old $\mathbb{Q}$-function with the new one returned by LSPI. As the offline phase does not affect the `OLIUS`'s execution time, we can say `OLIUS` is a low complexity algorithm and so it is practical.

***Remark* 3** (*Comparison with Previous Approximate MDP-based Approaches in Cellular Networks*)**.** There have been a number of works based on approximate MDP (e.g., [10]–[12]) in cellular networks. Their approaches directly approximate the $\mathbb{Q}$-function with a sum of per-user $\mathbb{Q}$-function and no theoretical analysis on the approximation error has been provided. Moreover, most of these works only deal with a very small action space. However, in this paper, we adopt the FMDP and the approximate $\mathbb{Q}$-function to deal with a much larger state and action space. We also show the performance guarantee with theoretical analysis in Sec. III-D. Our method can be extended to a general class of problems, which are usually referred to as weakly coupled multi-agent resource allocation problems, such as [32], [33].

D. Theoretical Analysis: Convergence and Error Bound

We formally show the convergence of `OLIUS` in Theorem 2.

Theorem 2 (Convergence of `OLIUS`)**.** *With the updating rule in Eqn. (14), the $\mathbb{Q}$-function will converge almost surely for any given initial $\mathbb{Q}$-function, i.e.,*

$$\lim_{t\to\infty} \mathbb{Q}^t_{m,k}(\mathbf{Q},\mathbf{S}) = \mathbb{Q}^\infty_{m,k}(\mathbf{Q},\mathbf{S}). \quad (17)$$

And $\epsilon_{l_{m,k}}$ should satisfy [34] $\sum_n \epsilon_n = \infty$ and $\sum_n \epsilon_n^2 < \infty$. Examples of step sizes satisfying this condition are $\epsilon_n = \frac{1}{n}, \frac{1}{n\log n}, \frac{\log n}{n}$, etc., for $n \geq 2$.

Proof. Please refer to Appendix C in the full version [15]. □

The approximation error of `OLIUS` with LSPI can be bounded by the Bellman error defined in Eqn. (15), which is given by the following Theorem 3.

Theorem 3 (Error Bound of `OLIUS`)**.** *Let $\mathcal{V}^*$ and $\mathcal{V}^\pi$ denote the $\mathbb{Q}$-function associated with the optimal control policy Ω^* and the approximate control policy Ω^π obtained by Eqn. (13), respectively. The loss due to acting according to the approximate control policy Ω^π instead of the optimal control policy Ω^* is bounded by:*

$$\|\mathcal{V}^* - \mathcal{V}^\pi\|_\infty \leq \frac{2\gamma BellmanErr(\mathbb{Q}^\infty)}{(1-\gamma)^2}, \quad (18)$$

where $\mathbb{Q}^\infty$ is the converged approximate $\mathbb{Q}$-function adjusted by the LSPI in Algorithm 2.

Proof. Please refer to Appendix D in the full version [15]. □

IV. Performance Evaluation

We have conducted extensive simulations to evaluate the performance of the proposed algorithm, `OLIUS`. In Problem 1, we aim to minimize the discounted accumulated cost function over an infinite time horizon; however, all the baselines are designed to optimize the average performance. For the sake of fairness, all the performance metrics in the section are calculated in average without the discount factor, including our proposed method. Moreover, all users are equal weighted (i.e., $\forall m,k,\ \eta_{m,k} = 1$).

We compare `OLIUS` to three classical scheduling methods in cellular networks and one approximate MDP-based approach:

- Round Robin Scheduler: to choose the uplink users in a circular order.
- CSIT (Channel State Information at the Transmitter) Only Scheduler: to optimize the uplink throughput without considering queueing status.

- Dynamic Backpressure [35]: a throughput-optimal scheduling policy (in stability sense).
- Average Cost $\mathbb{Q}$-Learning Algorithm: an approximate MDP-based approach with long-term average cost (instead of discounted cost), which is elaborated in the following remark.

