Energy Efficient Task Offloading for Compute-intensive Mobile Edge Applications

Xiaojie Zhang, Saptarshi Debroy
City University of New York
Emails: *xzhang6@gradcenter.cuny.edu, saptarshi.debroy@hunter.cuny.edu*

***Abstract*—In mobile edge computing (MEC) systems, offloading real-time and compute-intensive application tasks to remote edge servers is performed to relieve energy-constrained mobile devices of energy consuming computations. However, such practice often becomes counter-productive as transmission power requirements to offload such real-time tasks through wireless can make the mobile devices spend significant energy. In this paper, we propose an energy-efficient task offloading scheme for real-time and compute-intensive applications that optimizes energy consumption at mobile devices without violating such applications' strict latency requirements. In particular, for local energy savings at the mobile devices, we propose a Computation and Power Optimization (CPO) algorithm for optimal job partitioning. Then we propose a multi-device and multi-server task Joint Task Offloading Game (JTOG) algorithm in order to minimize the energy consumption for all mobile devices generating multiple tasks. Finally, using a realistic and detailed simulation, we prove that a tractable Nash Equilibrium always exists for the game that optimizes the energy savings of all mobile devices. We also show that the proposed JTOG algorithm performs significantly better than other default full task offloading schemes in terms of overall energy savings.**

***Index Terms*—Energy efficiency, task offloading, mobile edge systems, real-time applications, directed acyclic graphs, Nash equilibrium.**

I. Introduction

With real-time mobile applications becoming increasingly compute-intensive for many mission-critical use cases, the energy capacity of embedded mobile end-devices are proving to be insufficient to handle all in-device computation. Mobile edge computing (MEC) [1], [2] allows mobile devices to offload some or all of such real-time and compute-intensive tasks to edge servers as it brings cloud-scale compute resources closer to the mobile devices. Thus intuitively, in order to preserve the limited energy of mobile devices, all computing tasks should be offloaded to edge servers.

In many real-time application use-cases [3], such offloading often uses wireless transmission for data transfer between mobile devices and edge servers. However, inherent fluctuations in wireless channel quality caused by phenomenons such as, interference, path loss, shadowing, and fading result in varying end-to-end data rate. This in turn adds to the transmission cost of task offloading to edge servers that eventually increases the overall energy expenditure of mobile devices. Thus in energy-aware edge systems, data transmission costs of task offloading can often outweigh the energy preservation benefits of remote computation, which entirely defeats the purpose of task offloading. Therefore, an alternative approach of partial task offloading can be beneficial that seeks to reduce the size of input data by running certain lightweight pre-processing jobs at the mobile devices. The intermediate data can then be offloaded to edge servers for further processing, thereby reducing the transmission cost at mobile devices.

In this paper, we model such partial task offloading problem in mobile edge system under varying task Directed Acyclic Graph (DAG) partition models and study the in-device energy efficiency problem. In particular, we aim to find a task offload strategy that makes partial task offload decisions with edge server selection, as well as computes transmission power and computation speed configurations to satisfy real-time nature of the applications. For local energy savings at mobile devices, we propose a Computation and Power Optimization (CPO) algorithm for optimal job partitioning. The algorithm selects the best transmission power and computation speed that satisfy the task latency requirements. At the same time, in order to achieve a Nash equilibrium (NE) in edge server selection problem, we prove that the multi-device and multi-server task offloading game is a potential game and propose a Joint Task Offloading Game (JTOG) algorithm for strategy updating. Using a realistic and detailed simulation, we show how the JTOG algorithm optimizes the overall energy consumption and always leads to a NE within a reasonable number of iterations. We also show that the JTOG algorithm intelligently selects the optimal DAG partition model that is superior in terms of overall energy savings to other default strategies such as, greedy fully remote tasks offloading scheme and fully local task computation scheme.

The rest of the paper is organized as follows. Section II presents the related work. Section III discusses the system model. Section IV proposes the optimization problem of energy efficient task computation model. Section V presents the joint task offloading game. Section VI discusses simulation results. Section VII concludes the paper.

II. Related Work

Mobile edge systems provide the option to offload some or all of the computationally intensive application tasks to edge computing servers, often using last-mile wireless transmission. This approach releases the limited computing resources at hand-held mobile devices for other small applications and at the same time reduces excess energy consumption at the devices. However, due to the complexity of the wireless transmission, the heterogeneity of computation-intensive application tasks in terms of their resource and energy requirements and real-time nature, edge server selection and subsequent resource allocation remain challenging research problems.

