Future Generation Computer Systems 95 (2019) 89–99

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

# Energy efficient task allocation and energy scheduling in green energy powered edge computing

Lin Gu [a], Jingjing Cai [a], Deze Zeng [b,*], Yu Zhang [a], Hai Jin [a], Weiqi Dai [a]

[a] Services Computing Technology and System Lab, Big Data Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
[b] School of Computer Science, China University of Geosciences, Wuhan, China

## HIGHLIGHTS

- This paper investigates the energy cost minimization problem in edge computing.
- We jointly consider the task allocation, service migration and energy scheduling.
- A relaxation-based heuristic algorithm is proposed to reduce energy consumption.

## ARTICLE INFO

*Article history:*
Received 2 May 2018
Received in revised form 3 October 2018
Accepted 29 December 2018
Available online 3 January 2019

*Keywords:*
Green energy
Edge computing
VM migration
Task allocation
Energy allocation
Energy efficiency

## ABSTRACT

The ever-increasing computation tasks and communication traffic have imposed a heavy burden on cloud data centers and also resulted in a significantly high energy consumption. To ease such burden, edge computing is proposed to explore the distributed resources of edge devices (e.g., base stations) to provision the cloud services for latency-sensitive applications at the network edge. Owing to the geo-distribution of edge devices, edge computing is also an ideal energy efficient platform to leverage the distributed green energy for energy efficient computing. Thus, it is natural to integrate Energy Internet (EI) technology into edge computing for customizable energy scheduling. In such EI supported edge computing, both the green energy generation rates and the data processing demands vary in different time and space. To pursue high energy efficiency, it is desirable to maximize the utilization of green energy so as to reduce the brown energy consumption. This requires careful task allocation and energy scheduling to match the energy provision and demand. In this paper, we investigate the energy cost minimization problem with joint consideration of VM migration, task allocation and green energy scheduling and prove its NP-hardness. To tackle the computation complexity, a heuristic algorithm approximating the optimal solution is proposed. Through extensive simulations, we show that the proposed algorithm can efficiently reduce brown energy consumption and perform much close to the optimal solution.

© 2018 Published by Elsevier B.V.

## 1. Introduction

Nowadays, cloud computing provides various services to global users with high resource utilization, strong computing ability, and high service reliability. According to Cisco's Global Cloud Index Report (2015–2020), 92% of global computational workloads is processed in cloud. However, with the rapid increasing of user tasks, the bulk data transmission and processing impose a heavy burden on the communication bandwidth and computation resource of cloud, bringing unbearable service delay to end users [1–3]. Moreover, the significantly high energy consumption of cloud also becomes another critical issue with growing concerns [4,5].

To handle this challenge, the concept of edge computing, which extends cloud computing to the network edge, is proposed. Edge computing utilizes the distributed resources in routers, gateways, base stations and even mobile devices to offer "cloud" services to the users in proximity [6,7], as shown in Fig. 1. By directly processing user tasks in local edge devices, the transmission delay and the traffic congestion on the Internet can be effectively reduced, thereby improving the user experience [8]. Furthermore, edge computing platform is formed by geo-distributed edge devices, which can harvest green energy from the environment. Such paradigm naturally provides an ideal platform to utilize the local green energy [9]. Actually, many studies [10–13] have already mentioned the possibility and advantages of utilizing green energy in edge cloud to reduce the brown energy consumption. Meanwhile, Energy Internet (EI), as a newly emerging technology, provides a promising means for flexible and customizable energy

* Corresponding author.
*E-mail address:* deze@cug.edu.cn (D. Zeng).

https://doi.org/10.1016/j.future.2018.12.062
0167-739X/© 2018 Published by Elsevier B.V.

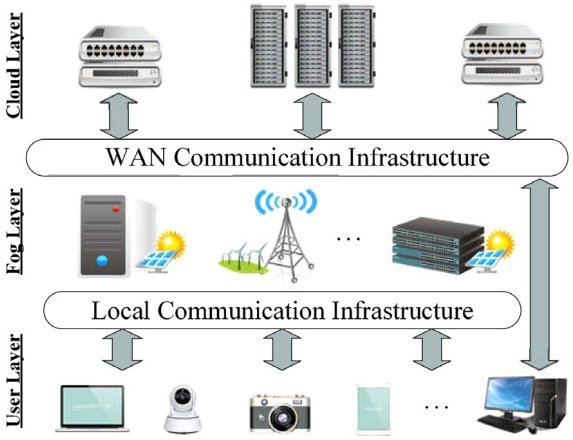



**Fig. 1.** A three-layer edge computing architecture.

scheduling in power grid [14,15]. Therefore, we can naturally integrate EI with edge computing to support the green energy powered edge computing such that the energy can be also managed in a fine-grain way like cloud resource management [16]. Thanks to the support of EI, it is possible to jointly manage the edge resources, including both computation and communication resources, and the energy resources, especially the green energy, in green energy powered edge computing.

