Computer Networks 167 (2020) 106904

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

# When QoE meets learning: A distributed traffic-processing framework for elastic resource provisioning in HetNets[☆]

Li Yu[a], Zongpeng Li[a,*], Yucun Zhong[a], Zhenzhou Ji[b], Jiangchuan Liu[c]

[a] School of Computer Science, Wuhan University, Hubei, China
[b] School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[c] School of Computing Science, Simon Fraser University, Canada

ARTICLE INFO

*Article history:*
Received 28 November 2018
Revised 23 July 2019
Accepted 16 September 2019
Available online 23 September 2019

*Keywords:*
Reinforcement learning
Resource provisioning
Software defined infrastructure
QoE
Traffic-processing framework

ABSTRACT

In heterogeneous networks (HetNets), dynamics of user requirements across the temporal domain lead to the order of magnitude traffic to be processed by macro cells. To achieve high quality of experience (QoE) for users and to perform resource allocation for cells intelligently, we first propose a distributed traffic-processing framework (SDVTS) for elastic resource partitioning, to accommodate dynamics from the user-centric and resource-oriented perspectives respectively. Assisted by a software defined infrastructure, SDVTS fulfills the responsibilities of the request-based and push-based services in an interactive loop. Second, we formulate a traffic-processing time model that computes the delay of handling traffic. The non-convex model is decomposed and a dual evolution algorithm is explored to approximate the optimal solution. Furthermore, we introduce a low-complexity reinforcement learning algorithm with the personalized QoE profiling. A distributed algorithm in coalition between user and cell is designed for seamless connection of an advanced reinforcement learning system (ARLS) components and engines embedded in SDVTS. Extensive simulation results with thorough analysis demonstrate that our framework SDVTS dominates in terms of QoE and cell's system performance when compared with competing approaches.

## 1. Introduction

Triggered by the proliferation of mobile Internet service and Internet of Things, a large number of devices will be connected to wireless networks with massive new and dynamic traffic produced by them [1]. Users equipped with content-rich applications (UEs) expect more personalized and interactive services with small cells or macro base stations (MBSs) that are responsible for network accessing, resource partitioning and even task offloading towards tenants. These will inevitably introduce an unprecedented surge in coping with the traffic for cellular system. As illustrated in Fig. 1, the dense small cells are termed as SBSs, as exemplified by pico cells and femtocells. They are overlaid on the existing MBS. The SBSs and device to device (D2D) connections share the spectrum resources in an underlay fashion. There exist substantial traffic requirements from UEs to cells. Consequently, the computation-intensive and data-driven tasks are a driving force to explore efficient techniques that can significantly improve the system performance for the cells, making them adapt to the envisioned next generation networks with heterogeneity, especially for the ultra-dense HetNets. A natural, challenging problem is how to process the traffic immediately for better QoE (i.e., satisfying user requests or user-originated performance targets) and how to establish a smart resource allocation mechanism in MBS.

To our knowledge, there are mainly four lines of efforts to tackle the challenge. The first is to explore the small-cell architecture or the macro-cell one to improve the spatial efficiency of spectrum [2–4]. The second aims to maximize the resource utilization such as spectrum efficiency, power control and energy efficiency [5,6]. The third focuses on offloading cellular traffic to the other spectrum ranges such as WiFi, 60GHz wireless band [7] or to the mobile edge cloud [8]. The last is to maximize radio coverage and meet the QoE requirement by the optical fiber networks [9,10]. None of the existing work, however, has considered the software defined framework with learning-enabled resource provisioning, facilitating mutual feedback with QoE profiling for the request-based and push-based services.

[☆] This work was supported by the NSFC Grants (61502504, and 61628209), the Nature Science Foundation of Hubei Province (2016CFA030), the Technological Innovation Major Projects of Hubei Province (2017AAA125), the Science and Technology Program of Wuhan City (2018010401011288), and the China Scholarship Council.

[*] Corresponding author.
*E-mail addresses:* 2016302180129@whu.edu.cn (L. Yu), zongpeng@whu.edu.cn (Z. Li), jizhenzhou@hit.edu.cn (Z. Ji), jcliu@cs.sfu.ca (J. Liu).

https://doi.org/10.1016/j.comnet.2019.106904
1389-1286/© 2019 Published by Elsevier B.V.

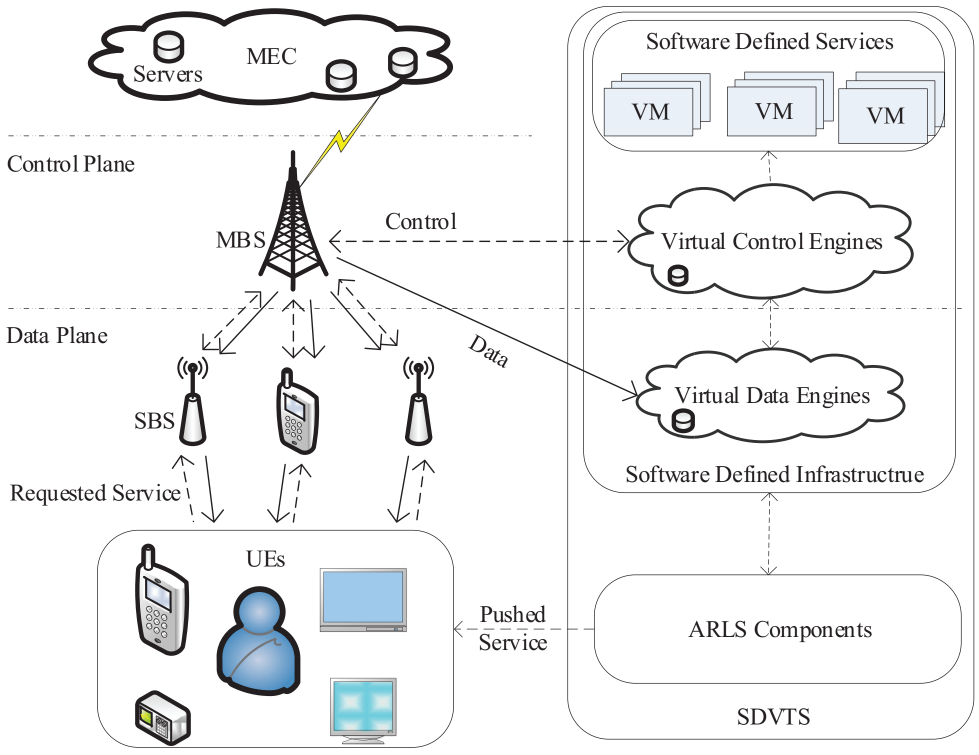


**Fig. 1.** The overview of serving UEs by MBS which is assisted by MEC and SDVTS for service in HetNets.

Driven by the availability of big data, mobile edge computing (MEC) [11] is evolving rapidly. It offers computing service with low delay for users through edge clouds or fog nodes deployed on the edge of networks, so as to address the challenge of timely offloading computation for cells, involving MBS. Furthermore, as a promising networking paradigm, software defined infrastructure (SDI) nowadays can decouple the control plane from the data plane through network function virtualization, achieving logically centralized control on the distributed UEs [12]. Besides, with advances in artificial intelligence and machine learning, smartness and automation have been considered as a new trend for domain-oriented applications. For example, AlphaGo [13] with deep reinforcement learning technology has brought great change in the era of artificial intelligence, making it feasible to realize smart learning which is analogous to human brain.

Assisted by MEC whose powerful capability of computation offloading and SDI whose specific capability of decoupling the signaling from data and control plane, we design a distributed traffic-processing framework of SDVTS (Software-defined Virtual Traffic System), where QoE meets learning. Our control plane consists of control engines responsible for managing communication, computing and storage resources of SDVTS; while the data plane incorporates physical radio interface equipments such as switches [14]. As shown in Fig. 1, SDVTS also exploits the software defined services driven by the virtual machine and the advanced reinforcement learning system (ARLS) components. The ARLS are developed by using a learning-enabled and low-complexity algorithm with QoE profiling, achieving resource provisioning intelligently despite of the explosive growth in network size.

We make the following contributions in this work.

- A software-defined framework, SDVTS, is proposed for traffic processing. SDVTS incorporates scalable ARLS components and two engines responsible for signal separation of data and control planes.
- SDVTS implements request-based and push-based services from the user-centric and resource-oriented perspectives, respectively. For the former, we formulate the traffic-processing time model and intuitively decompose the complex non-convex problem into sub-problems in an extreme manner, achieving the ultra-low-latency service for user requests. While, for the latter, we aim at adapting to the resource partitioning in dynamics via the ARLS components.
- A dual evolution algorithm is explored to approximate the optimal solver for power allocation, which is combined with Lagrangian multiplier and dual analysis approach to solve one of the sub-problems.
- A linear approximator of Q-learning with credit is introduced, which utilizes eligibility traces to resolve the impact of the delayed property which occurs in the procedure of reinforcement learning. Moreover, a factored approach with the tailored QoE profiling is leveraged to tackle the problem of the curse of dimensionality, expediting our algorithms convergence.
- A distributed algorithm in coalition between user and cell is devised for the seamless connection of the ARLS and engines in SDVTS.

The remainder of this paper is organized as follows. Section 2 reviews previous research. The SDVTS framework and problem formulation are elaborated on in Section 3. The method of an intuitive decomposition in an extreme fashion is presented in Section 4. In Section 5, we illustrate the learning-enable algorithm of the ARLS components with QoE profiling, and the in-depth analysis on simulations is described in Section 6. Finally, we conclude the paper in Section 7.

## 2. Related work

Substantial attention has been attracted in the macro cell paradigm, towards fulfilling the challenging mission of resource provisioning smartly and effectively.

In [2], the authors proposed to deploy small cells in dynamics, diminishing the voluminous traffic users sent. A new evolved

**Table 1**
Comparison of several traffic-processing schemes.

| Scheme in references | Computing resource pool | Scheme with learning | Task duration | QoE profiling |
|---|---|---|---|---|
| [15–18] | Local Cache | No | High | Low |
| [8] | Edge Cloud | No | Medium | Medium |
| [19–21] | Local Cache | Yes | Medium | Medium |
| SDVTS | Cache and Cloud | Yes | Low | High |

**Table 2**
Some notations.

| Notation | Description | Notation | Description |
|---|---|---|---|
| $k_u$ | Proportion of computing resource allocated to user by cell | $\mathcal{S}$ | Feasible set of states |
| $\vartheta^2$ | Variance of an additive white Gaussian noise | $\mathcal{A}$ | Feasible set of actions |
| $p_{u,b}$ | Transmission power of user associated with $m$ on $b$ | $T$ | Iterative episode |
| $C$ | Set of clouds for computation, where $c \in \{1, 2, \ldots, C\}$ | $p_m^{\max}$ | Maximal power available |
| $\aleph_u$ | Computing cycles of CPU consumed by edge cloud | $p_{u,b}^{need}$ | Needed power |
| $f_u$ | Computing capacity of CPU (cycles per second) | $r_{u,b}$ | Uplink data rate |
| $\widehat{\omega}$ | Weight of $Q(s, a)$ for each user-cell pair | $\rho$ | Convergence threshold of the weight |
| $e_{base}$ | Basic energy of maintaining communication | $p_m^{all}$ | Total transmitting power |
| $\Phi$ | Set of the non-negative Lagrangian multipliers | $Q_u$ | User's Q function |
| $c_u$ | Size of cloud set for computing the offloaded task user sent | $W$ | Cell's bandwidth |
| $w_{u,b}$ | Non-orthogonal spectrum bandwidth | $E^{\max}$ | Maximal energy of battery |
| $q(n)$ | Available energy at time slot $n$ on the user's side | $e_{loss}(n)$ | Consumed energy at $n$ |
| $D_u$ | Required number of CPU (cycles) for each task | $\varpi_{\mathbb{G}}$ and $\varpi_{\mathbb{E}}$ | Learning step size |
| $\varrho^n$ | Eligibility trace vector for $U$ features about users at $n$ | $\Xi$ | Set of parameters |
| $F_{u,c}(n)$ | Realization of computing capability $f_{u,c}$ | $\varepsilon^n$ | Probability of greedy selection |
| $R$ | The immediate revenue at next time slot | $\beta$ | The discount factor and $\beta \in [0, 1)$ |

packet core (EPC) infrastructure for next generation cellular networks was devised in [3]. The architecture and delay analysis in 5G networks were investigated from the perspective of cloud caching in [4].