The multi-device mobile edge systems have been extensively studied in [1], [2], [4], [5]. In latency-aware platform LAVEA [1], authors use DAGs to represent the task execution sequence where job dependency has been considered. This work aims to find the optimal task graph partition with the purpose of minimizing end-to-end latency. An energy-efficient offloading framework has been studied in [2] where the authors

applied a Sporadic Task model and each task is divided into local-only stages and offloadable stages. Compared to these works, authors in methods, such as [4], [5] proposed partial task offloading schemes with a concurrent execution model without job dependency. Authors in [4] studied joint computation and resource allocation policy under time-division and frequency-division settings driven by a offloading priority function. Authors in [5] investigated the user cooperation in mobile edge systems with helper nodes which work as decode-and-forward (DF) relay for cooperative communication.

In [8], [9], [10], the multi-device and multi-server edge systems have been studied. Authors in [8] proposed solutions to resource allocation and task placement problems with software-defined ultra dense network (SD-UDN) architecture. In [9], the energy efficiency problem in edge systems is decoupled into two: optimal power and sub-carrier allocation, and optimal task offloading strategy. The authors use Hungarian method to find the optimal offloading solution. Compared to [8] and [9], authors in [10] consider energy consumption of edge servers in terms of communication and computation power. However, none of the aforementioned works have considered partial offloading decision making and energy-efficient resource allocation problems together in multi-device and multi-server edge systems.

III. System Model

In this section, we will describe the mobile edge system model we used for this work. We define a mobile edge system containing $\mathcal{N} = \{1, 2, ..., N\}$ mobile devices (users) and $\mathcal{K} = \{1, 2, ..., K\}$ edge servers, both with heterogeneous computation capacities. We describe the real-time application/-task offloading decision by mobile device n as $a_n \in \mathcal{A} \triangleq \{0, 1, 2, ..., K\}$. More specifically, we define

$$a_n = \begin{cases} 0 & \text{if user } n \text{ executes task locally} \\ k & \text{if user } n \text{ executes task on edge server k} \end{cases}$$

In this paper, we define $I_{\{a_n=k\}} = 1$ as the indicator that the event $\{a_n = k\}$ is true; otherwise $I_{\{a_n=k\}} = 0$. Fig. 1 shows the overall mobile edge system model where devices offload all or some of the task jobs to edge servers using wireless transmission.

A. Application Model

In this paper, we study real-time compute-intensive mobile applications such as, video processing based on running machine learning algorithms on raw video data. More specifically, mobile devices will continuously send video frames to edge servers for face recognition or object detection with strict deadline requirements. As shown in Fig. 1 and Fig. 2, we model the applications as Sporadic Task [6] which has a Directed Acyclic Graph (DAG) $G_n = (V_n, E_n)$. The DAG contains a group of jobs that are executed sequentially. Each job has a computation requirement denoted by $\omega_{n,x}$ (i.e., the number of CPU cycles) and will contain multiple stages with intermediate data (e.g., face location and size, and face features) as shown in Fig. 2. As part of modeling their mission criticality, we denote the minimum release period of video frames and the deadline of DAG execution as T_n and D_n, respectively.

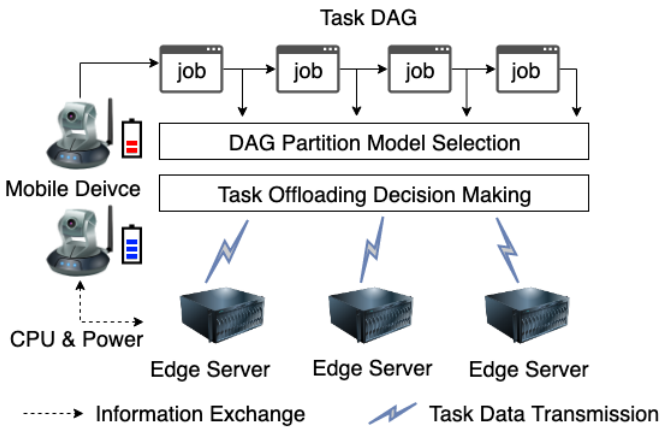

Fig. 1: System model showing mobile devices with limited compute and energy resources offloading tasks to edge servers.