To pursue sustainable green edge computing, it is always desirable to maximize the utilization of green energy and lower the reliance on brown energy. However, we notice that most of existing studies (e.g., [10–13]) on green energy powered edge computing only focus on scheduling local energy separately in each node and none of them considers the energy transferring between edge nodes. Thanks to the introduction of EI, it is possible to freely transfer the green energy between the edge nodes. In green energy powered edge computing, the availability of green energy vary in different areas and time, while the user tasks are also dynamically changing. This raises a critical challenge on how to migrate the virtual machine (VM) and allocate the tasks in response to the time-varying and geo-distributed diverse green energy generation and user task demands. In this case, a global and centralized scheduling of green energy in all edge nodes shall be helpful to match the user tasks and energy supply, e.g., transferring residual green energy to the heavy-loaded nodes. In this paper, we are motivated to study green energy scheduling, task allocation and VM migration problem in order to reduce the brown energy consumption. The following questions need to be answered: (1) Where shall we place the VMs according to user tasks and energy availability? (2) How to jointly schedule the green energy transmission between edge nodes with the consideration of energy transferring attenuation?

To better understand the problem discussed above, let us consider a simple edge computing platform with three nodes, as shown in Fig. 2. The total energy consumption consists of two aspects: (1) migration energy incurred by migrating VMs between edge nodes, and (2) processing and communication cost of user tasks in different edge nodes. Each edge node is with certain amount of green energy, i.e., $E_1(t) = 10$, $E_2(t) = 20$ and $E_3(t) = 15$, which can be transmitted to the other edge nodes with different attenuation ratios determined by the geographic locations. For example, the attenuation ratios of green energy transmission between edge nodes are $A_{12} = 0.3$, $A_{23} = 0.1$ and $A_{13} = 0.2$, respectively. While the user task arrival rates vary on different edge nodes, i.e., $\lambda_1 = 5$, $\lambda_2 = 3$ and $\lambda_3 = 2$. A task can only be processed in the edge node with the corresponding VM (e.g., node $n1$ in Fig. 2(a)). We assume that the energy consumption for processing one task in each edge node is 5. The communication energy of one task relies on the distances between edge nodes. For example, transmitting one task between nodes $(n1, n2)$, $(n2, n3)$, and $(n1, n3)$ are $H_{12} = 2$, $H_{23} = 1$ and $H_{13} = 2$, respectively. Now, we consider how to migrate the VM and schedule the energy between the three edge nodes, according to current green energy availability and user task demands. A simple solution is to process all user tasks in $n1$ and transmit green energy from both $n2$ and $n3$ to $n1$. In this case, we can see from Fig. 2(b) that the total energy consumed for processing all 10 tasks is 50 and the communication energy from $n2$ to $n1$ and $n3$ to $n1$ is $\lambda_2(t)\cdot H_{12}\cdot 1+\lambda_3(t)\cdot H_{13}\cdot 1 = 13$. That is, the total energy consumption is 63 and the available green energy on $n1$ is $E_1(t) + E_2(t)\cdot(1 - A_{12}) + E_3(t)\cdot(1 - A_{13}) = 38$, requiring 25 units of brown energy. If we migrate the VM to node $n2$ with a migration energy of 3, as shown in Fig. 2(c), the total energy consumption becomes $50 + 3 + \lambda_1(t)\cdot H_{12} + \lambda_3(t)\cdot H_{23} = 70$ while the green energy is $E_1(t)\cdot(1 - A_{12}) + E_2(t) + E_3(t)\cdot(1 - A_{23}) = 41.5$, resulting in a brown energy consumption of 28.5 units. By comparing the two solutions, we can observe that different VM migration decision will lead to different communication and VM migration energy consumption. Moreover, the location of VM might also affect energy consumption due to the location-related attenuation ratio during transmission. To address these issues, we are motived to jointly investigate the green energy scheduling and VM migration problem in green energy powered edge computing. Our main contributions are as follows:

- To the best of our knowledge, we are the first to investigate the VM migration and energy scheduling problem in green energy powered edge computing, jointly considering the time-varying green energy availability and user tasks.
- We formulate the VM migration and energy scheduling problem as a mixed integer linear programming (MILP) and formally prove its NP-hardness through reducing the generalized quadratic assignment problem.
- Based on our formulation, we design a heuristic relaxation-based algorithm and prove that it can achieve an approximation ratio of $\frac{K\cdot(1+\beta)-\alpha}{K-1}$ when the brown energy is needed or $\alpha\cdot(1+\beta)$ when no brown energy is needed. The efficiency and correctness of our proposal is validated through extensive simulation based experiments.

The remainder of the paper is organized as following. Section 2 introduces our system model. The energy consumption minimization problem is formulated into MILP in Section 3. Then we propose a heuristic relaxation-based algorithm in Section 4 and the effectiveness of proposed algorithm is verified by simulation experiments in Section 5. Finally, we discuss the related work in Section 6 and conclude our work in Section 7.

## 2. System model

In this section, we present the system model. The major notations used in this paper are listed in Table 1.

### 2.1. Network model

We use an indirected graph $G_n = (I, E)$ to present the distributed edge network, which includes a set of edge nodes $I$ and a set of network edges $E$. Each edge $e_{ij} \in E, i, j \in I$ is with a weight as $H_{ij}$, denoting the number of hops between nodes $i$ and $j$. Specially, we set the hops between the same node as 0, i.e., $H_{ii} = 0, \forall i \in I$. At each time slot $t$, the task arrival rate in node $i \in I$ is denoted as $\lambda_i(t)$. As mentioned in Section 1, the user tasks can only be processed in edge node with the corresponding VM with processing energy consumption of $E$. In this paper, we assume that there is only one VM in the network topology to process tasks. We define a binary

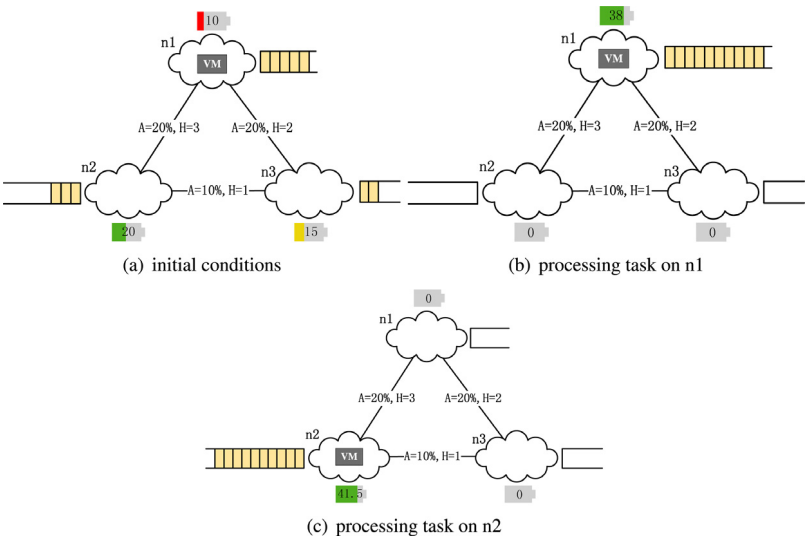



**Fig. 2.** An example on the task Dispatching, VM migration and energy scheduling.

**Table 1**
Notations.

| | |
|---|---|
| Constant | |
| $N$ | The set of edge nodes |
| $H_{ij}$ | The hops from nodes $i$ to $j$ |
| $A_{ij}$ | The energy transferring attenuation ratio from nodes $i$ to $j$ |
| $E$ | The energy consumption of task processing |
| $W$ | The energy consumption of task transmission |
| $V$ | The energy consumption of VM migration |
| $\lambda_i(t)$ | Newly arrival tasks in node $i$ at time slot $t$ |
| $G_i(t)$ | Green energy in node $i$ at time slot $t$ |
| Parameter | |
| $x_i(t)$ | A binary variable indicating whether node $i$ is selected for task processing or not |
| $\mu_{ij}(t)$ | The green energy transferred from node $i$ to $j$ at time slot $t$ |
| $g_i(t)$ | The green energy consumption for processing tasks in node $i$ at time slot $t$ |
| $s_i(t)$ | The total energy consumption for processing tasks in node $i$ at time slot $t$ |
| $r_i(t)$ | The remaining green energy in node $i$ at the end of time slot $t$ |