Some recent literature made efforts to enhance resource utilization in [15–18] and [22], which focused on the Stackelberg game theory or matching theory to carry out resource scheduling, to some extent, achieving effective resource management. The above game theory literature often rely on ideal assumptions on nodes in the homogenous environment. Many efforts concentrated on the existence of Nash-equilibrium solution, which implied to find out the optimal solution. Nevertheless, it was not necessary to acquire the optimal system performance combined with QoE profiling. Based on service-driven 5G networks, the authors in [23] exploited a matching method to handle the non-convex optimization of spectrum resource allocation and power control in non-orthogonal multiple access networks or in HetNets. Similarly, the method of offloading resource with the interference was proposed in [24]. The authors in [25] formulated resource allocation as an optimization problem with multiple constraints. They split the problem into multiple sub-optimal problems, i.e., the Lagrangian dual approach and the method of subgradient using genetic algorithm. However, to the best of our knowledge, these approaches are not scalable with the explosive growth in network size. A scheme of resource assignment combined with cloud computing was designed in [26], which is compatible with the specific application of drones. To be intelligent, the macro cells have to learn to assign their scarce resources to achieve the high-effective resource utilization in dynamics.

Recent advances in computing (i.e., cloud/fog/edge computing) has profound impacts on task offloading and resource allocation. A task offloading approach with edge cloud was proposed [8]. While it was not taken into consideration that how to make marco cells learn resource provisioning smartly regardless of the ever-lasting growth in traffic. In [19,20] and [27], Q-learning approach for the dynamic resource allocation was adopted to address the problem of the co-existence between LTE and WIFI. In [20], the authors leveraged the linear approximator with the actor-critic reinforcement learning method to find out the optimal or suboptimal solution so that they could schedule and allocate resources effectively. However, there was no description about how to reduce the dimensionality of state-action space. Similarly, a linear and logistic regression approximator was utilized in [27]. However, the logistic regression model may constrain the scalability of approximator.

For the non-linear and non-convex optimization problem, a max-min fairness rate control approach using Perron–Frobenius theory was investigated in [21]. In [28], a non-linear reinforcement learning approach with the neural network was employed for the vehicle networking, which integrated the well-established dual network model in [29] and [30].

Considering the MEC and SDI, we devise a SDVTS framework that promotes the efficiency of resource utilization in an interactive loop and in return offers better QoE for users via pushing the cell's effective resources to user. Further, we formulate the traffic-processing time model and intuitively decompose the complexly non-convex problem into sub-problems in an extreme manner, achieving the ultra-low-latency service for users' requests. Moreover, a factored approach with the customized QoE profiling is leveraged to tackle the problem of the curse of dimensionality, expediting our algorithms convergence.

A taxonomy of various schemes is provided in Table 1, where the QoE profiling is divided into different levels (i.e., low, medium and high). The high level of QoE profiling means the user's requests are highly satisfied.

## 3. SDVTS framework and problem formulation

In this section, we will elaborate on the system model and problem formulation. For ease of reference, notations are summarized in Table 2.

### 3.1. SDVTS framework

A schematic framework of SDVTS that is embedded in MBS is depicted in Fig. 2. It incorporates ARLS components and two engines. The UEs send requests to cell for service. MBS calls the data engine to handle the requested service as soon as possible according to its remaining resources, involving cloud and spectrum

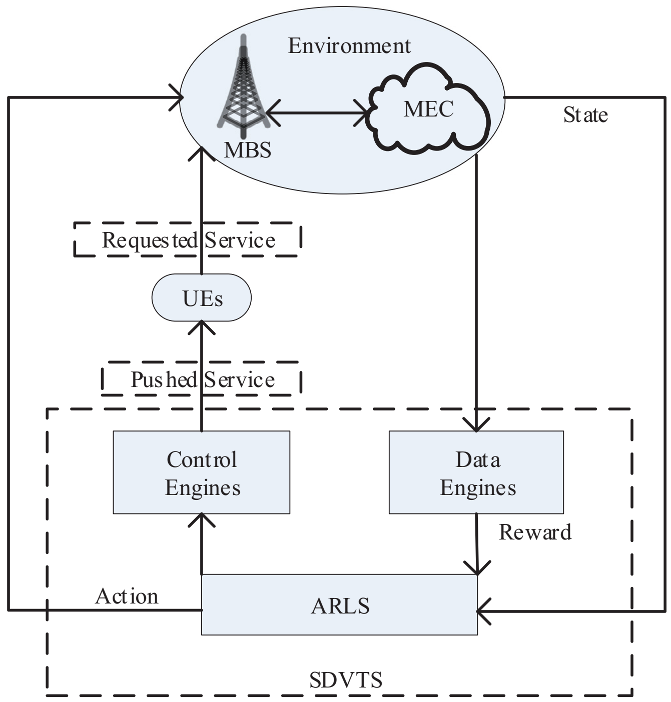


**Fig. 2.** The schematic framework of SDVTS in an interactive loop.

resources that are analyzed by the ARLS components. The results of the in-depth analysis, on one hand, realize the resource partitioning for the cell. On the other hand, they are sent to users via signaling constructed by control engines, pushing them to make a better choice of requesting resources or traffic, facilitating positive interaction for the resource utilization. Besides, the learning agent of the components in SDVTS observes the environmental state and receives its reward (or penalty). Then it takes certain appropriate action under certain policy.

The framework uses the ARLS for an interactive feedback to cells, which is capable of processing unprecedented growth of spatio-temporal traffic intelligently. It can be applied to many new delay-critical and computation-sensitive applications such as the unmanned aerial vehicle (UAV) [31], the tactile Internet [32] and runtime control on robotics [33].

### 3.2. Communication model

MBS has multiple cloud servers to manage small cells and users within its coverage. The traffic is mainly to be handled by macrocells. If necessary, the MBS will decide to offload a portion of the workload to the clouds according to the result of our proposed smart learning algorithms. Without loss of generality, the path loss has been considered and can be calculated. Let $g_{u,b}$ denote channel gain between the UEs $u$ $(u \in \{1, 2, \ldots, U\})$ and the virtual channel $b$ $(b \in \{1, 2, \ldots, B\})$. $g_{u,b}$ can be known by the pilot signal sent by cell when cell is associated with UEs. $U$ and $B$ are allocated by the cell. $B^h$ stands for the set of occupying sub-channels and $B^h \subseteq B$. The other users′ interference between the user $u$ and the channel set $B^h$ is viewed as $\sum_{d\in\{B^h\setminus\{b\}\}} g_{u,d}p_{u,d}$. According to the Shannon theory [34], we can obtain the uplink data rate $r_{u,b}$ as follows.

$$r_{u,b} = W\log_2(1 + SINR) \tag{1}$$

$$SINR = \frac{g_{u,b}p_{u,b}}{\sum_{d\in\{B^h\setminus\{b\}\}} g_{u,d}p_{u,d} + \vartheta^2} \tag{2}$$

where $SINR$ in Eq. (2) represents the received signal-to-interference-plus-noise ratio that has the intra-interference of the user $u$ associated with cell on the channel $b$. For simplification, we assume that the spectrum band $W$ is a constant given by the cell that has the information about other users′ power status. Based on the communication model, we present the following problem formulation.

### 3.3. Problem formulation

The users send requests for accessing the Internet or uploading (downloading) data to (from) the MBS dynamically. Once connection is established between the user and MBS at certain time slot, cells tend to cope with the corresponding service as soon as possible. However, since the caching, computing and spectrum resources in macro cells are restricted, MBSs are incapable of handling all tasks users sent. They would rather choose to deliver the part traffic to the MEC than deliver the whole task to them if the traffic can be offloaded. Because users can timely obtain the part traffic that is directly dealt with by the MBS. The waiting time of processing traffic will be sharply shortened. This is different from the existing work [8] where the traffic is handled either entirely by local processing units or totally by MEC. Nevertheless, how to offload the traffic, how much it is offloaded to MEC and whether it can be offloaded or not are beyond our concerned scope, which is detailed in [35]. Our focus is that the MBS can still handle the part traffic, achieving the ultra-low delay for user′s experience since the whole task for offloading to the MEC needs the waiting time which significantly affects the QoE.

Let $M^n(Z_u, D_u)$ denote a requested task in certain time instant $n$ $(n \in \{1, 2, \ldots, T\})$, where $Z_u$ is the traffic size acquired by the user $u$. The cell judges whether the request can be processed by itself or not according to its resource status and the computing capability. If the cell $m$ can completely cope with the task, the delay of tackling traffic is denoted as $t_{u,m}$, i.e., $t_{u,m} = \frac{Z_u}{r_{u,b}}$. Otherwise, the delay mainly involves the time $t_{u,m}$ produced by the cell for data processing and the part time $t_{u,c}$ caused by the cloud which processes the rest of the task user $u$ sent. Namely the totally traffic-processing time is $t_u^{all}$. $t_u^{all} = t_{u,m} + t_{u,c} = \frac{\overline{Z_u}}{r_{u,b}} + \frac{\aleph_u}{k_u f_u}$, where the size of traffic that is handled by the cell is $\overline{Z_u}$. The MEC will process the rest amount of traffic that is denoted as $(Z_u - \overline{Z_u})$ and transform it into the computing amount of CPU denoted as $\aleph_u$. We hypothesize the time of offloading the remainder of traffic to the cloud is negligible. The reason is that we anticipate the traffic user sent can be entirely processed by cell as possible as it can so that the user can obtain the feedback timely by our proposed framework. Further, the traffic for offloading is the small fraction. Based on the user-centric principle, we focus on the processing time for the traffic users sent. The main consumed resource of the cloud is measured by CPU. We expect to minimize the cell′s time of handling traffic, enhancing the QoE for all users. That is, the optimization problem **P1** is given as follows.

$$\mathbf{P1}: \min_{Y,X,P,K} \sum_{u=1}^{U}\sum_{b=1}^{B} y_{u,b}^n[x_{u,m}t_{u,m} + (1 - x_{u,m})(t_{u,m} + t_{u,c})] \tag{3}$$

$$\begin{array}{lll} s.t. \; C1 & \sum_{u=1}^{U}\sum_{b=1}^{B} y_{u,b}^n p_{u,b} = p_m^{all} & \forall u \in U, \forall b \in B, \forall n \in T. \\ C2 & x_{u,m} \in \{0, 1\} & \forall u \in U. \\ C3 & p_{u,b} \geq 0 & 0 \leq p_m^{all} \leq p_m^{max}. \\ C4 & \sum_{u=1}^{C_u} k_u \leq 1 & \forall c_u \subseteq C. \\ C5 & y_{u,b}^n \in \{0, 1\} & \forall u \in U, \forall b \in B, \forall n \in T. \end{array}$$

where the variables of $Y$, $X$, $P$ and $K$ represent the corresponding set respectively, i.e., $Y = [y_{u,b}^n]$, $X = [x_{u,m}]$, $K = [k_u]$, $P = [p_{u,b}]$. $x_{u,m}$ and $y_{u,b}^n$ are both the indicator function. If $y_{u,b}^n = 0$, the user $u$ does not occupy the channel $b$ at time instant $n$. It is one otherwise.