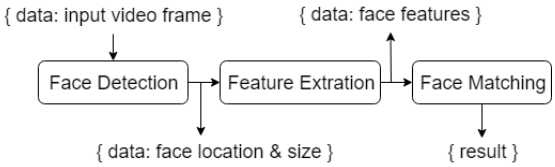

Fig. 2: The DAG example of face recognition application

B. DAG Partition Model

The different stages of DAG can be partitioned into two groups (local-processing jobs and remote-processing jobs) to achieve specific cost optimization. The remote-processing jobs cannot be executed until the intermediate data generated by local-processing jobs are received at the edge server. In addition, we allow the mobile devices to intelligently select their task DAG partition model, which is defined as,

$$(\Theta_n = m) \equiv (X_{n,m}, Y_{n,m}, Z_{n,m}), \forall m \in \{0, 1, .., M_n\} \quad (1)$$

where M_n is the length of jobs. We describe the accumulated computation requirement of jobs (i.e. CPU cycles) as $X_{n,m} = \sum_{x=m+1}^{M_n} \omega_{n,x}$ and $Y_{n,m} = \sum_{x=1}^{m} \omega_{n,x}$ at the edge server and at the mobile device, respectively. We define $Z_{n,m}$ as the size of intermediate data when $0 < m < M_n$ or the size of task input data when $m = 0$.

C. Communication Model

For our application data transmission (from mobile devices to edge servers) model, we assume that all edge servers apply the Orthogonal Frequency Division Multiple Access (OFDMA) communication [13]. We denote C_k as the set of sub-channels that can be used for data transmission at the edge server k. In this paper, we assume that the sub-channel gain simply depends on the distance between the mobile device and the edge server, which can be modeled as

$$h_{n,k} = \|d\|^{-s} \quad (2)$$

where $\|d\|$ is the distance between the mobile device and edge server and s is the path loss factor. Thus, the optimal

transmission power and the bit-rate will be equally distributed on each allocated sub-channel. The aggregated data rate for task data transmission is computed as,

$$r_n^k = \sum_{x=1}^{b_{n,k}} W_0 \log_2(1 + \frac{p_n h_{n,k}^2}{N_0}) = b_{n,k} W_0 \log_2(1 + \frac{p_n h_{n,k}^2}{N_0}) \tag{3}$$

where

$$b_{n,k} = \left\lfloor \frac{C_k}{\sum_{n=1}^{N} I_{\{a_n=k\}}} \right\rfloor$$

$b_{n,k}$ is the number of sub-channels assigned to the mobile device under fair sub-channel allocation. In Eq. (3), p_n is the transmission power of mobile device allocated on a single sub-channel (i.e. the total power is $b_{n,k}p_n$) and N_0 is the white Gaussian noise. The notations used for the system model and rest of the paper are summarized in Table I.

TABLE I: Notation used

Symbol	Definition
$\mathcal{K}$	The set of edge servers
$\mathcal{N}$	The set of mobile devices
$\mathcal{S}$	The offloading strategy set of mobile devices
p_n	The transmission power of device n on single sub-channel
f_n	The CPU speed of device n
$f_{n,k}$	The CPU speed of edge server k for device n
a_n	The offloading decision of device n
$\Theta_n(m)$	The mth DAG partition model of device task n
$\omega_{n,x}$	The computation requirement of xth job of device task n
$X_{n,m}$	The accumulated computation requirements of remote jobs
$Y_{n,m}$	The accumulated computation requirements of local jobs
$Z_{n,m}$	The size of transmitted data
C_k	The number of sub-channels available to edge server k
$b_{n,k}$	The number of sub-channels allocated to device n
$h_{n,k}$	The sub-channel gain between device n and edge server k
$r_{n,k}$	The achievable data rate between device n and edge server k
$\alpha_{n,k}$	The data transmission time of device task n
$\beta_{n,k}$	The remote computation time of device task n
D_n, T_n	The deadline and release period of device task n

IV. Energy Efficient Task Computation

In this section, we formulate and analyze task DAG partitioning problem at mobile devices for transmission power and computation speed optimization. The energy consumption and computation model follows methods described in [7] which allow the devices to adjust their computation speed for the purpose of energy saving based on Dynamic Voltage Scaling (DVS) technique.

