Multichoice Games for Optimizing Task Assignment in Edge Computing

Yongbo Li, Tian Lan
George Washington University, Washington, DC
{lib, tlan}@gwu.edu

***Abstract*—Mobile Edge Computing has fastly become a promising diagram to meet the ever-increasing demands imposed by emerging domains of applications and reduce the reliance on remote data centers in traditional cloud computing. In this paper, we address two problems unique in heterogeneous edge computing: the cost accounting and task assignment. To determine the cost of each unit of task being concurrently processed on an edge device, we propose to model the problem as a multichoice game, and use Shapley Value for cost accounting. With the total cost decoupled, we are able to leverage the distributive Hungarian algorithm to solve the task assignment problem efficiently. We adopt a hybrid manner for evaluations: we profile the costs (including energy and data transmission costs) using a fully implemented workload offloading framework in an edge environment, then the cost profiles are used to drive the simulations. Results show that our policy of task assignment guided by multichoice Shapley value is able to consistently outperform the two other baselines: *Random* policy and the policy of Hungarian assignment algorithm based on *Even* energy accounting. The advantage of our policy is further enlarged when the heterogeneity level of the network or computing resource in edge environments is increased. We also show interesting patterns of the joint effects of different resources' heterogeneity levels and the weighting factor between them, which provide useful inputs for edge resource optimization and management.**

I. Introduction

With the emerging of new domains of mobile applications, users now tend to perform more and more entertainments and works on their mobile devices. On the other hand, the new domains of applications, such as virtual reality, and deep learning that comes to mobile devices [13], also impose unprecedently strict requirements, in terms of both energy and latency. As mobile local computing and cloud-assisted computing [7] both fail to meet the requirements, a new computing diagram, named edge/fog computing [21], [6] aims to bring the computing resources closer to the users.

In an edge environment including several edge devices, a device can serve both as the role of edge client(a device needs to offload its workload to nearby edge devices of computing) and edge server (a device has spare resources to assist nearby devices to process tasks). A new property of edge environment is the higher level of resource heterogeneity compared to traditional cloud computing or cloud-let [19]. The edge devices can be equipped with different types of hardware setup (e.g., CPU, memory, network interfaces) and service configurations.

We consider the task assignment in an edge environment, with the goal of minimizing the total cost to process the edge workload. The number of various types of edge applications, edge devices make the optimization space prohibitively large to solve in a centralized manner. We formulate such task assignment optimization problem in this paper and prove that it is NP-hard. Further, the distributive nature of edge environment that each edge device self-optimizes its own benefits also motivates a low-cost and distributive solution to such task assignment problem. A key to achieve this is to decouple the overall cost consumed by the concurrent tasks on a single edge server, in other words, to determine the cost for each unit of task when an edge server is concurrently processing multiple units of different types.

In this paper, we propose to utilize the Shapley value for multichoice cooperative game for cost accounting, i.e. allocating the overall cost by an edge server to the delegated edge clients. The concurrently processing and cost accounting on a single edge server is modeled as the game itself, and overall cost (including computing cost and transmission cost) is the cost that needs to be accounted to all edge clients. Each type of the workload is considered as a single player in the game and the number of units for each type of workload is the 'choice' (or activity levels) [5] in the multichoice game. With the decoupled costs, the task assignment problem can be efficiently solved by a distributive Hungarian algorithm [16].

The main contributions of our work are summarized below:

(1) We propose a novel cost accounting method based on Shapley value for multichoice cooperative game, to allocate the overall cost consumed by an edge server to each unit of concurrently-processed tasks.
(2) We formulate the overall cost minimization in an edge environment as a joint task assignment problem and prove it is NP-hard. With the help of our accounting policy, we decouple the overall cost so as to relax the task assignment problem as a distributive problem and further efficiently solve it by Hungarian algorithm.
(3) We fully implement an edge framework which enables workload offloading and collaborative computing among heterogeneous edge devices. Evaluations with a hybrid manager of profiling and simulations using the framework prove our proposed task assignment algorithm based on Shapley value-based accounting policy provides consistent benefits over two other baseline policies.

II. Related Work

For the recently emerging area - Edge/Fog Computing [6], [21], prior work has studied multiple aspects include

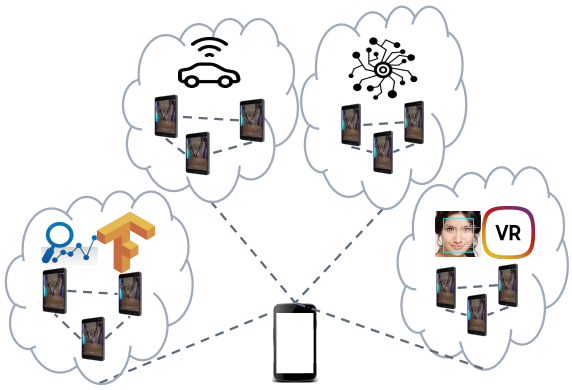

Fig. 1. Edge computing system diagram with multiple edge devices (each device can be part of multiple edge clouds)

edge-cloud cooperation [9], resource prediction and management [1], workload offloading techniques [7], [11], and task assignment optimization [14]. In specific, prior work [14] has considered a single client when optimizing workload offloading decisions, in contrast, the work in this paper considers a more realistic model when multiple edge clients and edge servers are active in the edge environment.