If $x_{u,m} = 1$, the traffic the user $u$ sent can be processed by the cell completely. Otherwise, the task is tackled by cell and clouds jointly.

As mentioned above, the objective function is given by

$$G(.) = \sum_{u=1}^{U}\sum_{b=1}^{B} y^n_{u,b}\left[x_{u,m}\left(\frac{Z_u}{r_{u,b}}\right) + (1 - x_{u,m})\left(\frac{\overline{Z_u}}{r_{u,b}} + \frac{\aleph_u}{k_u f_u}\right)\right] \quad (4)$$

where the function of $G(.)$ is with respect to $y^n_{u,b}$, $x_{u,m}$, $p_{u,b}$ and $k_u$.

Since there exist binary variables, **P1** is non-convex, implying that it cannot be addressed efficiently by numerical methods similar to gradient approaches. We decompose the problem in an extreme fashion in Section 4.

## 4. Decomposition method

Owing to the discrete nature of problem **P1**, where $y^n_{u,b}$ and $x_{u,m}$ are binary, the cell cannot compute the specific values on the fly. We decompose the non-convex problem into sub-problems in an extreme pattern, which is different from the literature in [25] where the authors addressed the problem using some given parameters. However, the fixed parameters are not applicable to real-world settings without *a priori* knowledge.

### 4.1. The solver of $y^n_{u,b} = 1$, $x_{u,m} = 1$

Assuming that the channel $b$ allocated to the user $u$ is occupied at the time instant $n$ and the cell associated with the user $u$ can manage the user's task. Namely $y^n_{u,b} = 1$, $x_{u,m} = 1$. The **P1** is transformed into the following **P1-1**.

$$\mathbf{P1-1}: \quad \min_{P} \sum_{u=1}^{U}\sum_{b=1}^{B}\frac{Z_u}{r_{u,b}} \qquad s.t. \quad C1 \;\&\; C3 \quad (5)$$

Substituting Eqs. (1) and (2) into **P1-1**, we acquire Eq. (6) as follows.

$$G_1(p_{u,b}) = \sum_{u=1}^{U}\sum_{b=1}^{B}\frac{Z_u}{W\log_2\left(1 + \frac{g_{u,b}p_{u,b}}{\sum_{d\in\{B^h\setminus\{b\}\}} g_{u,d}p_{u,d} + \vartheta^2}\right)} \quad (6)$$

It is known that $r_{u,b}$ is a monotonically increasing function with respect to $p_{u,b}$. However, it is not necessary to be the convex function for Eq. (6) (Proof of convexity is relatively straightforward and is left out). Hence, we transform it into its dual problem because of the convexity of the dual issue.

For simplicity, let $\varphi(p_{u,b}) = W\log_2(1 + \tau p_{u,b})$, where $\tau = \frac{g_{u,b}}{\sum_{d\in\{B^h\setminus\{b\}\}} g_{u,d}p_{u,d} + \vartheta^2}$. We can pick up the Lagrangian function of the optimization problem **P1-1** about $C1$ and $C3$ as follows.

$$L(P,\Phi) = \sum_{u=1}^{U}\sum_{b=1}^{B}\frac{Z_u}{\varphi(p_{u,b})} + \sum_{b=1}^{B}\phi_b\left(\sum_{u=1}^{U} p_{u,b} - p^{max}_m\right) \quad (7)$$

where let $\phi_b$ denote the $b$th Lagrangian multiplier in the set $\Phi$.

Then, the dual function is given by

$$O(\Phi) = \begin{cases}\min & L(P,\Phi)\\ s.t. & C1\,\&\,C3\end{cases} \quad (8)$$

The dual problem of **P1-1** is expressed as $\max_\Phi O(\Phi)$.

Let $\phi^*$ denote one of the optimal Lagrangian multipliers. Namely, $\phi^* = \arg\max_\Phi \min_P L(P,\Phi)$. Next, we present the dual evolution algorithm for power provisioning in order to find out the $\phi^*$. Since $p^{max}_m - \sum_{u=1}^{U} p^*_{u,b}(\phi^n)$ is one of the subgradients for the dual problem, we can obtain $\phi^*$ according to the following convergence equation.

$$\phi^{(n+1)} = \left[\phi^n - \alpha^n\left(p^{max}_m - \sum_{u=1}^{U} p^*_{u,b}(\phi^n)\right)\right]^+ \quad (9)$$

where $[x]^+ = \max\{x, 0\}$. $\alpha^n$ denotes the iterative step size, which satisfies the conditions that $\alpha^n \to 0$ as $n \to \infty$ and $\sum_{n=1}^{\infty}\alpha^n = \infty$ [36].

To ensure the cell to adapt to the dynamic environment, it is necessary to adjust the currently optimally fixed power assignment even if we obtain optimal or near-optimal solution via Algorithm 1 (lines 1–11). Similar to the work [25], we utilize the Mann iterative [37] to update the power, making it evolvable (lines 12–20).

**Algorithm 1** Dual Evolution Algorithm for Power Provisioning.

```
Input: Initialize $p^{max}_m$, convergence threshold $\Lambda$.
Output: $p^*_{u,b}$.
1:  while $(\|\phi^{(n+1)} - \phi^n\| \geq \Lambda)$ do
2:      if $(y^n_{u,b} = 1)$ then
3:          $p^*_{u,b} = \arg\min_{p_{u,b}}\left(\frac{Z_u}{\varphi(p_{u,b})} + \phi^n p_{u,b}\right) s.t. 0 \leq p_{u,b} \leq p^{max}_m$
4:          $p^{need}_{u,b} = \sum_{u=1}^{U}\sum_{b=1}^{B} p^*_{u,b}$
5:          $\phi^{(n+1)} = \left[\phi^n - \alpha^n(p^{max}_m - p^{need}_{u,b})\right]^+$
6:      else
7:          Call Algorithm 2 to offer available channels
8:      end if
9:  end while
10: $\phi^* = \phi^{(n+1)}$
11: $p^*_{u,b} = \arg\min_{p_{u,b}}\left(\frac{Z_u}{\varphi(p_{u,b})} + \phi^* p_{u,b}\right) s.t. 0 \leq p_{u,b} \leq p^{max}_m$
12: Update power: $p^{max}_m = p^{max}_m - p^{need}_{u,b}$
13: for $(i = u, n = 0; i \leq U, n < T;)$ do
14:     $p^{(n+1)}_{i,b} = (1 - \lambda^n)p^n_{i,b} + \lambda^n p^*_{u,b}$, where $\lambda^n = \frac{n}{2n+1}$ [37]
15:     if $\left(\sum_{i=1}^{U}\sum_{b=1}^{B} p^{(n+1)}_{i,b} \leq p^{max}_m\right)$ then
16:         $i++$
17:     else
18:         $n++$
19:     end if
20: end for
```

It is observed that the optimal value $\phi^*$ is the lower bound of the primal problem. The bound will be used in the section of performance evaluation.

### 4.2. The solver of $y^n_{u,b} = 1$, $x_{u,m} = 0$

Suppose that the channel the cell allocated is occupied by the corresponding user and that the traffic the user $u$ sent is handled by the cell and cloud $c$ jointly, i.e., $y^n_{u,b} = 1$, $x_{u,m} = 0$. The **P1** is decomposed into **P1-2**.

$$\mathbf{P1-2}: \quad \min_{P,K}\sum_{u=1}^{U}\sum_{b=1}^{B}\left(\frac{\overline{Z_u}}{r_{u,b}} + \frac{\aleph_u}{k_u f_u}\right) \qquad s.t. \quad C1,\; C3 \;\&\; C4 \quad (10)$$

Substituting Eqs. (1) and (2) into **P1-2**, we derive

$$G_2(p_{u,b}, k_u) = \sum_{u=1}^{U}\sum_{b=1}^{B}\left(\frac{\overline{Z_u}}{W\log_2(1 + \tau p_{u,b})} + \frac{\aleph_u}{k_u f_u}\right) \quad (11)$$

There exist two variables with respect to $P$ and $K$. Given $P = [p_{u,b}]$ by Algorithm 1, we can simplify Eq. (11), yielding $G_3(k_u)$ as depicted as

$$G_3(k_u) = \sum_{u=1}^{U}\sum_{b=1}^{B}\left(\mathcal{C} + \frac{\aleph_u}{k_u f_u}\right) \tag{12}$$

where let $K = [k_u]$ denote the set of the proportion about the cloud resources which are allocated to the user's task. $\mathcal{C}$ represents a constant, which replaces $\frac{\overline{Z_u}}{r_{u,m}}$ in Eq. (10). Because it is known according to the context.

**Lemma 1.** *The function $G_3(k_u)$ is convex when it is subjected to the condition C4 in Section 3.3.*

**Proof.** please see appendix. □

Hence, we can exploit the Lagrangian approach to solve the globally optimal solution as follows.

$$\Gamma(K,\Theta) = \sum_{u=1}^{U}\sum_{b=1}^{B}\left(\mathcal{C} + \frac{\aleph_u}{k_u f_u}\right) + \sum_{b=1}^{B}\theta_b\left(\sum_{u=1}^{c_u} k_u - 1\right) \tag{13}$$

Since the gradient of Eq. (13) is obtained, the constant $\mathcal{C}$, in essence, does not affect the optimal solution with regard to $K$. We prefer to adopt $k_u^* = k_u$. The optimal solution is given by

$$k_u^* = \frac{\sqrt{\aleph_u}}{\sum_{u=1}^{c_u}\sqrt{\aleph_u}} \tag{14}$$

where $c_u$ is not the variable. Once the traffic request is built, the cell will obtain its value.

Substitute $p_{u,b}^*$ obtained by Algorithm 1 and $k_u^*$ in Eq. (14) above into Eq. (11). The optimal solution of $G_4(k_u^*, p_{u,b}^*)$ is calculated as

$$\begin{aligned} G_4(k_u^*, p_{u,b}^*) &= \sum_{u=1}^{U}\sum_{b=1}^{B}\left(\frac{\overline{Z_u}}{W\log_2(1+\tau p_{u,b}^*)} + \frac{\aleph_u}{k_u^* f_u}\right) \\ &= \sum_{u=1}^{U}\sum_{b=1}^{B}\left(\frac{\overline{Z_u}}{W\log_2(1+\tau p_{u,b}^*)} + \frac{\sum_{u=1}^{c_u}\aleph_u}{f_u}\right) \end{aligned} \tag{15}$$

## 5. Double approximated learning algorithm with QoE profiling in ARLS components

As the compensation for the extreme method, a learning-enabled scheme is designed, making the variables flexible such as $y_{u,b}^n$ and $x_{u,m}$. To make the MBS intelligent, we develop a double approximated learning algorithm with the tailored QoE profiling in an online-offline fashion, which is the main function of the ARLS in SDVTS.