***Remark** 1:* When a mobile device decides to execute a task locally (i.e., $a_n = 0$ and $\Theta_n = M_n$ signifying both jobs are processed at the mobile device), the minimal energy consumption $w.r.t.$ the task deadline can be stated as,

$$E_n^l = \kappa(X_{n,m} + Y_{n,m})[\frac{(X_{n,m} + Y_{n,m})}{D_n}]^2 \tag{4}$$

where κ is an energy consumption coefficient (i.e. J/cycle) [7]. It is evident that a task can be locally executed if and only if the device's maximum computation speed satisfies the constraint,

$$\frac{X_{n,m} + Y_{n,m}}{D_n} \leq f_n^{max} \tag{5}$$

Otherwise, the jobs must be offloaded to a edge server and processed remotely.

A. Adaptive Task Computation

In our model, when $I_{\{a_n=k\}} = 1$, mobile device n decides to offload part or all of the jobs to the edge server k. We first define the concept of beneficial task offloading decision in terms of energy consumption.

***Definition** 1:* A beneficial task offloading (to edge server) decision should in result less energy consumption compared to executing all jobs at the mobile device, i.e., the sum of energy consumed for local computation of part of the task and energy consumed for transmission of the rest of the task should be less than E_n^l.

To minimize the energy consumption, the device needs to carefully choose DAG partition model Θ_n and select the optimal resource configuration: the transmission power p_n (in Watts) and the local computation speed f_n (in CPU cycles/s). We create two auxiliary variables $\alpha_{n,k}$ and $\beta_{n,k}$ as the transmission time and the edge server computation time for a task. By definition, we have $Z_{n,m} \overset{def}{=} \alpha_{n,k} r_n^k$ and $X_{n,m} \overset{def}{=} f_{n,k}\beta_{n,k}$ where $f_{n,k}$ is the edge server computation speed allocated to the device task. Base on Eq. (3), the transmission power allocated to single sub-channel can be stated as,

$$p_n = \textbf{min}\left\{ \frac{N_0}{h_{n,k}^2} g(\frac{Z_{n,m}}{\alpha_{n,k}}), \frac{p_n^{max}}{b_{n,k}} \right\} \tag{6}$$

where p_n^{max} is the maximum transmission power and $g(x) = 2^{(\frac{x}{b_{n,k}W_0})} - 1$. According to DVS, the optimal computation speed at the mobile device should be adjusted to,

$$f_n = \textbf{min}\left\{ \left[\frac{Y_{n,m}}{D_n - \alpha_{n,k} - \beta_{n,k}}\right]^+, f_n^{max} \right\} \tag{7}$$

with f_n^{max} as the maximum computation speed at the mobile device.

B. Energy Consumption Problem

To simplify the problem, we redefine the symbol Θ as the joint computation configuration,

$$\Theta_n = \Theta_n \times \{p_n\} \times \{f_n\}, \forall n \in \mathcal{N}$$

which includes the DAG partition model, and the corresponding optimal configuration for transmission power and computation speed at the mobile device. Based on above discussion, the overall energy consumption function can be formulated as,

$$\begin{aligned} \xi_{n,k}(\Theta_n) &= \kappa Y_{n,m} f_n^2 + b_{n,k} p_n \frac{Z_{n,m}}{r_{n,k}} \\ &= \kappa Y_{n,m}[\frac{Y_{n,m}}{D_n - \alpha_{n,k} - \beta_{n,k}}]^2 + b_{n,k}\frac{N_0}{h_{n,k}^2} g(\frac{Z_{n,m}}{\alpha_{n,k}})\alpha_{n,k} \end{aligned} \tag{8}$$

Thus, the joint optimization of task energy efficient problem can be stated as,

$$E_{n,k} = \min_{\Theta_n \in \{1,\ldots,M_n\}} \left\{ \min_{\alpha_{n,k}} \xi_{n,k}(\Theta_n) \right\}, \ \forall k \in \mathcal{K} \qquad \textbf{(P1)}$$

s. t.

C1: $\alpha_{n,k} \in [\frac{Z_{n,m}}{b_{n,k} W_0 \log_2(1+\frac{p_n^{max} h_{n,k}^2}{b_{n,k} N_0})}, D_n - \beta_{n,k} - \frac{Y_{n,m}}{f_n^{max}}]$

C2: $E_{n,k} < E_n^l$

Such formulation makes (**P1**) a Mixed-Integer Nonlinear Programming (MINLP) problem. The constraint **C1** specifies the limitations in terms of task deadline, maximum transmission power and computation speed of the mobile device. **C2** identifies the beneficial task offloading decision. It can be easily proved that the inner objective function Eq (8) is convex *w.r.t.* the variable $\alpha_{n,k}$. Therefore, the optimal solution $\alpha_{n,k}^*$ exists either at the stationary point or at the minimum or at maximum boundary of the feasible solution. After the inner problem is solved, the solution to (**P1**) can be obtained by iterating all partition models and selecting the model that provides the minimal energy consumption.