To decouple individual edge client's offloading decision making, we employ the Shapley value for multichoice games [5], an extension to the Shapley Value [20], which has been broadly utilized to solve Engineering problems, such as bill accounting [3], incentive design, and optimization decision decentralization [22].

III. System Model and Problem Formulation

We consider an edge computing system, as illustrated in Figure 1, which can contain multiple *edge clouds*. An edge cloud consists of the edge devices supporting a given type of edge service. Note that an edge device can concurrently support multiple edge services, i.e., it can participate in multiple edge clouds at the same time. An edge server contributes available spare resources and assists edge clients in task processing. We note that the same edge device can be both an edge server and/or an edge client. For a given period, We use $\mathbf{M}$ to denote a set of M edge servers and $\mathbf{K}$ a set of K edge clients.

Suppose that there are W distinct types of tasks (denoted by a set $\mathbf{W}$) supported by the edge computing system. Due to heterogeneity of the edge servers and tasks, each server $m \in \mathbf{M}$ is able to handle a subset of the tasks, denoted by $\mathbf{W}_m \subseteq \mathbf{W}$. Without loss of generality, we assume that each edge client $k \in \mathbf{K}$ has only a single type of task $w_k \in \mathbf{W}$ to offload. It is easy to see that any edge devices generating more than one type of tasks can be split into multiple (logical) edge clients with the same physical configuration and network conditions. An edge client k can only offload its workload to an edge server m if it is supported by the edge server, i.e., $w_k \in \mathbf{W}_m$. Similarly, the set of edge servers that can process type-w workload is given by $\mathbf{M}_w = \{m | w \in \mathbf{W}_m\}$. We use a set of binary variables $x_{k,m}$ for all k, m to denote our workload offloading and assignment strategy, i.e,

$$x_{k,m} = \begin{cases} 1, & \text{if task } k \text{ is assigned to edge server } m, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Due to capacity constraints, we assume that each edge server m is able to concurrently process at most $a_{m,w}$ tasks of type w. This implies that the total number of type-w tasks assigned to edge device m (denoted by $b_{m,w}$) must be bounded by $a_{m,w}$: $b_{m,w} = \sum_{k:w_k=w} x_{k,m} \leq a_{m,w}, \ \forall m, w$, where the summation is over all edge clients that has task type $w_k = w$. Due to the task-processing constraints, $x_{k,m}$ can be 1 only if task k is supported by edge server m, namely $w_k \in \mathbf{W}_m$. Further, each task must be assigned to one single edge server, resulting in another constraint $\sum_m x_{k,m} = 1$.

The goal of this paper is to find the optimal offloading strategy and task assignment to minimize the total cost in the edge computing system. Similar to [12], we take into account (i) network cost $r_{k,m}$ for data transmission (between edge client k and edge server m) and (ii) energy consumption c_m at the edge servers to process assigned tasks. More precisely, for each edge server m, its energy consumption jointly depends on the numbers of tasks of different types that are assigned to server m, i.e., $\{b_{m,w}, \ \forall w\}$. We model this with an energy consumption function: $c_m = v_m(b_{m,w}, \ \forall w)$. For network cost, similar to [14], we consider the bandwidth consumption and transmission time to offload each client's task, and collectively represent the edge network cost by $r_{k,m}$, i.e., to offload edge client k's task to edge server m. Given equation (1), the network cost of edge client k is $\sum_{m \in \mathbf{M}} r_{k,m} x_{k,m}$. Thus, the cost considered in our optimization objective is

$$C = \alpha \sum_{k \in \mathbf{K}} \sum_{m \in \mathbf{M}} r_{k,m} x_{k,m} + \sum_{m \in \mathbf{M}} c_m, \quad (2)$$

where α is a tradeoff factor reflecting the relative importance of computing and network cost with respect to system design preferences. In practice, prior to formulating the task assignment problem, we can estimate the edge server energy cost and edge network cost $r_{k,m}$ based on system traces, the transmitted data size, and current system/network condition [14].