### 5.1. Bellman equation

It is generally acknowledged that the optimally stochastic control problem of resource allocation can be constructed as the discrete time Markov decision process (MDP) with the continuous spaces concerning state and action [38].

The macro cells learn to take next action $a'$ $(a' \in \mathcal{A})$ according to certain policy $\pi$ at the current state $s$ $(s \in \mathcal{S})$ and the current action $a$ $(a \in \mathcal{A})$ to yield the lower cost (or higher profit) of $Q(s, a)$ given by Bellman equations [39] in a recursive manner as follows.

$$Q(s,a) = E_{s'}\left[R + \beta\sum_{s'\in\mathcal{S}} Pr(s'|s,a)\min_{a'\in\mathcal{A}} Q^{\pi}(s',a'|a)\right] \tag{16}$$

where $E[.]$ refers to the expected rewards in the long term. $s'$ is next state. $Pr(.)$ is the transition probability. In other word, the Q function captures the expectations of cumulative costs (or rewards) with discount when the system takes the action $a$ at $s$.

The objective of Q-learning is to learn how to map environment states to the optimal actions under certain policy after gathering the learning experience by trial and error. The optimal policy $\pi$ is obtained by $\pi^{(n+1)}(s,a) = \arg\min_a Q^{\pi_n}(s,a)$ for $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$.

### 5.2. Enhanced Q function

Due to the time-varying channels during the procedure of serving users, we construct the communication model as the finite-state Markov chains, which is widely accepted to act as an effective approach of characterizing the correlation structure of fading process [40]. Therefore, the *SINR* as a parameter is used to measure the quality of channel. The received *SINR* for user $u$ on channel $b$ is viewed as a random variable $h_{u,b}$ whose range can be quantized into the discrete levels with the number of $H$. Namely, $H_0$, if $h_0 \le h_{u,b} < h_1$; $H_1$, if $h_1 \le h_{u,b} < h_2$;...; $H_{H-1}$, if $h_{u,b} \ge h_{H-1}$. Each level corresponds to a state of a Markov chain, forming $H$-element state space which is denoted as $S^h = \{S_0^h, S_1^h, S_{H-1}^h\}$. The channel state of realization of $h_{u,b}$ at time instant $n$ can be denoted as $H_{u,b}(n)$. The received $h_{u,b}$ of *SINR* varies from one state $s_h$ to another $s_h'$ when one time slot elapses with the uncertain transition probabilities of $\mathbb{P}_{u,b}(n)$ at time slot n. The $H \times H$ matrix of the state transition probability for user $u$ occupying the channel $b$ is defined as

$$\mathbb{P}_{u,b}^{pipe}(n) = \left[Pr(H_{u,b}(n+1)) = s_h' | H_{u,b}(n) = s_h\right]_{H\times H} \tag{17}$$

where $s_h, s_h' \in S^h$. Let $\mathbb{J}$ (bit per second) denote the wireless communication capacity between the cell $m$ and its all associated users. Owing to the interference to the user $u$ from other users that are also connected to the cell, the achievable spectrum efficiency of the user $u$ connecting to the channel $b$ can be expressed as $\mathbb{k}_{u,b}$. It can be easily calculated by the cell after the information exchange with the user. Consequently, the communication rate ${R_{u,b}}^{pipe}$ for the user $u$ at time slot $n$ can be given by

$$R_{u,b}^{pipe}(n) = y_{u,b}(n) * w_{u,b}(n) * \mathbb{k}_{u,b}(n), \forall u \in U, \forall b \in B \tag{18}$$

The sum rate of all users that are associated with the cell on $b$ should satisfy the inequality that $\sum_{u=1}^{U}\sum_{b=1}^{B} R_{u,b}^{pipe}(n) \le \mathbb{J}$.

Consider let $f_{u,c}$ denote the cell's computational capability that is allocated to the user $u$. $f_{u,c}$ is measured by CPU (cycles per second). In the wireless networks, different requests have different computing speed for the cell. Thus the $f_{u,c}$ can be viewed as a random variable and divided into discrete levels denoted by $S^c = \{S_0^c, S_1^c, \ldots, S_{N-1}^c\}$, where $N$ is the number of states about computing capability available. We hypothesize the realization of the capability for $f_{u,c}$ is represented as $F_{u,c}(n)$ at time slot $n$. Similar to the channel power allocation, the corresponding transition probability matrix of the cell $m$ for the user $u$ is defined as

$$\mathbb{P}_{u,c}^{comp}(n) = [Pr(F_{u,c}(n+1)) = s_c' | F_{u,c}(n) = s_c]_{N\times N} \tag{19}$$

where $s_c, s_c' \in S^c$. The running time of computing the traffic $M^n(Z_u, D_u)$ at the cell can be calculated as $t_{u,c} = \frac{D_u}{F_{u,c}(n)}$. We formulate the computing rate (bits computed per second) as

$$\begin{aligned} R_{u,c}^{comp}(n) &= x_{u,m}(n) * y_{u,b}(n) * \frac{Z_u}{t_{u,c}} \\ &= x_{u,m}(n) * y_{u,b}(n) * \frac{Z_u * F_{u,c}(n)}{D_u} \end{aligned} \tag{20}$$

where the caching volume of cell should satisfy the inequality of $\sum_{u=1}^{U}\sum_{b=1}^{B} y_{u,b}(n) * Z_u \le \mathbb{O}$ where $\mathbb{O}$ is the total caching volume for caching user's tasks.

Furthermore, we define the immediate cost (or payoff) $R$ on the cell's side as

$$R(n) = R_{u,b}^{pipe}(n) + R_{u,c}^{comp}(n) \tag{21}$$

As aforementioned, we present the following definition of Q function.

**Definition.** Q function of cell: $Q_m(\upsilon, \mathcal{A}) = Q_m(\upsilon(n), \mathcal{A}(n))$, where $\upsilon(n) = [H^{\mathcal{T}}_{u,b}(n), F^{\mathcal{T}}_{u,c}(n)]^{\mathcal{T}}$. $H^{\mathcal{T}}_{u,b}(n) = [H^1_{u,b}(n), H^2_{u,b}(n), \ldots, H^B_{u,b}(n)]^{\mathcal{T}}$, indicating the status of the logical channels for the user at time slot $n$ in cell, which is described above. $F^{\mathcal{T}}_{u,c}(n) = [F^1_{u,c}(n), F^2_{u,c}(n), \ldots, F^C_{u,c}(n)]^{\mathcal{T}}$, indicating the situation of the current resource about cloud. $\mathcal{T}$ is the transpose of a matrix or vector. $\mathcal{A}(n) = [Y^{\mathcal{T}}(n), X^{\mathcal{T}}(n)]^{\mathcal{T}}$ is a binary matrix that $Y(n) = [y^n_{u,b}]$ and $X(n) = [x_{u,m}]$ for $\forall u \in U$, $\forall b \in B$, $y_{u,b}(n) \in \{0, 1\}$ and $x_{u,m}(n) \in \{0, 1\}$.

**Definition.** Q function of user: $Q_u(\Psi, \mathcal{A}) = Q_u(\psi(n), \mathcal{A}(n))$, where $\Psi(n) = [q^n_u, l^n_u]^{\mathcal{T}}$. $l(n)$ is the priority of handling the traffic, which is picked up by the fields in the package user sent. $q(n) = E^{\max} - e_{loss}(n)$.

Combined with the QoE at the user side, we model the advanced Q function as

$$Q(s, a) \triangleq Q_u(\Psi, \mathcal{A}) + Q_m(\upsilon, \mathcal{A}) \tag{22}$$

where we use "$\triangleq$ " to denote "is defined to be equal to" and use $\|.\|$ to denote the $\mathcal{L}^2$ norm in this paper.

Subsequently, a approximated Q function with eligibility traces is described in detail.

### 5.3. Approximated Q function with eligibility traces

There are many approximators such as the linear and non-linear ones. Nevertheless, reinforcement learning algorithms tend to diverge when used with the non-linear function approximators like the neural network [28]. Whereas the linear one is a popular approach for rendering the Q-learning algorithm applicable to the real-world settings [20]. To guarantee the fast convergence and the stability, we, therefore, develop a novel linear Q-function approximator $\widehat{Q}(s, a)$ as follows.

$$\widehat{Q}(s, a) = \widehat{\omega}^{\mathcal{T}} Q(s, a) \tag{23}$$

The temporal difference error (TD error) $\delta$ stands for the gap between the approximated value of the Q-function at next time slot and the current value, which is usually estimated as

$$\delta^n = R^{(n+1)} + \beta \widehat{Q}(s^{(n+1)}, a^{(n+1)}) - \widehat{Q}(s^n, a^n) \tag{24}$$

Reinforcement learning is a method of interaction with environment, making it have the property of delayed reward (or penalty). To be specific, the current action may impact the immediate cost (or reward), so do the following ones in all subsequent time steps. Consequently, we introduce the eligibility traces that offer more efficient weight by assigning the credit to the previous states and actions that are experienced and by recording the historical value of Q function temporarily.

The updated equation of the eligibility trace vector [41] is given by

$$\varrho^{(n+1)} = \ell \beta \varrho^n + Q(s, a) \tag{25}$$

where $\ell$ is the trace-decay parameter and its range is [0,1). If $\ell = 0$, it will be updated as the current Q value achieved. Since it accumulates the state-action pair for each step, the eligibility trace increases for the Q value achieved at this step. The trace will decline gradually if the Q value is not achievable, which suggests that the current state-action pair achieved are more eligible for participating in the learning procedure. Thereby, the non-trivial weight of the approximated Q function becomes

$$\widehat{\omega}^n = \xi^n \delta^n \varrho^n \tag{26}$$

where $\xi$ is the updated rate of the following Q function and $\xi \in [0, 1)$.

The update of Q function proceeds with the iteration as follows.

$$\begin{aligned}\widehat{Q}^{(n+1)}(s^n, a^{(n+1)}) &= (1 - \xi^n)\widehat{Q}^n(s^n, a^{(n+1)}) \\ &+ \xi^n \Big[ R(s^n, a^{(n+1)}) + \beta \min_{a^{(n+1)}} \widehat{Q}^n(s^{(n+1)}, a^{(n+1)}) \Big]\end{aligned} \tag{27}$$

### 5.4. Factored Q-learning algorithm and SDVTS mechanism

To endow our algorithm not only with the fast convergence but also with the scalability, we propose a factored learning approach with low complexity subsequently.

Recall that we have redefined the Q function with QoE profiling. Namely $Q(s, a) \triangleq Q_u(\Psi, \mathcal{A}) + Q_m(\upsilon, \mathcal{A})$. The first term of Q function on the user's side is expressed as

$$Q_u(\Psi, \mathcal{A}) \triangleq \sum_{u=1}^{U} \zeta_u \mathbb{1}_{\{q^n_u \geq e_{base}\}} \tag{28}$$

where the symbol of the summation represents all the requests users sent. The indicator function $\mathbb{1}_{\{.\}}$ takes the value 1 if its argument holds and 0 otherwise. While $\zeta_u$ is the average cost of handling the request if the user's energy left is larger than the basic energy for transmitters or receivers at time slot $n$. We can describe the user's Q function as

$$Q_u(\Psi, \mathcal{A}) = \mathbb{G}^{\mathcal{T}}(q^n) \tag{29}$$

where $\mathbb{G}(q^n) = [\zeta(q^n_1 \geq e_{base}), \zeta(q^n_2 \geq e_{base}), \ldots, \zeta(q^n_U \geq e_{base})]^{\mathcal{T}}$.