In this paper, we propose the Computation and Power Optimization (CPO) Algorithm 1 to calculate the optimal data transmission time and find the best DAG partition model under given offloading decision a_n. The algorithm will run separately for each task and the step size ϵ will control the accuracy of the algorithm.

Algorithm 1: Computation and Power Optimization

initial step size $\epsilon = 10^{-3}$, minimal energy consumption $\xi_n^* = +\infty$
for $m \in \{1, 2, .., M_n\}$ **do**
  get α_{min} and α_{max} based on constraint **C1**
  if $\alpha_{min} > \alpha_{max}$ **then**
    continue
  set $\alpha = \alpha_{min}$
  while $\alpha \leq \alpha_{max}$ **do**
    based on Eq. (8), calculate energy ξ_n
    if $\xi_n < \xi_n^*$ **then**
      get p_n and f_n based on Eq. (6) and Eq. (7)
      $\Theta_n^* \leftarrow \{m, p_n, f_n\}$, $\xi_n^* \leftarrow \xi_n$
    $\alpha = \alpha + \epsilon \cdot (\alpha_{max} - \alpha_{min})$
if $\xi_n^* < E_n^l$ **then**
  return Θ^*
else
  return $\emptyset$

V. Task Offloading Game

In this section, we describe the formulation and solution of task offloading game in order to minimize the energy consumption at mobile devices. In order to achieve this, we introduce time variable t as the iteration slot and denote $a_n(t)$ as the offloading decision made by device n at tth iteration. Since the resources are shared among device who have chosen the same offloading decision (except $a_n(t) = 0$), any single decision update $a_n(t-1) = k \rightarrow a_n(t) = k'$ at iteration t would incur energy consumption overhead to the existing tasks on the edge server k', and also release the occupied resources at the original edge server k. Subsequently, those changes result in re-optimizing the transmission power and computation speed of mobile devices; otherwise, such tasks might fail to meet the deadline and the devices would spend extra energy from unnecessary computation.

A. Game Formation

Based on the above discussion, we define a two dimension strategy space of all devices as,

$$\mathcal{S} \triangleq [\mathcal{A}, \Theta] \tag{9}$$

where $s_n(t) = [a_n(t), \Theta_n(t)]$ indicates the joint DAG partition model $\Theta_n(t)$ along with the offloading decision $a_n(t)$. We also denote $s_{-n}(t) = (s_0(t), ..., s_{n-1}(t), s_{n+1}(t), ..., s_N(t))$ as the offloading strategies by all other devices except for n. The energy consumption function for device n can be formulated as,

$$\eta_n(s_n(t), s_{-n}(t)) = \begin{cases} E_n^l & \text{if } I_{\{a_n(t)=0\}} \\ E_{n,k}(\Theta_n(t)) & \text{if } I_{\{a_n(t)=k\}} \end{cases} \tag{10}$$

We next formulate the multi-device and multi-server task offloading problem as a strategic game with the energy consumption function η_n. Given $s_{-n}(t)$, each device would like to choose a ideal strategy $s_n(t)$ in order to minimize its own energy consumption (in a selfish manner), that is $\forall n \in \mathcal{N}$,

$$\min_{s_n(t) \in S} \eta_n(s_n(t), s_{-n}(t)) \tag{11}$$

B. Nash Equilibrium and Convergence

Here, we introduce the concept of Nash equilibrium for the proposed task offloading game. We assume that the mobile devices are selfish in nature and are only concerned about minimizing their own energy consumption.