In this paper, we formulate a Joint Optimization of Task Assignment (JOTA) problem of all tasks, over $\{x_{k,m}, \ \forall k, m\}$, to minimize the aggregate cost C defined in equation (2). The JOTA problem is formulated as follows:

$$\begin{aligned}
&\text{Problem JOTA :} \\
&\min \quad \alpha \sum_{k \in \mathbf{K}} \sum_{m \in \mathbf{M}} r_{k,m} x_{k,m} + \sum_{m \in \mathbf{M}} (c_m) && (3) \\
&\text{s.t.} \quad c_m = v_m(b_{m,w}, \ \forall w), \ \forall m && (4) \\
&\qquad b_{m,w} = \sum_{k:w_k=w} x_{k,m} \leq a_{m,w}, \ \forall m, w && (5) \\
&\qquad \sum_{m \in \mathbf{M}} x_{k,m} = 1, \ \forall k && (6) \\
&\qquad x_{k,m} = 0 \text{ if } w_k \notin \mathbf{W}_m, \ \forall k, m && (7) \\
&\qquad x_{k,m} \in \{0, 1\}, \ \forall k, m && (8) \\
&\text{var.} \quad \{x_{k,m}, \ \forall k, m\}. && (9)
\end{aligned}$$

In the JOTA problem above, the number of type-w tasks assigned to each server m must satisfy a capacity constraint (5). Each task must be assigned to a single edge server, i.e., constraint (6). Finally, edge client k's task can be assigned to server m only if the task type w_k is supported by the edge server, as shown in constraint (7).

IV. Our Proposed Approach via Multichoice Games and Cost Accounting

We first prove that the JOTA problem is NP-hard even for 2 edge servers and 1 task type. The combination nature of the problem (e.g., energy cost $v_m(b_{m,w},\ \forall w)$ is jointly determined by all tasks assigned to an edge server) also prevents the efficient application of existing heuristics. Then, inspired by multichoice games in cooperative game theory, which provides a single-valued solution to decide each player's contribution to the overall payoff/cost, we make novel use of Shapley Value for multichoice game [5] - which is an extension of traditional Shapley Value [20] taking into account different activity levels of the players - to attribute the aggregate cost in the JOTA problem to individual edge clients/tasks, thus decoupling the joint optimization of task assignment. The resulting optimization after cost accounting reduces to a maximum bipartite matching [4] problem that can be readily solved using Hungarian algorithm [16] in polynomial time.

We note that while Shapley Value, without considering the multiple choices of players, has been widely applied to cost allocation problems [3], [2], [17], including that in mobile energy accounting [8], to the best of our knowledge, this is the first work using multichoice games for cost accounting and joint task assignment optimization in edge computing.

A. Problem JOTA is NP-hard

To prove the JOTA problem is NP-hard, we show a JOTA problem with two edge servers and a single task type can be reduced to a well-known NP-hard problem, max cut [10].

Theorem IV.1. *The problem JOTA is NP-hard.*

Proof. Consider a weighted undirected graph $\mathbb{G} = \mathbf{V}, \mathbf{E}$. The max cut problem finds a partition of the vertices in $\mathbf{V}$ into two sets, $\mathbf{S}_1$ and $\mathbf{S}_2$, to maximize the weight of the cut, i.e., the summation of weights of the edges that connect vertices of one set to the vertices of the other.

Now we formulate the max cut problem as a JOTA problem with two identical edge servers and a single task type supported by both edge servers. We represent each vertex in $\mathbf{V}$ as an edge client/task. Then, partitioning $\mathbf{V}$ into two sets is equivalent to assigning the tasks to the two edge servers. Let $\epsilon(a, b)$ be the weight of an edge $(a, b) \in \mathbf{E}$. We consider zero network cost $r_{k,m} = 0$ and construct an energy cost function $v(\mathcal{S}_i)$ for any subset of the tasks $\mathcal{S}_i \subseteq \mathbf{V}$ as follows:

$$v(\mathcal{S}_i) = \sum_{(a,b)\in\mathbf{E}: a\in\mathcal{S}_i, b\in\mathcal{S}_i} \epsilon(a, b), \tag{10}$$

which is the total weights of edges connecting two vertices within $\mathcal{S}_i$. Since total edge weights of the graph $\mathbb{G}$ is fixed, maximizing cut weight is equivalent to minimizing the total remaining edge weight, i.e., minimizing $v(\mathcal{S}_1) + v(\mathcal{S}_2)$, which is exactly the objective in the JOTA problem. Therefore, the JOTA problem is NP-hard following from the fact that max cut is NP-hard [10]. □

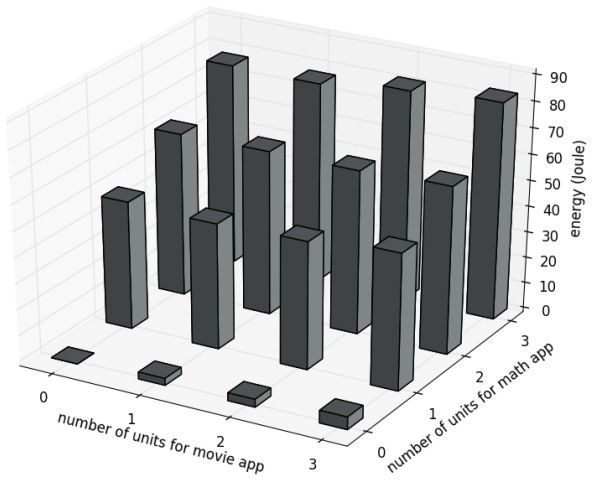

Fig. 2. Non-linearity in energy consumed by two types of concurrently-processed tasks