***Remark* 4** (Average Cost $\mathbb{Q}$-Learning Algorithm)**.** *When solving scheduling problems, most previous works using MDP models adopted the average cost $\mathbb{Q}$-Learning algorithm (e.g., [10]–[12], [19]). We extend their algorithms to solve the scheduling problem in HetNets as a baseline.*

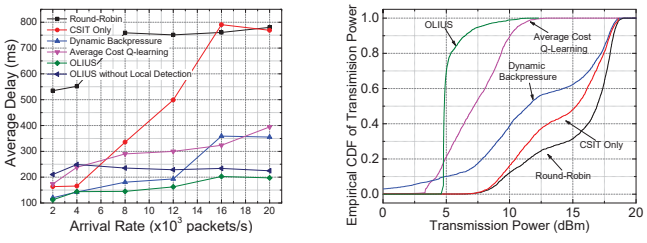
Denote $\varphi_{m,k}(t) \triangleq \left(\mathbf{T}_{m,k} \circ \mathbf{Q}(t), \mathbf{S}_m(t), S_0(t)\right)$ as the state and action for the user $k \in \mathcal{K}_m$ in the BS $m \in \mathcal{M} \cup \{0\}$. At time-slot t, when the system state $\mathbf{Q}(t)$ is in a reference state (i.e., $\mathbf{Q}(t) \in \mathbf{Q}_I$), the $\mathbb{Q}$-function for the user k in BS m updates its value according to:

$$\mathbb{Q}^{t+1}_{m,k}(\varphi_{m,k}(t)) = \mathbb{Q}^{t}_{m,k}(\varphi_{m,k}(t)) + \epsilon_{l_{m,k}(\varphi_{m,k}(t))}\Big[\eta_{m,k} f_{m,k}(\varphi_{m,k}) + \mathbb{Q}^{t}_{m,k}(\varphi_{m,k}(t+1)) - \mathbb{Q}^{t}_{m,k}(\varphi^r) - \mathbb{Q}^{t}_{m,k}(\varphi_{m,k}(t))\Big], \quad (19)$$

where φ^r is the prescribed reference state-action pair. $l_{m,k}(\varphi_{m,k})$ is to count how many times $\varphi_{m,k}$ appears, i.e., $l_{m,k}(\varphi_{m,k}) \triangleq \sum_{i=0}^{t} \mathbf{I}[\varphi_{m,k}(i) = \varphi_{m,k}]$. And $\epsilon_{l_{m,k}}$ is the step size sequence for updating.

Next, we evaluate the impacts of the packet arrival rate, the backhaul capacity, the number of macro users, as well as the performance gain brought by LSPI. The frequency bandwidth is 10 MHz with -50 dBm/Hz spectrum noise. The packet size is 1000 bytes and the packet arrival follows a Poisson distribution with 20 ms slot duration. A mobile user has a 2Mb data buffer. A backhaul link has a 50 Mb backhaul buffer. The macro-BS is equipped with 4 receiving antennas. The maximum transmit power of mobile users is 24 dBm. OLIUS uses Eqn. (4) as the cost function. The discount factor γ is 0.9.

A. Impact of Packet Arrival Rate



(a) Average Delay

(b) CDF of Transmission Power (arrival rate = 20×10^3 packets/s)

Fig. 3. Impact of packet arrival rate

By varying the packet arrival rate of users, we observe the average delay and power consumption of OLIUS and the baselines under different transmission loads. Fig. 3(a) and Fig. 3(b) illustrate the average delay

and empirical CDF of the transmission power, respectively, with 3 picocells, 5 users in each picocell and 5 macro-users

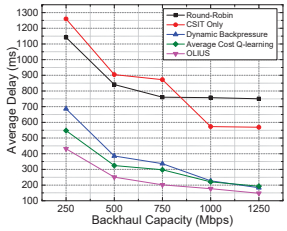


Fig. 4. Average delay versus different backhaul capacity.

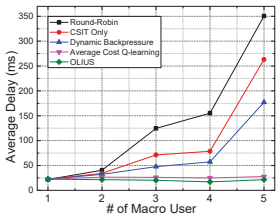


Fig. 5. Average delay versus the number of macro users.

in the macrocell. It can be observed that OLIUS has the least average delay and transmission power. One interesting phenomenon is the performance gain brought by local detection. In Fig. 3(a), the line marked "OLIUS without Local Detection" is the delay performance of OLIUS using only joint detection. It shows that using local detection intelligently could bring significant improvement to the performance, especially when the arrival rate is low. The performance gain brought by local detection degrades with higher arrival rate. It is because higher arrival rate increases the demand for higher data rate, and thus joint detection at the macro-BS is more preferred for interference cancellation.