***Definition** 2:* A strategy profile $\mathcal{S}^* = \{s_1^*, s_2^*, ..., s_N^*\}$ is a Nash equilibrium (NE) of multi-device and multi-server task offloading game, if at the equilibrium $\mathcal{S}^*$, no player (device) can further reduce its energy consumption by unilaterally altering its strategy, i.e.,

$$\eta_n(s_n^*, s_{-n}^*) \leq \eta_n(s_n, s_{-n}^*), \forall s_n \in \{1, 2, ..., K\}, n \in \mathcal{N}$$

***Theorem** 1:* The multi-device and multi-server task offloading game with a global cost function $\phi(\mathcal{S})$ defined in Eq. (12) is a potential game which always has a NE.

$$\phi(\mathcal{S}) = \sum_{n=1}^{N} \kappa Y_{n,m} f_n^2 + p_n \frac{Z_{n,m}}{W_0 \log_2(1+\frac{p_n h_{n,k}^2}{N_0})} \tag{12}$$

Proof 1: Let us consider the update $s_n(t-1) \neq s_n(t)$ for n. Such motivation can be stated as,

$$\eta(s_n(t-1), s_{-n}(t-1)) > \eta(s_n(t), s_{-n}(t-1))$$

Base on Eqs. (3) and (8), the energy consumption is independent of sub-channel allocation i.e., the strategy chosen by device n is independent of the transmission power and computation speed of other devices. Thus, the energy consumption of all other devices will remain unchanged at that moment. Thus, it is evident that in such a scenario the following inequality will also be satisfied.

$$\phi(s_n(t-1), s_{-n}(t-1)) > \phi(s_n(t), s_{-n}(t-1))$$

■

This shows that the multi-device and multi-server task offloading game using the potential function (given in Eq (12)) is a potential game. Therefore, the game will always have a NE $\mathcal{S}^*$ and the finite improvement property (FIP).

C. Algorithm

Based on the FIP of the proposed task offloading game, we select the best strategy to find the improvement path that leads to a NE [11], [12]. We develop a Joint Task Offloading Game (JTOG) Algorithm 2 to perform strategy updates.

In JTOG, the strategies of all other devices $s_{-n}(t)$ will be given at each iteration. Each device **separately** uses CPO algorithm (Algorithm 1) to solve its own energy minimization problem (11) by iterating for all edge servers and sends the best strategy $s_n(t) = [a_n(t), \Theta_n(t)]$ as a request to the central edge controller to compete for an update opportunity. However, only one update request is accepted at each iteration among devices who want to change their offloading decisions ($s_n(t-1) \neq s_n(t)$). The algorithm terminates when there is no new update request.

Algorithm 2: Joint Task Offloading Game

initial $\mathcal{S}^* = [0, \emptyset]$, $t = 0$ and max iteration t_M
all devices select local computation model based on Eq. (4)
while $t < t_M$ **do**
  $request = \emptyset$
  for *each device* $n \in \mathcal{N}$ **do**
    solve $s_n(t) = \underset{s_n(t) \in S}{\textbf{argmin}}\ \eta_n(s_n(t), s_{-n}(t))$
    if $s_n(t-1) \neq s_n(t)$ **then**
      $request.add(s_n(t))$
  if $request \neq \emptyset$ **then**
    randomly accept one $request : s_n^* \leftarrow s_n(t)$
  else
    break
return $\mathcal{S}^*$

VI. Simulation Results

We evaluate the performance of the proposed multi-device and multi-server task offloading game using a simple yet realistic simulation. The number of sub-channels for each edge server is set to $[20, 50]$ and the bandwidth of each sub-channel is 1 MHz. The computation capacity of mobile devices and the computation capacity allocated from edge servers are $[1.75, 2.5]$ GHz and $[1.25, 2.75]$ GHz, respectively. The sub-channel gains are modeled based on [4] and the white Gaussian noise N_0 is set to 10^{-9}. The energy consumption coefficient is set to 10^{-28} Joules/cycle. The compute-intensive tasks are generated by DAGs with $[2-4]$ jobs where the job complexity varies between $[15, 150]$ cycle/bit. The size of the task input data and the intermediate data are set to $[0.35, 1.5]$ MB. The deadline and release period of tasks are set to $[0.5, 1]$ seconds.

A. Existence of Nash Equilibrium

Here we demonstrate how the process of joint task offloading game terminates after few iterations, beyond which no mobile device can further reduce its energy consumption by unilaterally changing its strategy; i.e., all devices satisfy their final task offloading strategies at NE and have no incentive to deviate from their strategies.

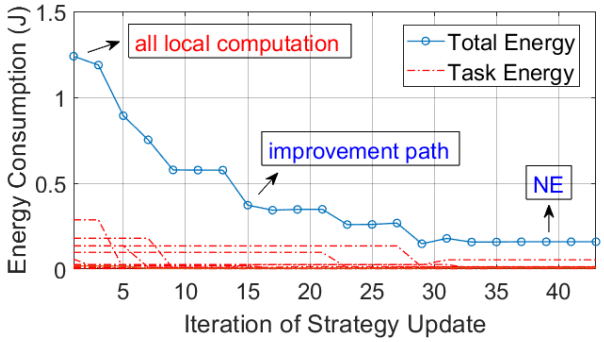

Fig. 3: The energy consumption of task offloading game with 15 tasks and 3 edge servers.