It is worth noting that it is not possible to solve the JOTA problem using existing heuristics developed for the max cut problem (or other graph cut problems), not only due to the large numbers of edge servers and task types that are encountered in edge computing, but also because a practical energy cost function $v_m(\cdot)$ can be arbitrary and it is difficult to transform it into a graph with edge weights satisfying (10). In this paper, we utilize cost profiling with several types of real-world mobile edge workload on real devices (with details given in Section V) and here we present a simple example of the cost profiles when only two applications (movie recommendation and computation-intensive math calculation) are active. The profiled energy costs are depicted in Figure 2 as a 3-D plot. It can be seen that the marginal increment in energy cost is affected by the number of tasks of both applications. The energy cost is not linearly proportional to the unit of any one of the two active applications.

Due to combinatorial nature of the JOTA problem, i.e., $v_m(\cdot)$ depends on the combination of all tasks assigned to an edge server, it incurs a huge solution space for arbitrary, nonlinear energy cost functions. Even finding an approximated numerical solution is very challenging, especially in an edge computing environment where a distributed solution is highly desirable due to the decentralized property of edge devices.

B. Cost Accounting via Multichoice Games and Shapley Value

The main difficulty in solving the JOTA problem is that energy cost $v_m(\cdot)$ is collectively determined by the combination of all tasks assigned to an edge server. To tackle such challenge, we make novel use of Shapley value for multichoice game to attribute $v_m(\cdot)$ to each task, thus decoupling

the energy cost and resulting in a linear approximation of the JOTA problem. We focus on an energy cost accounting problem - Given a set of tasks that are scheduled to the same edge server, how to determine their individual contribution to the total energy cost?

Consider a set of player denoted as $\mathbf{N} = \{1, 2, \ldots, n\}$. Each player i has the set of activity levels, $\mathbf{B}_i = \{1, 2, \ldots, \beta_i\}$. A coalition in multichoice cooperative game [5] is denoted by a vector $\vec{b} = [b_1, b_2, \ldots, b_i, \ldots, b_n]^T$, where $b_i \in \mathbf{B}_i$ shows the activity level of player $i \in \mathbf{N}$ in the coalition $\vec{b}$. Given a payoff (or cost) function $v(\vec{b})$ defined for each coalition $\vec{b}$, Shapely value of this multichoice cooperative game provides a single-value solution to distribute the grand payoff among all players and their activity levels.

Our key idea is that the energy cost accounting problem can be modeled as a multichoice cooperative game. For simplicity, we suppress the subscript m representing an edge server in Section III without causing any confusion, since each of energy cost accounting problem is solved with the scope of a single edge server. We denote each type of task as a player i, the number of type-i tasks assigned to an edge server as activity level b_i, the energy cost $v(\vec{b})$ as the payoff if a combination of tasks $\vec{b}$ are assigned to the edge server. In particular, the activity level must satisfy edge server capacity constraints (5) in the JOTA problem, i.e., the maximum activity level β_i denotes the highest number of type-i tasks that can be processed by the edge server. Therefore, the energy cost accounting problem - which determines the energy cost per task for each task type - can be solved using Shapley value for this multichoice cooperative game.

Let $\theta = [0, 0, \ldots, 0]$ be an all-zero activity vector with size n, $\mathbf{A}(\vec{b})$ the set of active players in the game with non-zero activity levels, $\vec{b}^{-i} = [b_1, b_2, \ldots, b_i - 1, \ldots, b_n]^T$ the activity vector obtained by reducing player i's activity level by 1 from $\vec{b}$. Then, Shapley value of the multichoice cooperative game is obtained by calculating the expected marginal contributions of individual players over all permutations of players and different activity levels:

$$\begin{aligned}\psi(v, \vec{b})(i) \quad = \sum_{\theta \le \vec{a} \le \vec{b}^{-i}} & \frac{\vec{a}(N)! \cdot (\vec{b}(N) - \vec{a}(N) - 1)!}{\vec{b}(N)!} \cdot \\ & \prod_{j \in \mathbf{A}(\vec{b})} \binom{b_j^{-i}}{\vec{a}_j} \cdot [v(\vec{a}^{+i}) - v(\vec{a})]\end{aligned} \tag{11}$$

where $\psi(v, \vec{b})(i)$ is the energy cost distributed to each unit of type-i task (i.e., each activity level of player i) in coalition $\vec{b}$. $\theta \le \vec{a} \le \vec{b}^{-i}$ if and only if $0 \le \vec{a}_j \le b_j^{-i},\ \forall j \in \{1, 2, \ldots, n\}$. Similar to $\vec{b}^{-i}$, $\vec{a}^{+i}$ is obtained by increasing the activity level of player i in $\vec{a}$ by 1. $\vec{a}(N) = \sum_{j=1}^{n} \vec{a}_j$. Note that the energy cost per activity level for player i not only depends on the activity level of player i itself, but is also affected by the activity levels of all other players in the game. Using equation (11) to calculate the per unit level's cost, we are able to decouple the cost for each unit of tasks when they are concurrently processed on an edge server.