Similarly, The second term of Q function at the cell side is described as

$$Q_m(\upsilon, \mathcal{A}) \triangleq \sum_{g=0}^{H-1} \sum_{f=0}^{N-1} \epsilon_{g,f} \mathbb{1}_{\{s^h_g = 1\}} \mathbb{1}_{\{s^c_f = 0\}} \tag{30}$$

where it is different from the first term that the symbol of the summation represents all the levels of *SINR* and the utilizable cloud resources which is elaborated on in Section 5.2; while $\epsilon_{g,f}$ captures the average cost of system if the subchannel is occupied and the traffic is dealt with by cell without the help of cloud servers who cope with the offloaded traffic. By defining the $H \times N$ matrix with $(g, f)^{th}$ entry $\mathbb{E} = [\epsilon_{g,f}]$, $Q_m(\upsilon, \mathcal{A})$ is rewritten as

$$Q_m(\upsilon, \mathcal{A}) = \mu^{\mathcal{T}}(S^h_g)\mathbb{E}(1 - S^c_f) \tag{31}$$

where $\mu(S^h_g) = [\mu(S^h_0), \mu(S^h_1), \ldots, \mu(S^h_{H-1})]^{\mathcal{T}}$.

Then we define $\Xi \triangleq \{\mathbb{G}, \mathbb{E}\}$. The original Q function of Eq. (22) can be redefined as

$$Q_{\Xi}(s, a) = \mathbb{G}^{\mathcal{T}}(q^n) + \mu^{\mathcal{T}}(S^h_g)\mathbb{E}(1 - S^c_f) \tag{32}$$

Clearly, the learning function is transformed from the original Q function $Q(s, a)$ to the novel function $Q_{\Xi}(s, a)$. The original Eq. (23) about Q function is replaced by the double approximators as follows.

$$\widehat{Q}(s, a) = \widehat{\omega}^{\mathcal{T}} Q_{\Xi}(s, a) \tag{33}$$

Note that we mainly address the problem about the learning parameter set $\Xi$.

Given the current estimates of parameters $\{\mathbb{G}, \mathbb{E}\}$ at $n$ after the phase of fulfilling the information exchange at next time slot $(n + 1)$, the instantaneous error is calculated by

$$\begin{aligned}\widehat{e}(s^n, a^{(n+1)}) &\triangleq R(s^n, a^{(n+1)}) \\ &+ \beta \min_{a^{(n+1)}} Q_{\Xi_n}(s^{(n+1)}, a^{(n+1)}) - Q_{\Xi_n}(s^n, a^{(n+1)})\end{aligned} \tag{34}$$

The average value of instantaneous error is defined as

$$\eta(s^n, a^{(n+1)}) \triangleq \frac{1}{2}(\widehat{e}(s^n, a^{(n+1)}))^2 \tag{35}$$

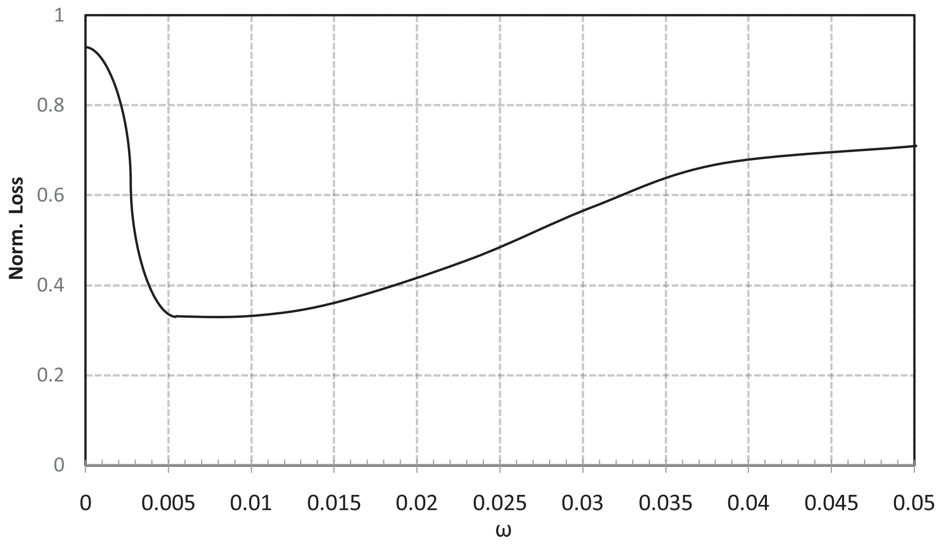


**Fig. 3.** Impact of $\varpi$ on the normalized loss function of Q-learning.

The updated rule of parameter are obtained by utilizing the iterations of the stochastic gradient descent, which is described as

$$\begin{aligned}\mathbb{G}^{(n+1)} &= \mathbb{G}^n - \varpi_{\mathbb{G}} \nabla \eta(s^n, a^{(n+1)}) \\ &= \mathbb{G}^n + \varpi_{\mathbb{G}} \widehat{e}(s^n, a^{(n+1)}) \nabla_{\mathbb{G}} Q_{\Xi_n}(s^n, a^{(n+1)}) \\ &= \mathbb{G}^n + \varpi_{\mathbb{G}} \widehat{e}(s^n, a^{(n+1)}) \mathbb{G}^{\mathcal{T}} \qquad (36)\end{aligned}$$

$$\begin{aligned}\mathbb{E}^{(n+1)} &= \mathbb{E}^n - \varpi_{\mathbb{E}} \nabla \eta(s^n, a^{(n+1)}) \\ &= \mathbb{E}^n + \varpi_{\mathbb{E}} \widehat{e}(s^n, a^{(n+1)}) \mu(S_g^h(n))(\nVdash - S_f^c)^{\mathcal{T}} \qquad (37)\end{aligned}$$

where $\varpi_{\mathbb{G}}$ and $\varpi_{\mathbb{E}}$ are both set to 0.005 that are obtained by our simulation presented in Fig. 3 wherein the $\varpi$ stands for the two parameters of $\varpi_{\mathbb{G}}$ and $\varpi_{\mathbb{E}}$. If the learning step size is set to the large value, which speeds up the learning procedure. It is easy to miss the optimal value, which is acquired from Fig. 3 that the normalized loss function is increasing. The function will be supervising the Q-learning procedure.

We notice that the updates of $Q_{\Xi}(s, a)$ as follows is similar to Eq. (27). However, the low-complexity approximator of Q function is $Q_{\Xi}(s, a)$ instead of the original $Q(s, a)$.

$$\begin{aligned}Q_{\Xi_{(n+1)}}(s^n, a^{(n+1)}) &= (1 - \xi^n) Q_{\Xi_n}(s^n, a^{(n+1)}) \\ &+ \xi^n \left[ R(s^n, a^{(n+1)}) + \beta \min_{a^{(n+1)}} Q_{\Xi_n}(s^{(n+1)}, a^{(n+1)}) \right] \qquad (38)\end{aligned}$$

The Q-learning algorithm with the tailored QoE profiling is tabulated in Algorithm 2, where the low-complexity learning procedure occurs in an iterative manner at the online stage (lines 1–6), which implies the Q function does not converge. Note that line 6 is the updated process of Q function, which is also the learning process online. If the objective function converges, the offline learning occurs. The QoE profiler is produced in the offline learning phase (lines 8 to 25). Combined with the eligibility traces, we exploit the least square method to further adjust the weight, circumventing the problem of the delayed reward (or penalty) in the reinforcement learning algorithm. Combined with our improved Q function and the weight, the offline learning is to update the weight and the other parameters that are needed in the deep convolutional neural network (DNN). The updated parameters are in return guiding the selection of next action (line 2), which is at the online stage. From the perspective of the algorithm complexity, we have done massive related experiments. Then we confirm that if the offline learning is revised as the online learning, the results show the complexity is higher than that with our proposed Algorithm 2. Furthermore, the delay of feedback is larger, which impact the user's QoE. Note that there exist two-fold improvements, namely the Q function itself and the credit weight with the eligibility traces.

**Algorithm 2** Q-learning Algorithm with Tailored QoE Profiling.

```
Input: $Q(s,a) \leftarrow 0$; $n \leftarrow 0$
Output: $\widehat{Q}(s,a)$
 1: while $(n < T)$ do
 2:
      $a^{(n+1)} = \begin{cases} \arg\max_{a \in \mathcal{A}} Q^n_{\Xi}(s^n, a) & w.p. \quad 1 - \varepsilon^n \\ random \quad a \in A & w.p. \quad \varepsilon^n \end{cases}$
 3:   Perform action $a^{(n+1)}$, obtain $R(n)$ and $s'$
 4:   Save the tuple $(s, a, R(n), s')$ in the replay as the samples of
      the accumulated experience
 5:   if (Q function convergence = false) then
 6:     Update Q function according to Eq. (27)
 7:   else
 8:     $\widehat{\omega}^n \leftarrow 1$; $\mathcal{Q}(s,a) \triangleq [Q_{\Xi}(s,a)]$
 9:     $\widehat{A} \leftarrow \nvdash$        // $U \times U$ matrix
10:     $\widehat{b} \leftarrow \nvdash$        // $U \times 1$ vector
11:     Calculate total $R$ by Eqs. (18), (20) and (21)
12:     for all (sample $(s, a, R(n), s')$ in the replay) do
13:       Calculate $\widehat{\omega}^n$ by Eqs. (24)–(27)
14:       $\widehat{A} \leftarrow \widehat{A} + \mathcal{Q}(s,a)(\mathcal{Q}(s,a) - \beta \mathcal{Q}(s', a'))$
15:       $\widehat{b} \leftarrow \widehat{b} + \mathcal{Q}(s,a) R(s,a)$
16:       $\widehat{\omega}^{(n+1)} \leftarrow \widehat{A}^{-1}\widehat{b}$
17:       if $(\|\widehat{\omega}^{(n+1)} - \widehat{\omega}^n\| < \rho)$ then
18:         Calculate $\{\mathbb{G}, \mathbb{E}\}$ via Eqs. (36) and (37)
19:         Calculate $Q_{\Xi}$ via Eq. (32)
20:         Return $\widehat{Q}(s,a) \leftarrow \widehat{\omega}^{(n+1)} Q_{\Xi}(s,a)$
21:       else
22:         Update $\delta^n$ and $\varrho^n$ by Eqs. (24) and (25)
23:         Update $Q_{\Xi}$ according to Eq. (38)
24:       end if
25:     end for
26:   end if
27:   $n++$
28: end while
```

To exhibit how the SDVTS works, We devise a distributed algorithm in coalition between user and cell for the seamless connection of the ARLS components and engines as depicted as Algorithm 3 where we observe that the UEs should make response to MBS according to their energy and the priority of the traffic-processing urgency. The value of the priority $l(n)$ is high, medium or low, which can be yielded on the user's side. Notice that the users' energy or priority has been received by MBS. The MBS enables to do next action according to algorithms. It may do action according to the greedy algorithm [42] to make alternative suggestions if the users are sending their energy or the priority. In this case, it is an open problem and we will do further research in our future work.