Fig. 3 shows that all devices select local computation model (i.e., $a_n(0) = 0$) at the very beginning of the game. Based on FIP, the improvement path shown in the figure that demonstrates the strategy update sequence carried out by the best response strategy leading to a NE after 40 iterations. In the simulation of a fixed number of edge servers, we found that the number of required iterations for achieving a NE increases almost linearly with the number of tasks. As shown in Fig. 4, the proposed task offloading game using the proposed parameters obtains a NE after at most $3N$ iterations. Therefore, the convergence of the JTOG algorithm (Algorithm 2) is fast and guaranteed.

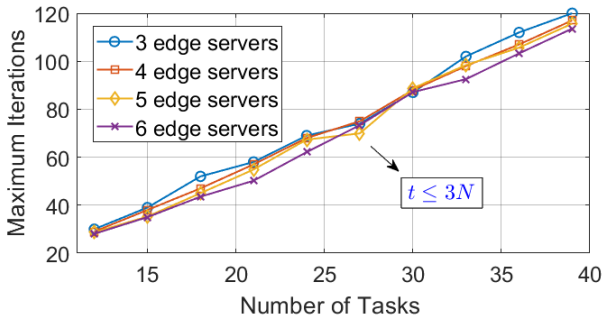

Fig. 4: The maximum iterations at NE for task offloading game with 3 to 6 edge servers and different number of tasks.

B. Performance Comparison

Next, we compare the performance of our proposed joint task offloading game against two other strategies: a) strategy where all jobs are executed at the edge server (EDGE ONLY), i.e., whenever a device chooses $a_n(t) > 0$, $\Theta_n(0)$ is selected as the DAG partition model; and b) strategy with local only computation (LOCAL ONLY), where mobile devices perform all jobs locally. The performance metric is simply the energy savings of all the devices combined which is expressed as,

$$1 - \sum_{n=1}^{N} \eta_n(s_n, s_{-n}) / \sum_{n=1}^{N} E_n^l \tag{13}$$

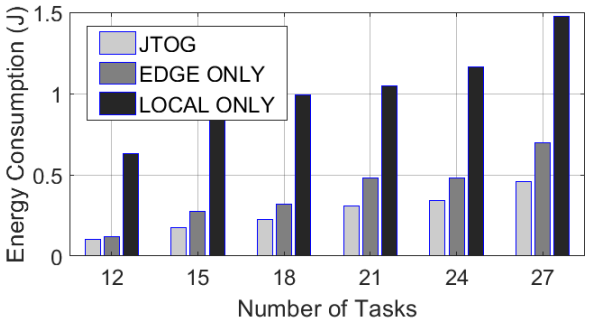

Fig. 5: The energy consumption of task offloading game with 3 edge servers and different number of tasks.

Fig. 5 shows the overall energy consumption over different number of tasks with fixed amount of network and computation resources. The figure shows that in comparison to LOCAL ONLY, JTOG performs (using Eq. (13)) significantly better in terms of overall energy consumption. However, the performance improvement decreases from 80% (12 tasks) to 60% (27 tasks) when there are more tasks compete for network and computation resources.

In contrast, Fig. 6 analyzes performance comparison for a given number of tasks against different number of sub-channels. Whereas, Fig. 7 shows the performance comparison against varying computation speed at the edge servers with a given number of tasks. From both figures, it is evident that the performance improvement increases when more resources (i.e., either network, or compute or both) are introduced into the system. However, due the nature of energy consumption function, the speed of such improvement reduces as more sub-channels are introduced as shown in Fig. 6.

At the same time, both Fig. 6 and Fig. 7 show that the JTOG and EDGE ONLY schemes can save considerable energy consumption at the NE point in comparison to LOCAL ONLY. However, JTOG saves more energy (in some cases twice as much) compared to full offloading strategy EDGE ONLY. The reason being that JTOG adaptively selects the DAG partition model Θ_n which ensures further energy reduction. This signifies that in certain situations running lightweight jobs at the mobile device (in order to get smaller intermediate data) before processing at the edge can conserve more energy than offloading the entire data to the edge for processing.