Theoretically, calculating Shapley value of this multichoice cooperative game seems to require to know the cost (i.e., payoff) of all coalitions $\vec{b}$. When the energy cost function is only partially known, we can calculate Shapley value using various approximation approaches [24], [8], making the calculation feasible in practice.

C. Task Assignment Based on Accounted Cost

We apply the multichoice game approach to each edge server m to distribute the total energy cost $v_m(b_{m,w},\ \forall w)$ to different tasks, i.e., the energy cost for executing each type-w task on edge server m is $\psi_{m,w}$. Thus, the energy cost for edge client k with task type w_k is given by $\sum_m \psi_{m,w_k} x_{k,m}$, since $x_{k,m}$ is 1 if and only if edge client k's task is assigned to edge server m. This allows us to linearly approximate the optimization objective in the JOTA problem, which now becomes a Shapley-based JOTA problem:

Problem SJOTA :

$$\min \quad \sum_{k \in \mathbf{K}} \sum_{m \in \mathbf{M}} (\alpha r_{k,m} + \psi_{m,w_k}) x_{k,m} \tag{12}$$

$$\text{s.t.} \quad \sum_{k: w_k = w} x_{k,m} \le a_{m,w},\ \forall m, w \tag{13}$$

$$\sum_{m \in \mathbf{M}} x_{k,m} = 1,\ \forall k \tag{14}$$

$$x_{k,m} = 0 \text{ if } w_k \notin \mathbf{W}_m,\ \forall k, m \tag{15}$$

$$x_{k,m} \in \{0, 1\},\ \forall k, m \tag{16}$$

$$\text{var.} \quad \{x_{k,m},\ \forall k, m\}. \tag{17}$$

Now $\alpha(r_{k,m} + \psi_{m,w_k})$ is the aggregate energy and network cost for assigning edge client k's task to edge server m. The objective function and other constraints are linear in $x_{k,m}$.

Next, it is easy to see that the SJOTA problem can be modeled as a *maximum bipartite matching* [4]. We denote the edge clients/tasks as the first set of vertices $\mathbf{S}$, the servers' task slots (rather than the server itself) as the second set of vertices $\mathbf{D}$, the aggregate energy and network cost $\alpha(r_{k,m} + \psi_{m,w_k})$ (accounted using Equation (11)) as an edge weight between two vertices in $\mathbf{S}$ and $\mathbf{D}$. Further, according to (13), each edge server has $a_{m,w}$ slots for type-w tasks, and we assign an infinite cost if a task type is not supported by the slot, due to (15). The *bipartite matching* assigns each task to exactly one server slot to minimize the total weights of selected edges, yielding a solution to the SJOTA problem. We implement a distributive version of the Hungarian algorithm based on [16] to solve this bipartite matching. The algorithm is summarized in Figure 4.

V. Evaluations

We utilize a hybrid manner of simulations for evaluation of our work. We first deploy four types of mobile edge workload to multiple types of edge devices and profile their energy costs by different combinations of tasks. The edge devices include commodity laptops and workstations. Their CPU frequencies and network bandwidths are configured differently. With the energy profiles, we simulate an edge environment with 36 edge

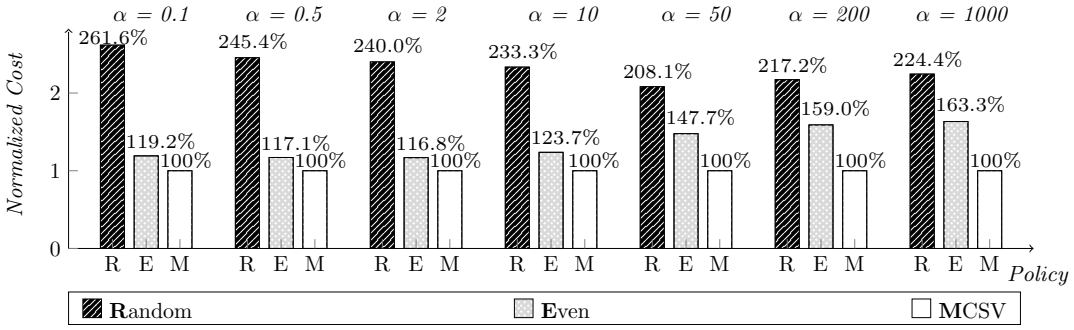

Fig. 3. Overall costs of an edge computing system of 12 servers and 24 clients. All costs are normalized by these achieved by multichoice Shapley value guided assignment optimization (M). α denotes the tradeoff factor reflecting the relative importance of computing cost and transmission cost