**Algorithm 3** Distributed Algorithm in Coalition between User and Cell.

```
Step (1): On the cell's side
Control engines:
1: Send commands of priority to data engines via the received traffic commands and user state
2: Call Algorithm 1
3: Combined with its current state, cell sends commands of power allocation to data engines
4: Call algorithm 2
5: Combined with its current state, cell sends commands of resource allocation to data engines
Data engines:
1: Perform commands of power allocation received by control engines
2: Perform commands of resource provisioning by control engines
3: Calculate $G(y^n_{u,b}, x_{u,m}, p_{u,b}, k_u)$ by Eq. (4)
4: Return pushed service to user
Step (2): On the user's side
1: $n \leftarrow 0$
2: while $(n < T)$ do
3:   if $((E^{\max} - e_{loss}) \geq q(n)$ or l(n) is high) then
4:     if (receiving push-based service) then
5:       Send the corresponding resource request
6:       Update energy
7:     else
8:       Actively send requested service if needed
9:       Update energy
10:    end if
11:  else
12:    Send its information of Q function to MBS
13:  end if
14:  $n++$
15: end while
```

## 6. Performance evaluation

Based on the real-world settings, we make an in-depth analysis about the performance of our approaches via extensive simulations in this section.

### 6.1. Experiment settings

The main parameters in our simulation are listed in Table 3. Other parameters are set in accordance with the 3rd Generation Partnership Project (3GPP) specifications [43].

Since the iterations vary from the network scales in our experiments as follows, the convergence threshold is dynamic in Algorithm 1 in Section 4.1. We need to adjust the iterative threshold $\Lambda$ via the empirical value such as 300. The value was derived from the Fig. 4 by extensive simulation, which is the empirical value. However, the different network scale will have influence on it. From Fig. 4, we observe that the oscillation of the curve occurs with the increase of network size. However, the amplitude of oscillation becomes smaller than before and is approaching to 300. We take the average of iterative threshold $\Lambda$ and yield the empirical value.

**Table 3**
Parameters values in experiments.

| Parameters | Values |
|---|---|
| $W$ | 20 MHz |
| $D_u$ | 100 Mcycles |
| $Z_u$ | 1 Mbits |
| $T$ | $10^{-2}$ s |
| $\beta$ | 0.9 |
| $p_m^{\max}$ | 45 dBm |
| $\vartheta^2$ | $-100$ dBm |
| $F_{u,c}(n)$ | [4,6,8,10,12] GHz [37] |
| $\varepsilon^n$ | 0.05 |

There are four algorithms for comparison in our simulation: 1) SDVTS: Our proposed algorithms are introduced by SDI and linear learning approximator with low complexity so as to allocate resource intelligently; 2) ACL: It is the abbreviation of the actor-critic learning algorithm in [20], which adopts the linear appximator without taking the curse of dimensionality into account; 3) TOC: The algorithm [8] leverages the cloud for offloading the traffic without the aid of the learning approaches; 4) SGT: The approach utilizes the Stackelberg game theory for the resource provisioning proposed in [15–18] with no assistance of the learning algorithm.

### 6.2. Performance metrics

To evaluate the performance of the novel framework SDVTS we proposed, We employ the following metrics.

- Delay of dealing with traffic: The MBS processes the requests users sent. The delay produced by cell has a non-trivial influence on the quality of experience. From the UEs' perspective, we expect the traffic-handling latency as low as possible.
- Average value of *SINR* for users: We tend to demonstrate the effect of our learning scheme by exhibiting the values of *SINR* which acts as another metric of the QoE. We desire to receive the high *SINR* in order to have the high quality of the received signal.
- Throughput rate of system: It is a crucial criterion for the MBS to measure whether it is efficient for the resource assignment with different algorithms or not. The lower throughput rate suggests the higher delay of coping with the traffic, implying that the scheme of the resource provisioning is not optimal.
- Number of iterations: It is a significant measurement to weigh the efficiency of algorithms. As we all know, the fewer iterations are, the better the algorithm is.
- Convergence time: It is a quantified metric for the efficiency of certain algorithm. It is various in different scenes. It also weighs the feasibility of algorithm.

### 6.3. QoE comparison

As shown in Fig. 5(a), it shows the impact of the traffic-processing delay on the network size with different schemes. The trend of SGT almost presents prominently increasing, suggesting

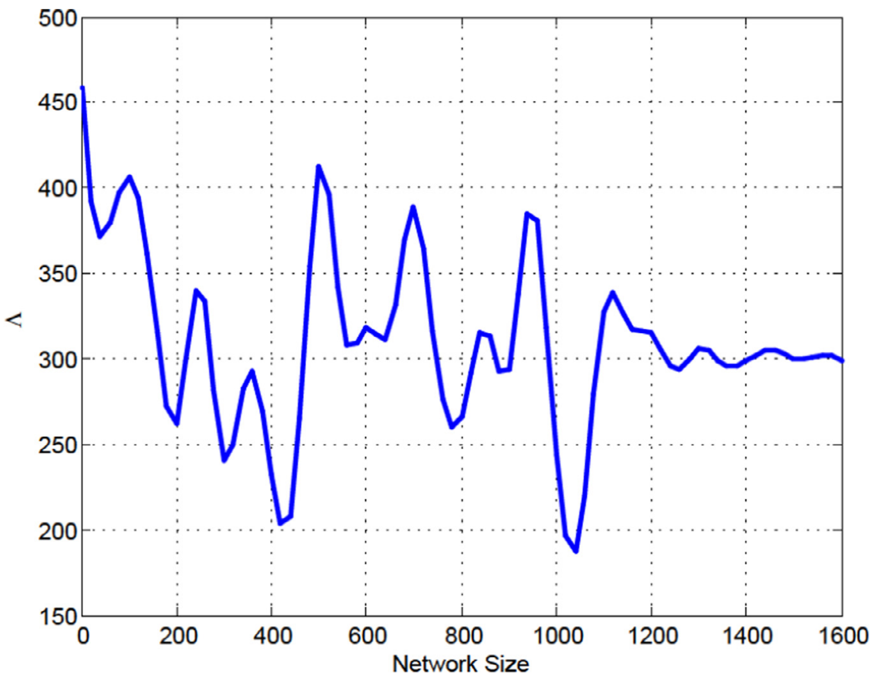


**Fig. 4.** Impact of Λ on the network size.

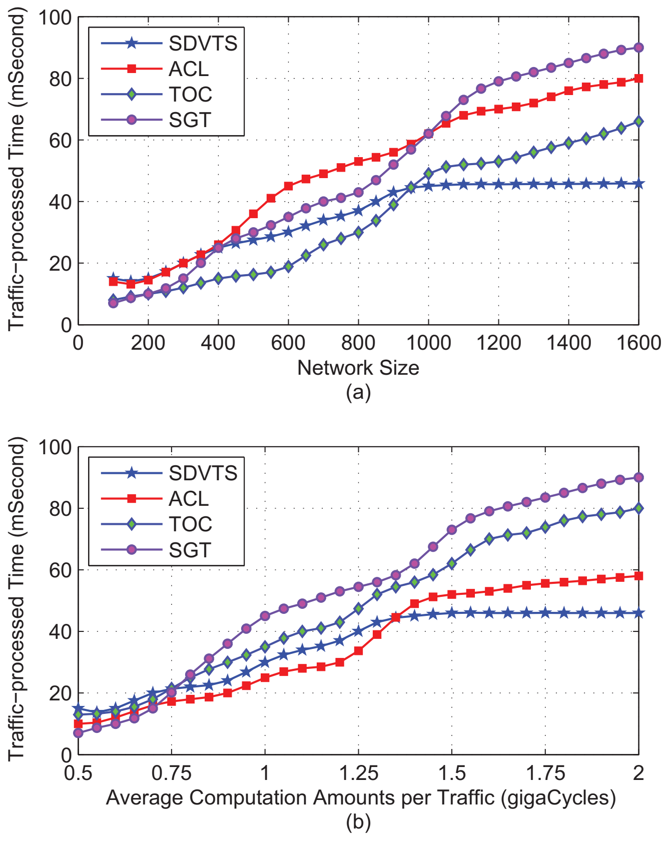


**Fig. 5.** Impact of the traffic-processing delay on the network size and task amounts.

that the delay for dealing with traffic increases with the expanding of network size. While the curves of ACL and SDVTS rise slowly, implying the learning algorithms, to some extent, take effect on the latency especially for the ultra-dense networks. The curve of ACL is still ascending with the increase of network scale because the demands is diverse and the space for storing states and actions is limited. While the trend of curve with SDVTS is almost stop ascending, achieving the ultra-low latency. The reason is that the SDVTS leverages the software defined engines which separate the data and control signaling and the double approximators of the learning algorithms with low complexity. In contrast to the ACL, the traffic-processing delay of the SDVTS approach decreases about 43%. That is, our proposed framework for processing traffic outperforms ACL in terms of latency, which enhances the QoE. Nevertheless, note that the delay of TOC aided by clouds without the learning-enable algorithm is rather lower than that of ACL and SDVTS when the network size is small. It reflects that the effect of using the learning algorithm does not work well in the small networks since the learning procedure takes up time and it is easy to suffer from the problem of oscillation in the first training phases. Note that the traffic-processing time of SDVTS is becoming unchanged when the network size is around 970; while the time of TOC is almost increasing linearly.

Suppose the requested traffic is subjected to the normal distribution, we simulate the time of handling task on the cell's side with different schemes under the premise of normal distribution. As illustrated in Fig. 5(b), we notice that the traffic-processing time of SDVTS presents a non-continuous increase when it is compared with the time of the other strategies from the global perspective. However, SGT and TOC initially perform a little better than the learning method of ACL and SDVTS. While the ACL is still worse than the SDVTS with the increasing amount of traffic. Because, assisted by SDI, the SDVTS leverages the improved reinforcement learning algorithm with low complexity, fulfilling the intelligent resource assignment with the personalized QoE profiling in the long run.