B. Impact of Backhaul Capacity

Fig. 4 demonstrates the impact of the backhaul capacity on the average delay with 3 picocells, 5 users in each picocell and only 1 macro-user in the macro-cell. With larger backhaul capacity, all methods perform better due to the reduced delay in the backhaul links. The proposed algorithm outperforms all three baselines in this experiment. Note that the performance gain of OLIUS increases with less backhaul capacity. It indicates that OLIUS is more efficient when the backhaul capacity is limited because our method can make the joint/local detection decision more intelligently via the joint uplink scheduling among all cells.

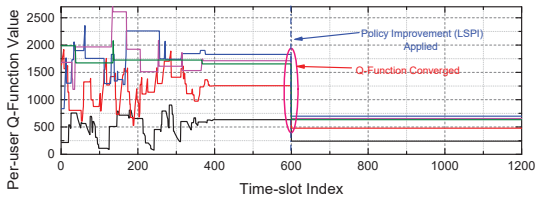
C. Impact of the Number of Macro Users

Both OLIUS and Average Cost Q-Learning consider the impact brought by the macrocell to picocells, which is not taken into account by the other three baselines (Round-Robin, CSIT Only and Dynamic Backpressure). Fig. 5 illustrates the average delay versus the number of macro users. In this experiment, there are 5 macro-users in the macro-cell but only 1 pico-user in each of the 3 pico-cells. It can be observed that the average delay of OLIUS and Average Cost Q-Learning remains almost unchanged with more macro-users, while it degrades in the other three baselines. It indicates that the cross-tier mitigation can improve the performance by considering the impact from macro-users.

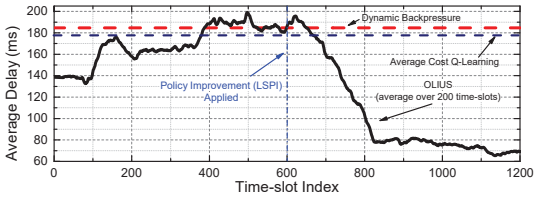
D. Impact of LSPI

Fig. 6(a) and 6(b) illustrate the impact of LSPI on the per-user $\mathbb{Q}$-function and the average delay respectively. After applying LSPI, the average delay significantly drops by over 50%. It shows that the policy improvement made by LSPI

can bring significant performance gain based on the per-user $\mathbb{Q}$-function learned with Discounted Cost $\mathbb{Q}$-Learning.



(a) Per-user $\mathbb{Q}$-function



(b) Average Delay

Fig. 6. The variation of selected per-user $\mathbb{Q}$-function and the average delay with 3 picocells, 3 users in each picocell, 5 macro-users in the macrocell.

V. Conclusion

In this paper, we study the delay-optimal uplink scheduling problem in HetNets with limited backhaul capacity. In addition to considering user scheduling and power allocation, we further allow local detection at the pico-BSs. We first formulate the problem as an MDP problem. Then, we propose our algorithm OLIUS that adopts approximate $\mathbb{Q}$-Function and Factored MDP to exploit the model structure so that the problem can be solved approximately in polynomial time.

OLIUS works online without any prior statistic information of the user behavior or the channel characteristics. We theoretically prove the convergence of OLIUS and its error bound with the approximate $\mathbb{Q}$-Function. Extensive simulations show that our algorithm dramatically outperforms existing scheduling algorithms both in delay and power consumption. In this work, we only consider the scheduling in one HetNet cell. It would be interesting as future work to jointly schedule the communication and the computation in networking systems using reinforcement learning, such as in HetNets as well as data centers and cloud computing [36].

References

[1] H. Li *et al.*, "Efficient hetnet implementation using broadband wireless access with fiber-connected massively distributed antennas architecture," *IEEE Wireless Communications*, vol. 18, no. 3, pp. 72–78, 2011.

[2] D. Fooladivanda and C. Rosenberg, "Joint resource allocation and user association for heterogeneous wireless cellular networks," *IEEE Transactions on Wireless Communications,*, vol. 12, no. 1, 2013.

[3] R. Madan *et al.*, "Cell association and interference coordination in heterogeneous lte-a cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 9, 2010.

[4] M. Hong *et al.*, "Joint base station clustering and beamformer design for partial coordinated transmission in heterogeneous networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 2, 2013.

[5] H. S. Dhillon *et al.*, "Modeling and analysis of k-tier downlink heterogeneous cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, 2012.

[6] J. Ghimire and C. Rosenberg, "Revisiting scheduling in heterogeneous networks when the backhaul is limited," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 10, 2015.

[7] N. Sapountzis *et al.*, "Optimal downlink and uplink user association in backhaul-limited hetnets," *INFOCOM*, 2016.