Initialization
  Construct cost matrix $\mathcal{C}$ based on Equation (11)
  Append dummy all-zero rows to make $\mathcal{C}$ square
// *(a) Extract mins from all rows and columns*
  for each row i, **do**
    $\mathcal{C}_{i,j} = \mathcal{C}_{i,j} - \min(\mathcal{C}_{i,j}, \forall j), \forall j$
  end for
  for each column j, **do**
    $\mathcal{C}_{i,j} = \mathcal{C}_{i,j} - \min(\mathcal{C}_{i,j}, \forall i), \forall i$
  end for
// *(b) Find maximum matching using only edges with* $\mathcal{C}_{i,j} = 0$
  if found
    return
  end if
  find minimum vertex cover V for subgraph of edges with $\mathcal{C}_{i,j} = 0$
// *(c) Adjust cost matrix* $\mathcal{C}$
  let $\sigma = \min_{i \notin V, j \notin V} \mathcal{C}_{i,j}$
  for all i and j, **do**
    if $i \notin V$ and $j \notin V$
      $\mathcal{C}_{i,j} = \mathcal{C}_{i,j} - \sigma$
    else if $i \in V$ and $j \in V$
      $\mathcal{C}_{i,j} = \mathcal{C}_{i,j} + \sigma$
    end if
  end for
  goto Step (b)

Fig. 4. Algorithm of maximum bipartite matching with energy accounting

devices. To simulate the heterogeneous property of the edge environment, each device randomly picks one from the energy profiles. Unless stated otherwise, we make 24 out of the total 36 devices as edge clients and 12 as edge servers. Each edge server can concurrently process 3 units of edge tasks in the experiment setup. The types and average unit data sizes for the workload are summarized in Table I.

To verify the effectiveness of our proposed multichoice Shapley value (*MCSV*) policy in guiding assignment optimization, we compare our policy with two other baselines: (1) *Random* - the offloaded workload is randomly assigned to available edge servers; (2) *Even-guided Optimization* - the total cost of an edge server is evenly allocated to all the units of tasks processed by the server, and based on the evenly accounted cost, the same Hungarian algorithm as our

TABLE I
FOUR TYPES OF PROFILED EDGE WORKLOAD USED FOR EVALUATION

Index	Edge Workload Type	Unit Data Size (Bytes)
1	collaborative filtering-based movie recommendation [18]	5349
2	eigenface-based face recognition [23]	59454
3	neural network-based emotion detection [15]	59454
4	computation-intensive math calculation	339

policy *MCSV-guided Optimization* is used to optimize the task assignment. To simplify the notation and distinguish our policy with *Even-guided Optimization* policy, we simply use *MCSV* and *Even* to denote these two policies.

A. Comparing Policies in Cost Minimization

We run 50 rounds of workload offloading and assignment simulations. In each run, we randomize the type of workload generated by each edge client. Considering the randomness in mobility and each device's network conditions, we also randomize the pairwise transmission costs between all devices. The results averaged over all runs are depicted as Figure 3. The comparison between the *Random* group and *Even* group shows the cost-saving achieved by Hungarian algorithm used for task assignment. More importantly, the comparison of our policy versus *Even* policy further demonstrates the difference made by our proposed *MCSV* accounting policy in guiding assignment optimization.

To illustrate the effects of different accounting and task assignment policies, we show the break-down offloading decisions for representative weight 50. We randomly generate one set of tasks using 24 edge clients. The number of units for workload with type index $[1, 2, 3, 4]$ are $[3, 6, 6, 9]$. The break-down task assignment decisions are seen in Table II. Especially, for *Even* policy and *MCSV* policy, even if both of them adopt Hungarian algorithm in guiding task assignment, their optimal decisions obtained vary significantly, due to the different accounting policies. Investigations of the edge servers' power profiles reveal that the accounting policy adopted by *MCSV* is more effective to allocate cost to individual unit of task in a discriminative manner, so as to better reflect the relative efficiency of edge servers in processing different types

TABLE II
TASK ASSIGNMENT DECISIONS UNDER DIFFERENT ACCOUNTING POLICIES. DETAILS FOR WORKLOAD OF EACH INDEX ARE SHOWN IN TABLE I

Edge Server Index	1	2	3	4	5	6	7	8	9	10	11	12
Random	0,0,0,0	0,1,1,1	0,0,0,1	1,1,0,0	1,0,0,2	0,1,0,0	0,2,0,0	1,0,0,1	0,0,1,1	0,0,3,0	0,0,0,2	0,1,1,1
Even	0,1,1,1	0,0,0,3	0,3,0,0	0,0,0,0	0,0,0,0	0,0,1,2	0,0,0,0	1,1,1,0	1,0,2,0	0,0,0,0	1,1,0,1	0,0,1,2
MCSV	0,0,3,0	0,0,3,0	0,3,0,0	0,0,0,0	0,0,0,0	0,0,0,3	0,0,0,0	3,0,0,0	0,0,0,3	0,0,0,0	0,3,0,0	0,0,0,3

of workload. For example, server 1 and server 2 has lower cost in processing the type-2 workload, while server 6, 9, and 12 are better off in processing the type-3 workload. Such differences in different servers' cost profiles are reflected in the accounted cost by our proposed policy, thus serving as effective inputs in guiding the Hungarian task assignment algorithm. In contrast, an improper accounting policy like *Even* policy fails to achieve this.