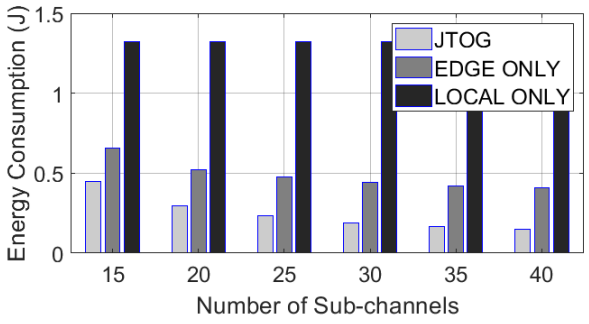

Fig. 6: The energy consumption of task offloading game with 15 mobile users and 3 edge servers against different number of sub-channels.

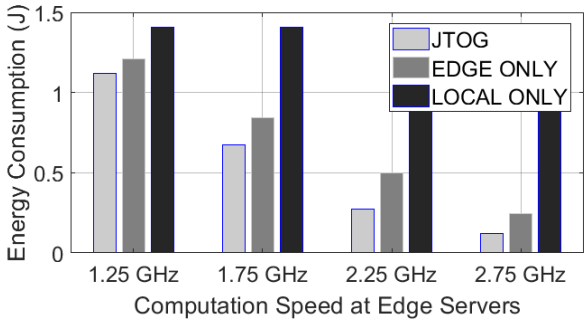

Fig. 7: The energy consumption of task offloading game with 15 mobile users and 3 edge servers against different edge server computation capacities.

VII. Conclusions

In this paper, we proposed a multi-device and multi-server task offloading game for edge systems running real-time compute-intensive applications. Through rigorous analysis, we showed that the proposed Joint Task Offloading Game (JTOG) algorithm always optimizes the energy saving for all mobile devices at a tractable Nash Equilibrium. Th results from a realistic simulation show that the proposed JTOG algorithm conserves more energy than traditionally intuitive schemes where all data are offlaoded to remote edge servers for processing.

References

[1] S. Yi, Z. Hao, Q. Zhang, Q. Zhang, W. Shi, Q. Li, "LAVEA: Latency-Aware Video Analytics on Edge Computing Platform," *Proc. of IEEE ICDCS*, 2017.

[2] Z. Dong, Y. Liu, H. Zhou, X. Xiao, Y. Gu, L. Zhang, C. Liu, "An energy-efficient offloading framework with predictable temporal correctness" *Proc. of ACM/IEEE SEC*, 2017.

[3] C-C. Hung, G. Ananthanarayanan, P. Bodk, L. Golubchik, M. Yu, P. Bahl, M. Philipose, "VideoEdge: Processing Camera Streams using Hierarchical Clusters," *Proc. of ACM SEC*, 2018.

[4] C. You, K. Huang, H. Chae, B. Kim, "Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397-1411, 2017.

[5] X. Cao, F. Wang, J. Xu, R. Zhang, S. Cui, "Joint computation and communication cooperation for mobile edge computing," *Proc. of WiOpt*, 2018.

[6] V. Bonifaci, A. Marchetti-Spaccamela, S. Stiller, A. Wiese, "Feasibility Analysis in the Sporadic DAG Task Model," *Proc. of Euromicro RTS*, 2013.

[7] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, D. O. Wu, "Energy-Optimal Mobile Cloud Computing under Stochastic Wireless Channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569-4581, 2013.

[8] M. Chen, Y. Hao, "Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587-597, March 2018.

[9] K. Cheng, Y. Teng, W. Sun, A. Liu, X. Wang, "Energy-Efficient Joint Offloading and Wireless Resource Allocation Strategy in Multi-MEC Server Systems," *Proc. of IEEE ICC*, 2018.

[10] J. Opadere, Q. Liu, N. Zhang, T. Han, "Joint Computation and Communication Resource Allocation for Energy-Efficient Mobile Edge Networks," *Proc. of IEEE ICC*, 2019.

[11] X. Chen, "Decentralized Computation Offloading Game for Mobile Cloud Computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974-983, April 2015.

[12] X. Chen, L. Jiao, W. Li, X. Fu, "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795-2808, October 2016.

[13] X. Zhang, S. Debroy, "Migration-driven Resilient Disaster Response Edge-Cloud Deployments", *Proc. of IEEE NCA*, 2019.