It has described the impact of average value of *SINR* for users on the network size with different approaches in Fig. 6(a). We conclude that the SDVTS achieves the highest SINR in the other approaches. Because its ARLS components works well, which employs the double approximators of the learning algorithm with the tailored QoE profiling. Besides, note that the *SINR* with SDVTS strategy, overall, is higher than the threshold of *SINR* despite of

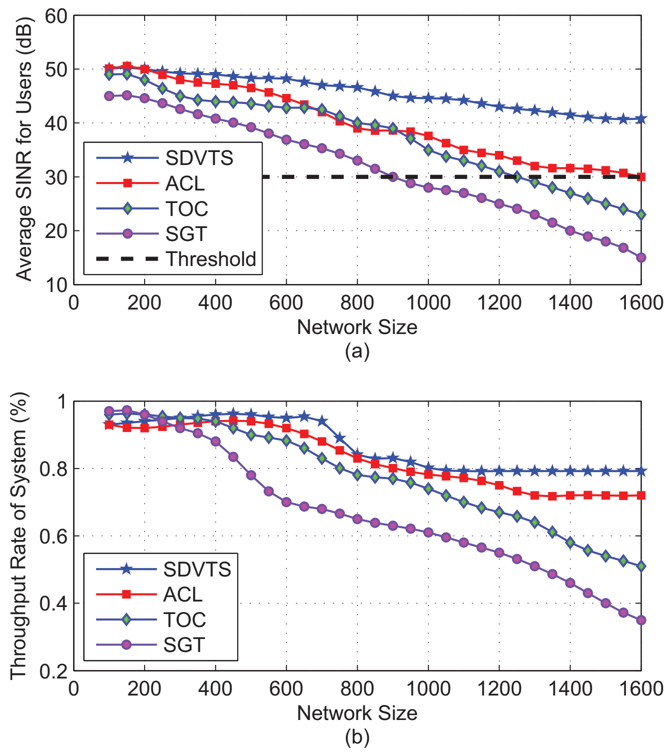


**Fig. 6.** Comparison of *SINR* and throughput rate with the network size for different strategies.

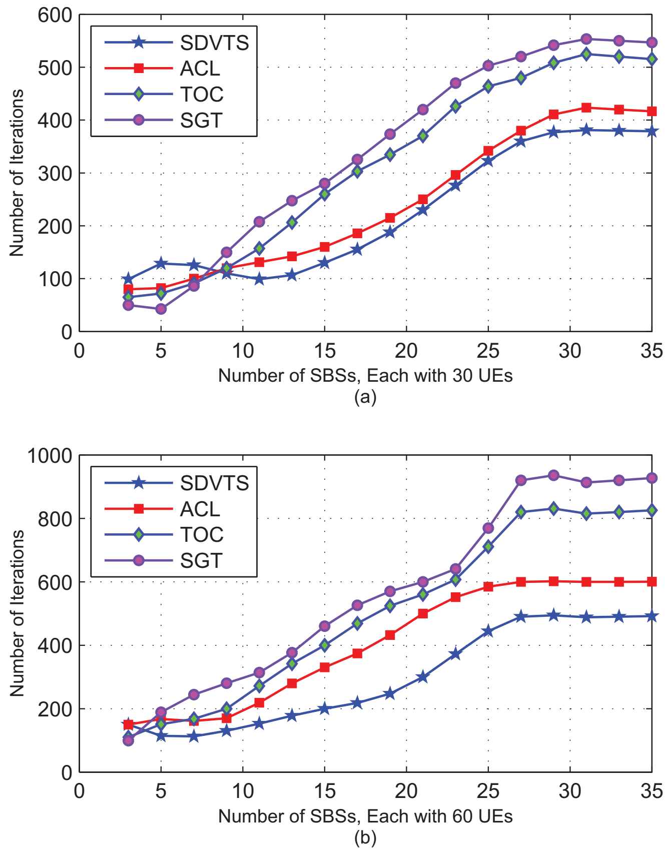


**Fig. 7.** Comparison of iterations for different SBSs with the different amounts of UEs under different schemes.

the continuous growth in the network size. Further, we notice that the QoE of the learning-enable algorithms in the aspect of *SINR* is better than that without the learning ones with the network scale increasing. In addition, the curve of *SINR* with SDVTS reflects that the dual evolution algorithm we developed achieves the optimal or near-optimal solution, since we offer the lower bound of power related to *SINR*, which is illustrated in Section 4.1.

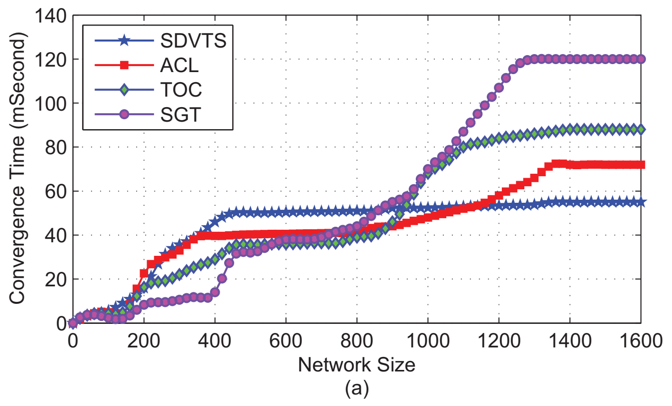
While, as depicted as Fig. 6(b), it is observed that the throughput rate of the MBS system with the learning algorithm such as SDVTS and ACL is lower than that without the learning ones of TOC and SGT at the initial stage. Because the initial learning algorithm is easily encountered with the oscillation and instability without the sufficient samples. However, their throughput rate becomes higher than that of TOC and SGT as the increase of network size, suggesting that SDVTS and ACL is more adaptive to the larger networks that cater to the characteristics of next generation wireless networks. While our proposed SDVTS dominates in terms of the cell's throughput rate when in comparison with the ACL as it is the low-complexity learning algorithm in provisioning the efficient resources.

## 6.4. *Performance verification*

The comparison of iterations for different SBSs with the different amounts of UEs under different schemes is revealed in Fig. 7(a) and (b). We observe that the iterations increase because of the growing SBSs. Whereas ACL and SDVTS are faster than TOC and SGT in the convergence time, showing that leveraging the learning-enable algorithm takes effect. Note that the gap between Fig. 7(a) and (b) is rather distinct when the range of the number of SBSs is from 15 to 28, and the iterations of SDVTS are the least among the compared schemes. It is concluded that the SDVTS, which is assisted by SDI and the double approximators of reinforcement learning with low complexity, is adaptable to the large networks.

The comparison of convergence time with different approaches is depicted as shown in Fig. 8(a). We observe that the convergence time of SDVTS is the shortest among the other schemes despite of increasing the network scale. Unlike the ACL and SGA approaches, the SDVTS adopts the double approximators with the online-offline learning to approximate the optimal policy via the QoE profiling. While compared with the SGA and TOC approaches without the learning algorithm, SDVTS and ACL converge faster than them, suggesting the learning-enable algorithm can make the system smart to adapt to the dynamic environment in the ultra-dense HetNets. Note that almost all curves except for SDVTS in Fig. 8(a) have undergone two stable changes in convergence time with different network size, which means they is prone to be unstable. Compared with them, SDVTS we proposed has advantages over stability.

In order to evidence that the elaborate framework, SDVTS, has the generalization ability, the system performance in different scenarios with different schemes is illustrated in Fig. 8(b). It is not easy to quantize the performance for different systems because of the various demands in different applications. We carry out extensive simulations on different scenarios with vehicle networks, cellular networks, drones networks and even with the HetNets (hybrid networks) respectively. Herein, we adopt the same MBS under the different networks to measure the performance of it. In MBS, we exploit a 2-layer fully-connected feedforward neural network which has 64 and 32 neurons in the first and second layer respectively. The Leaky Rectifier [44] is acted as the activation function before the final output layer which utilizes the softmax for activation. The activation function is to ensure the sum of the output values equals one. That is why the range of y-axis of Fig. 8 (b) belongs to [0, 1]. The unit of y-axis is the percentage of the performance of MBS. According to our results as shown in



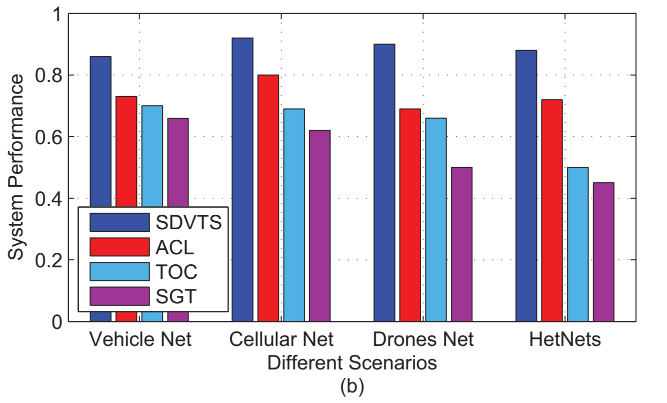


**Fig. 8.** Comparison of convergence time and system performance with different approaches.

Figs. 6(a) and 8(b), we observe that, contrary to ACL and SDVTS with the learning algorithm, the system performance of SGA and TOC may strap into the unstable state in the growing network. Compared with ACL, we notice that the curve of SDVTS changes less than that of ACL, suggesting that SDVTS approach has a better compatibility with the different network scenes. Since the problem of the curse of dimensionality is not taken into consideration, the system performance of ACL is lower than that of SDVTS in HetNets. Further, we observe that the ACL suffers from oscillation when the network scale is enlarging continuously. For example, the size is set to about 1300. Since the ACL is the pure online learning algorithm, which is different from the SDVTS that involves the online-offline learning mechanism. Overall, our proposed SDVTS exhibits the high stability and scalability.

## 7. Conclusion

The proliferation of mobile devices and data-intensive applications is driving up a significant demand for services with the high QoE. In this paper, assisted by SDI, we first propose a traffic-processing framework SDVTS, which boosts the efficiency of resource utilization and in return offers better QoE for users in an interactive loop. Further, we construct the traffic-processing time model and decompose the non-convex optimization intuitively. Subsequently, we illustrate the dual evolution algorithm for the power provisioning to solve one of the sub-problems. Moreover, we introduce a double approximators of the Q-learning with credit using the eligibility traces, addressing the problem of the delayed cost (or delayed reward) which occurs in the reinforcement learning approaches. Besides, a factored approach with the tailored QoE profiling is developed to tackle the problem of the curse of dimensionality, expediting the convergence of algorithm. Additionally, a distributed algorithm in coalition between user and cell is designed for illustrating how our framework SDVTS works. Finally, compared with the regular counterparts, our approach outperforms the other schemes in terms of the QoE and the cell's system performance via the massive simulations.

## Declaration of competing interest

We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted "When QoE Meets Learning: A Distributed Traffic-Processing Framework for Elastic Resource Provisioning in HetNets" (ID:COMNET_2018_873).

## Appendix A

Proof of lemman 1.

**Proof.** $G_3(K)$ is a group of functions with respect to $k_u$. Since the given premise that $y^n_{u,b} = 1$, $x_{u,m} = 0$, there are $B$ elements of vector $K$. $K$ is denoted as $\{k_{u,1}, k_{u,2}, \ldots, k_{u,B}\}$. Then we can acquire the Hessian matrix of functions $G_3(k_u)$ as follows:

$$I = \begin{bmatrix} \frac{\partial^2 G_3}{\partial^2 k_{u,1}} & \frac{\partial^2 G_3}{\partial k_{u,1}\partial k_{u,2}} & \cdots & \frac{\partial^2 G_3}{\partial k_{u,1}\partial k_{u,B}} \\ \frac{\partial^2 G_3}{\partial k_{u,2}\partial k_{u,1}} & \frac{\partial^2 G_3}{\partial^2 k_{u,2}} & \cdots & \frac{\partial^2 G_3}{\partial k_{u,2}\partial k_{u,B}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial^2 G_3}{\partial k_{u,B}\partial k_{u,1}} & \frac{\partial^2 G_3}{\partial k_{u,B}\partial k_{u,2}} & \cdots & \frac{\partial^2 G_3}{\partial^2 k_{u,B}} \end{bmatrix}$$

where each specific element is calculated as

$$\frac{\partial^2 G_3}{\partial k_{u,\imath}\partial k_{u,\jmath}} = \begin{cases} \frac{2\aleph_u}{(k_u)^3 f_u}, & If\ \imath = \jmath, \imath \in B, \jmath \in B \\ 0, & Otherwise \end{cases} \tag{A.1}$$

It is obvious that $\frac{\partial^2 G_3}{\partial k_{u,\imath}\partial k_{u,\jmath}} \geq 0$ because of all positive parameters in Eqn. (A.1). Hence, all eigenvalues of Hessian matrix $I$ are positive numbers and matrix $I$ is a symmetrical positive definite matrix. It is concluded that the function $G_3(k_u)$ is convex according to Theorems in [45]. □