B. Studying Effects of Number of Edge Clients

To further study the effects of different ratios of clients to servers, we fix the number of edge servers to 12 while increasing the number of clients from 12 to the maximum possible amount 36 (since the maximum number of units of tasks concurrently processed by an edge server is 3). For each setup, the same set of 50 rounds of experiments is conducted. The results are plotted as Figure 5, from which we can see that as the numbers of clients increase, the normalized cost of *Random* policy decreases while that of *Even* policy increases. Finally, when the number of clients reaches the maximum, these two policies converge to very close numbers. In other words, when the number of clients is large, without proper cost accounting policies, even with Hungarian algorithm applied to optimize task assignment, it can perform at most slightly better than a totally random task assignment. While our approach of task assignment algorithm guided by multichoice Shapley value accounting policy largely and consistently outperforms the baselines. Such pattern is also verified in experiments with varied numbers of servers and tradeoff factors α. For the sake of space limitation, they are left out here.

C. Studying Effects of Network and Computing Heterogeneity

The costs in an edge computing environment involve the network cost and workload computing cost. To evaluate the effects of different levels of heterogeneity, we tune the network conditions and the computing capacity of edge devices, and run the same batches of workload offloading experiments. To test the effects of different network conditions, we add two more groups of experiments with new environment setups: (1) decreased network heterogeneity, in which we decrease the differences between network conditions of different edge servers; and (2) increased network heterogeneity, which is with the opposite changes. The results are presented as Figure 6. Similarly, we add two more groups of experiments of decreased computing heterogeneity and increased computing heterogeneity, to study the effects of varying computing capacity of edge servers. The results are shown in Figure 7.

These two new sets of results firstly demonstrate that our policy is superior to the two baselines *Random* and *Even*

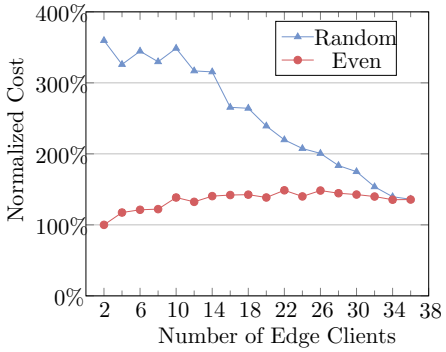

Fig. 5. Costs of two baseline policies normalized by these of our policy, when number of edge clients increases from 12 to the maximum 36

policies, even with different conditions of network and computing heterogeneity. Further, comparing the results with the main results shown in Figure 3, it is clear that when the heterogeneity level of either network or computing capacity is increased, the percentage of improvements of our policy over others is further enlarged. From this perspective, it also demonstrates the necessity for a task assignment policy to consider multiple resources in an edge environment, where different devices normally are highly heterogeneous in terms of different types of resources. Finally, these results together with the results of Figure 3 also give more information on the overall effects of different resources' heterogeneity levels and the weighting factor α. In general, which type of resource heterogeneity outweighs the other will affect the trend of results over the weighting factor α. For example, in Figure 3, comparing the results of our policy with *Random* policy, we can see as the α increases, the relative benefits of our policy first diminishes, and then it begins to increase at the turning point $\alpha = 50$. This is because when α is smaller than 50, the computing capacity heterogeneity dominates the results, and after the turning point, the network heterogeneity begins to dominate. With the results of Figure 6, we can verify the fact that by increasing the network heterogeneity level, the 'turning point' moves to left, and decreasing it moves the 'turning point' to right. The results of Figure 7 shows changing the computing heterogeneity level gives the opposite effect with network heterogeneity. Such interesting observations give useful insights for edge computing environment, so that the optimization for utilization of multiple resources can accommodate different requirements in an informed manner.

VI. CONCLUSION

In this paper, we study two correlated problems with significant importance for heterogeneous edge environments: First,

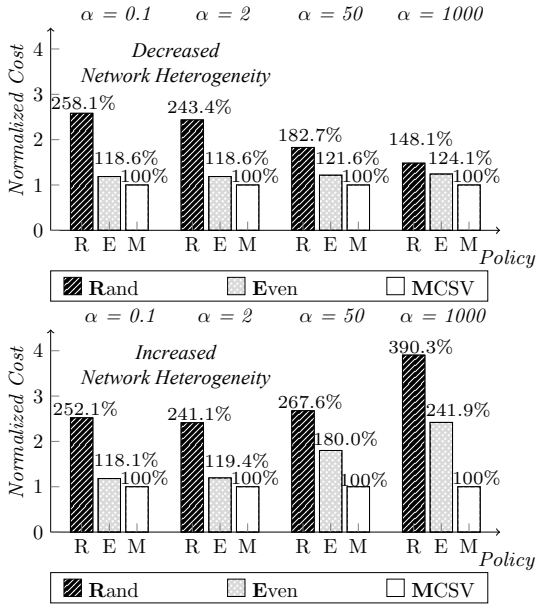

Fig. 6. Overall normalized costs of different policies, when varying the network heterogeneity level.