[8] H. M. T. Ho *et al.*, "Joint load balancing and interference management for small-cell heterogeneous networks with limited backhaul capacity," *IEEE Transactions on Wireless Communications*, 2016.

[9] S.-B. Lee et al., "Proportional fair frequency-domain packet scheduling for 3gpp lte uplink," in *IEEE INFOCOM 2009*.

[10] Y. Cui and V. K. Lau, "Distributive stochastic learning for delay-optimal ofdma power and subband allocation," *IEEE Transactions on Signal Processing*, vol. 58, no. 9, 2010.

[11] H. Huang and V. K. Lau, "Delay-optimal user scheduling and inter-cell interference management in cellular network via distributive stochastic learning," *IEEE Transactions on Wireless Communications*, 2010.

[12] R. Wang and V. K. Lau, "Delay-aware two-hop cooperative relay communications via approximate mdp and stochastic learning," *IEEE Transactions on Information Theory*, vol. 59, no. 11, 2013.

[13] J. G. Andrews, "Seven ways that hetnets are a cellular paradigm shift," *IEEE Communications Magazine*, vol. 51, no. 3, pp. 136–144, 2013.

[14] K. I. Pedersen *et al.*, "eICIC functionality and performance for lte hetnet co-channel deployments," in *IEEE VTC Fall 2012*.

[15] "Full version with appendices." https://goo.gl/HDDh2w.

[16] M. J. Neely, E. Modiano, and C.-P. Li, "Fairness and optimal stochastic control for heterogeneous networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, 2008.

[17] C. E. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[18] J. D. Little, "A proof for the queuing formula: L= λ w," *Operations research*, vol. 9, no. 3, 1961.

[19] Z. Han, H. Tan, G. Chen, R. Wang, Y. Chen, and F. C. M. Lau, "Dynamic virtual machine management via approximate markov decision process.," in *IEEE INFOCOM 2016*.

[20] D. Blackwell, "Discrete dynamic programming," *The Annals of Mathematical Statistics*, 1962.

[21] M. Schäl, "Average optimality in dynamic programming with general state space," *Mathematics of Operations Research*, vol. 18, no. 1, pp. 163–172, 1993.

[22] E. A. Feinberg, P. O. Kasyanov, and N. V. Zadoianchuk, "Average cost markov decision processes with weakly continuous transition probabilities," *Mathematics of Operations Research*, vol. 37, no. 4, 2012.

[23] L. I. Sennott, *Stochastic dynamic programming and the control of queueing systems*, vol. 504. John Wiley & Sons, 2009.

[24] O. Hernández-Lerma and J. B. Lasserre, *Discrete-time Markov control processes: basic optimality criteria*, vol. 30. Springer Science & Business Media, 2012.

[25] R. Bellman, "Dynamic programming and lagrange multipliers," *Proceedings of the National Academy of Sciences*, vol. 42, no. 10, 1956.

[26] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*, vol. 23. Prentice hall Englewood Cliffs, NJ, 1989.

[27] D. Koller and R. Parr, "Policy iteration for factored mdps," in *UAI 2000*.

[28] C. Boutilier, R. Dearden, M. Goldszmidt, *et al.*, "Exploiting structure in policy construction," in *IJCAI 1995*.

[29] D. P. Bertsekas, *Dynamic programming and optimal control*. 1995.

[30] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *The Journal of Machine Learning Research*, vol. 4, pp. 1107–1149, 2003.

[31] R. Munos, "Error bounds for approximate policy iteration," in *ICML 2003*.

[32] D. A. Dolgov and E. H. Durfee, "Resource allocation among agents with mdp-induced preferences," *Journal of Artificial Intelligence Research*, pp. 505–549, 2006.

[33] D. A. Dolgov and E. H. Durfee, "Optimal resource allocation and policy formulation in loosely-coupled markov decision processes.," in *ICAPS*, pp. 315–324, 2004.

[34] J. Abounadi, D. Bertsekas, and V. S. Borkar, "Learning algorithms for markov decision processes with average cost," *SIAM Journal on Control and Optimization*, vol. 40, no. 3, 2001.

[35] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource allocation and cross-layer control in wireless networks*. Now Publishers Inc, 2006.

[36] H. Tan, Z. Han, X. Y. Li, and F. C. M. Lau, "Online job dispatching and scheduling in edge-clouds," in *IEEE INFOCOM 2017*.