## References

[1] L. Duan, L. Huang, C. Langbort, A. Pozdnukhov, J. Walrand, L. Zhang, Human-in-the-loop mobile networks: a survey of recent advancements, IEEE J. Sel. Areas Commun. 35 (4) (2017) 813–831.
[2] S. Cai, Y. Che, L. Duan, J. Wang, S. Zhou, R. Zhang, Green 5G heterogeneous networks through dynamic small-cell operation, IEEE J. Sel. Areas Commun. 34 (5) (2016) 1103–1115.
[3] Z.A. Qazi, M. Walls, A. Panda, V. Sekar, S. Ratnasamy, S. Shenker, A high performance packet core for next generation cellular networks, in: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, SIGCOMM, 2017, pp. 348–361.
[4] M. Chen, Y. Qian, Y. Hao, Y. Li, J. Song, Data-driven computing and caching in 5G networks: architecture and delay analysis, IEEE Wireless Commun. 25 (1) (2018) 70–75.
[5] H. Zhang, Y. Nie, J. Cheng, V.C.M. Leung, A. Nallanathan, Sensing time optimization and power control for energy efficient cognitive small cell with imperfect hybrid spectrum sensing, IEEE Trans. Wireless Commun. 16 (2) (2017) 730–743.
[6] H. Yu, M.J. Neely, Learning aided optimization for energy harvesting devices with outdated state information, in: Processings of International Conference on Computer Communications, INFOCOM, 2018.
[7] F. Rebecchi, M.D. de Amorim, V. Conan, A. Passarella, R. Bruno, M. Conti, Data offloading techniques in cellular networks: a survey, IEEE Commun. Surv. Tut. 17 (2) (2015) 580–603.
[8] M. Chen, Y. Hao, Task offloading for mobile edge computing in software defined ultra-dense network, IEEE J. Sel. Areas Commun. 36 (3) (2018) 587–597.
[9] H. Yang, J. Zhang, Y. Ji, Y. Lee, C-RoFn: multi-stratum resources optimization for cloud-based radio over optical fiber networks, IEEE Commun. Mag. 54 (8) (2016) 118–125.

[10] H. Yang, A. Yu, X. Zhao, Q. Yao, J. Zhang, Y. Lee, Multi-dimensional resources allocation based on reconfigurable radio-wavelength selective switch in cloud radio over fiber networks, Opt. Expr. 26 (26) (2018) 34719–34733.
[11] P. Mach, Z. Becvar, Mobile edge computing: a survey on architecture and computation offloading, IEEE Commun. Surv. Tut. 19 (3) (2017) 1628–1656.
[12] H. Wu, T. Wang, Z. Yuan, C. Peng, Z.L. et al., The tick programmable low-latency SDR system, in: Processings of International Conference on Mobile Computing & Networking, MOBICOM, 2017, pp. 101–114.
[13] D. Silver, J. Schrittwieser, K. Simonyan, I.A. et al., Mastering the game of go without human knowledge, Nature 550 (7676) (2017) 354–360.
[14] A. Syed, J. Van der Merwe, Proteus: a network service control platform for service evolution in a mobile software defined infrastructure, in: Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking, MOBICOM, 2016, pp. 257–270.
[15] H. Zhang, Y. Zhang, Y. Gu, D. Niyato, Z. Han, A hierarchical game framework for resource management in fog computing, IEEE Commun. Mag. 55 (8) (2017) 52–57.
[16] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F.R. Yu, Z. Han, Computing resource allocation in three-tier IoT fog networks: a joint optimization approach combining stackelberg game and matching, IEEE Internet Things J. 4 (5) (2017) 1204–1215.
[17] T.Z. Oo, N.H. Tran, W. Saad, D. Niyato, Z. Han, C.S. Hong, Offloading in hetnet: a coordination of interference mitigation, user association, and resource allocation, IEEE Trans. Mob. Comput. 16 (8) (2017) 2276–2291.
[18] Z. Zheng, L. Song, D. Niyato, Z. Han, Resource allocation in wireless powered relay networks: a bargaining game approach, IEEE Trans. Veh. Technol. 66 (7) (2017) 6310–6323.
[19] I. AlQerm, B. Shihada, Energy-efficient power allocation in multitier 5G networks using enhanced online learning, IEEE Trans. Veh. Technol. 66 (12) (2017) 11086–11097.
[20] Y. Wei, F.R. Yu, M. Song, Z. Han, User scheduling and resource allocation in hetnets with hybrid energy supply: an actor-critic reinforcement learning approach, IEEE Trans. Wireless Commun. 17 (1) (2018) 680–692.
[21] L. Zheng, D.W.H. Cai, C.W. Tan, Max-min fairness rate control in wireless networks: optimality and algorithms by perron-frobenius theory, IEEE Trans. Mob. Comput. 17 (1) (2018) 127–140.
[22] S. DOro, L. Galluccio, S. Palazzo, G. Schembra, A game theoretic approach for distributed resource allocation and orchestration of softwarized networks, IEEE J. Sel. Areas Commun. 35 (3) (2017) 721–735.
[23] J. Zhao, Y. Liu, K.K. Chai, A. Nallanathan, Y. Chen, Z. Han, Spectrum allocation and power control for non-orthogonal multiple access in hetnets, IEEE Trans. Wireless Commun. 16 (9) (2017) 5825–5837.
[24] T.Z. Oo, N.H. Tran, W. Saad, D. Niyato, Z. Han, C.S. Hong, Offloading in hetnet: a coordination of interference mitigation, user association, and resource allocation, IEEE Trans. Mob. Comput. 16 (8) (2017) 2276–2291.
[25] B. Xu, Y. Chen, J.R. Carrin, T. Zhang, Resource allocation in energy-cooperation enabled two-tier NOMA HetNets toward green 5G, IEEE J. Sel. Areas Commun. 35 (12) (2017) 2758–2770.
[26] J. Wang, C. Jiang, Z. Han, Y. Ren, R.G. Maunder, L. Hanzo, Taking drones to the next level: cooperative distributed unmanned-aerial-vehicular networks for small and mini drones, IEEE Veh. Technol. Mag. 12 (3) (2017) 73–82.
[27] Y. Bao, H. Wu, X. Liu, From prediction to action: improving user experience with data-driven resource allocation, IEEE J. Sel. Areas Commun. 35 (5) (2017) 1062–1075.
[28] Y. He, N. Zhao, H. Yin, Integrated networking, caching, and computing for connected vehicles: a deep reinforcement learning approach, IEEE Trans. Veh. Technol. 67 (1) (2018) 44–55.
[29] H.V. Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning artificial intelligence, in: Proceedings of the National Conference on Artificial Intelligence, AAAI, 2016, pp. 2094–2101.
[30] Z. Wang, N. de Freitas, M. Lanctot, Dueling network architectures for deep reinforcement learning, 2015, arXiv:1511.06581.
[31] X. Zhang, L. Duan, Optimal deployment of UAV networks for delivering emergency wireless coverage, 2017. arXiv:1710.05616v1.
[32] L. Yu, Z. Wang, X. Nan, B. Zhou, DMGR: a multipath geographic routing strategy with the on-demand mobile sink in WSN, Ad Hoc Sensor Wireless Netw. 35 (2) (2017) 1–24.
[33] R.J.W. Steven I. Rich, C. Majidi, Untethered soft robotics, Nature Electron. (2018) 102–112.
[34] X. Huang, X. Yi, W. Tang, S. Hu, M. Zhu, J. Zhang, B. Xu, K. Qiu, Closed-form solutions for nonlinear shannon limit due to kerr effect in optical fibre transmissions with digital backpropagation, Opt. Commun. 436 (2019) 243–247.
[35] P. Mach, Z. Becvar, Mobile edge computing: A Survey on architecture and computation offloading, IEEE Commun. Surv. Tut. 19 (3) (2017) 1628–1656.
[36] J.W. Lee, R.R. Mazumdar, N.B. Shroff, Non-convex optimization and rate control for multi-class services in the internet, IEEE/ACM Trans. Netw. 13 (4) (2005) 827–840.
[37] Soteit, SaharSecci, Stefano, Game theory in wireless and communication networks: theory, models, and applications, IEEE Commun. Mag. 50 (10) (2012) 20–30.
[38] Z. Xu, J. Tang, J. Meng, e. a. Weiyi Zhang, Experience-driven networking. a deep reinforcement learning based approach, in: Processings of International Conference on Computer Communications, INFOCOM, 2018.
[39] E. Alpaydin, Reinforcement Learning, MIT Press, 2014.
[40] D. Altinel, G.K. Kurt, Finite-state markov channel based modeling of rf energy harvesting systems, IEEE Trans. Veh. Technol. 67 (2) (2018) 1713–1725.
[41] L. A., P. et al., A constrained optimization perspective on actorccritic algorithms and application to network routing, Syst. Control Lett. 92 (2016) 46–52.
[42] C. Jiang, Z. Chen, R. Su, Y.C. Soh, Group greedy method for sensor placement, IEEE Trans. Signal Process. 67 (9) (2019) 2249–2262.
[43] 3GPP, Overview of 3GPP, 2014, Release 12. V0.1.4, 3GPP, Sophia Antipolis Cedex, France.
[44] Y.B. I. Goodfellow, A. Courville, Deep Learning, MIT Press, 2016. http://www.deeplearningbook.org/.
[45] T. Chen, Q. Ling, G.B. Giannakis, An online convex optimization approach to proactive network resource allocation, IEEE Transactions on Signal Processing 65 (24) (2017) 6350–6364 .

**Li Yu** received her M.S. degree in Computer Applications Technology from Zhengzhou University, China in 2016 and she is currently a Ph.D. candidate in Computer System Architecture at Wuhan University. Her research includes reinforcement learning, resource management, Internet of Things, mobile cloud computing and software defined network.

**Zongpeng Li** received his BE in Computer Science from Tsinghua University in 1999, and his PhD from University of Toronto in 2005. He has been affiliated with University of Calgary and then Wuhan University. His research interests are in computer networks, network algorithms, and cloud computing. Zongpeng received the Outstanding Young Computer Scientist Prize from the Canadian Association of Computer Science, and a few Best Paper Awards from conferences in related fields.

**Yucun Zhong** will received the B.S. degree from WuHan University. His major is computer science. He is interested in next generation wireless networks and machine learning. He has further done exploration in this direction together with seniors and researchers in the laboratory.

**Zhenzhou Ji** received Ph.D. in Computer Science in 2000 from Harbin Institute of Technology. He is now a professor in the Harbin Institute of Technology. His research interests include computer architecture, parallel processing computer and computer network security. He is the Vice Chairman of Computer Architecture of China Computer Federation (CCF).

**Jiangchuan Liu** (F'17) received the B.Eng. degree (cum laude) in computer science from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree in computer science from The Hong Kong University of Science and Technology, in 2003. He is currently a University Professor with the School of Computing Science, Simon Fraser University, BC, Canada. He is the Steering Committee Chair of the IEEE/ACM IWQoS, from 2015 to 2017, and the TPC Co-Chair of the IEEE IC2E 2017 and the IEEE/ACM IWQoS 2014. He serves as an Area Chair for the IEEE INFOCOM, ACM Multimedia, and the IEEE ICME. He has served on the Editorial Boards of the IEEE Transactions on Big Data, the IEEE Transactions on Multimedia, the IEEE Communications Surveys and Tutorials, the IEEE Access, the IEEE Internet of Things Journal, Computer Communications, and Wireless Communications and Mobile Computing (Wiley). (Based on document published on 9 December 2018).