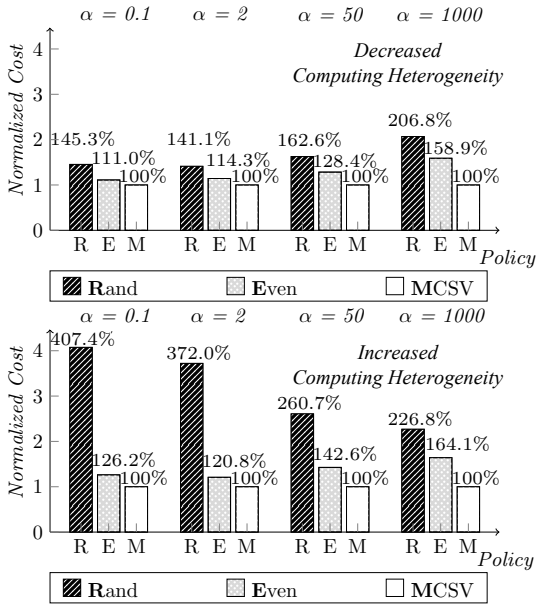

Fig. 7. Overall normalized costs of different policies, when varying the computing heterogeneity level.

given the tasks concurrently processed on an edge server and the overall cost, how to determine the cost for each individual unit of task. Second, how to assign submitted workload from edge clients to edge servers such that the overall cost is minimized. For the first question, we propose an answer based on Shapley value for multichoice game. Then, with the total cost decoupled, we utilize the distributive Hungarian algorithm to optimize the task assignment. By hybrid evaluations with real implementation-based cost profiling and profile-driven simulations, we show that our policy outperforms the two other baselines. Further, increased levels of resource heterogeneity further enlarge the improvements of our policy, which make our policy advantageous to fit the need of heterogeneous edge environment. The joint effects of different resources' heterogeneity levels and the weighting factor, which we present in this paper also provide meaningful insights for edge resource optimization.

REFERENCES

[1] M. Aazam and E.-N. Huh. Fog computing micro datacenter based dynamic resource estimation and pricing model for iot. In *AINA*, pages 687–694. IEEE, 2015.

[2] L. J. Billera and D. C. Heath. Allocation of shared costs: a set of axioms yielding a unique procedure. *Mathematics of Operations Research*, 7(1):32–39, 1982.

[3] L. J. Billera, D. C. Heath, and J. Raanan. Internal telephone billing rates - a novel application of non-atomic game theory. *Operations Research*, 26(6):956–965, 1978.

[4] R. E. Burkard, M. Dell'Amico, and S. Martello. *Assignment problems, revised reprint*, volume 125. Siam, 2009.

[5] E. Calvo and J. C. Santos. A value for multichoice games. *Mathematical Social Sciences*, 40(3):341–354, 2000.

[6] M. Chiang. Fog networking: An overview on research opportunities. *arXiv preprint arXiv:1601.00835*, 2016.

[7] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl. MAUI: making smartphones last longer with code offload. In *MobiSys*. ACM, 2010.

[8] M. Dong, T. Lan, and L. Zhong. Rethink energy accounting with cooperative game theory. In *MobiCom*. ACM, 2014.

[9] U. Drolia, K. Guo, J. Tan, R. Gandhi, and P. Narasimhan. Cachier: Edge-caching for recognition applications. In *ICDCS*. IEEE, 2017.

[10] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified np-complete graph problems. *Theoretical computer science*, 1976.

[11] M. S. Gordon, D. A. Jamshidi, S. Mahlke, Z. M. Mao, and X. Chen. COMET: code offload by migrating execution transparently. In *OSDI*. USENIX, 2012.

[12] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel. The cost of a cloud: research problems in data center networks. *ACM SIGCOMM computer communication review*, 39(1):68–73, 2008.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[14] Y. Li, Y. Chen, T. Lan, and V. Guru. MobiQoR: Pushing the Envelope of Mobile Edge Computing via Quality-of-Result Optimization. In *ICDCS*. IEEE, 2017.

[15] L. Ma and K. Khorasani. Facial expression recognition using constructive feedforward neural networks. *IEEE SMC*, 34(3):1588–1595, 2004.

[16] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

[17] A. E. Roth and R. E. Verrecchia. The shapley value as applied to cost allocation: a reinterpretation. *Journal of Accounting Research*, pages 295–303, 1979.

[18] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*. ACM, 2001.

[19] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The case for vm-based cloudlets in mobile computing. *IEEE pervasive Computing*, 8(4), 2009.

[20] L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317, 1953.

[21] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 2016.

[22] M. Shubik. Incentives, decentralized control, the assignment of joint costs and internal pricing. *Management science*, 8(3):325–343, 1962.

[23] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *CVPR*. IEEE, 1991.

[24] S. J. Willson. A Value for Partially Defined Cooperative Games. *International Journal on Game Theory*, 21:371–384, 1993.