$x_i(t)$ to indicate whether node $i$ is selected to host the VM for task processing or not, i.e.,

$$x_i(t) = \begin{cases} 1, \text{ if VM shall be located in edge node } i \text{ at time slot } t, \\ 0, \text{ otherwise.} \end{cases}$$

Obviously, when $x_i(t) \neq x_i(t-1)$, VM migration incurs at time $t$, resulting in VM migration with energy consumption $V$.

### 2.2. Green energy model

We use a graph $G_e = (I, E)$ to present the green energy scheduling and transferring between the edge nodes, including a set of edge nodes $I$ and a set of edges $E$. The node set $I$ and edge set $E$ are the same as in graph $G_n$. Unlike graph $G_n$, each node $i \in I$ is with a parameter $G_i(t)$ denoting the green energy generated by node $i$ at time slot $t$. As mentioned in Section 1, the user tasks can only be processed in edge node $i$ where $x_i(t) = 1$ and the green energy can be transferred between edge nodes. At each time slot $t$, an energy scheduling decision must be made to transmit $\mu_{ij}$ units of green energy through edges $e_{ij} \in E, \forall i, j \in I$ between node $i$ and node $j$ with energy transferring attenuation ratio $A_{ij}$. Specially, we set the attenuation ratio between the same node as 0, i.e., $A_{ii} = 0, \forall i \in I$, for tractability of our analysis. Obviously, different edge nodes can provide different amount of green energy, and different green energy scheduling decisions may lead to different green energy loss.

## 3. Problem formulation

Based on our system model, we now formulate our VM migration and energy scheduling problem into a mixed-integer linear programming (MILP) with the objective of minimizing the energy cost.

### 3.1. VM migration and energy consumption

As shown in $G_n$ graph, the VM can be freely migrated between edge nodes and only the node with VM can be selected to process user tasks. We are interested in finding the appropriate VM migration decision to minimize the brown energy consumption. The VM shall be adaptively migrated among the edge nodes so as to cater to the spatial and time diverse green energy generation and task demands. As we consider one VM in the whole network, we must have

$$\sum_{i \in I} x_i(t) = 1. \tag{1}$$

to ensure the existence of the VM, regardless of the VM migration decision.

Since the number of arriving tasks and the amount of green energy generated in each node is time-varying. We make the VM migration and energy scheduling decision at the beginning of each time slot $t$. Note that, there is only one VM in the network topology, the tasks arrived at the nodes without VM must be transmitted to

the node with VM for processing. Hence, the total energy required for processing tasks includes three parts: processing tasks, VM migration and task transmission. The total energy consumption $s_i(t)$ can be described as:

$$s_i(t) = \sum_{j\in I} x_i(t)(\lambda_j(t)\cdot E + x_j(t-1)\cdot V\cdot H_{ji} + \lambda_j(t)\cdot W\cdot H_{ji}), \forall i \in I, \tag{2}$$

where $V$, $W$ and $E$ is the unit energy consumption of VM migration, task transmission, and task processing, respectively. It can be observed that (2) is nonlinear because of the products of integer variables as $x_i(t)\cdot x_i(t-1)$. To linearize this equation, we define new binary variables $z_i(t)$ as follows:

$$z_i(t) = x_i(t)\cdot x_j(t-1), \forall (i,j) \in I, \tag{3}$$

which can be equivalently replaced by the following linear constraints:

$$0 \le z_i(t) \le x_i(t), \forall (i,j) \in I. \tag{4}$$

$$x_i(t) + x_j(t-1) - 1 \le z_i(t) \le x_j(t-1), \forall (i,j) \in I \tag{5}$$

Now, (2) can be equivalently written into a linear form as:

$$s_i(t) = \sum_{j\in I}(x_i(t)\cdot\lambda_j(t)\cdot E + x_i(t)\cdot\lambda_j(t)\cdot W\cdot H_{ji} + z_i(t)\cdot V\cdot H_{ji}), \forall i \in I, \tag{6}$$

with constraints (4) and (5).

### 3.2. Green energy scheduling

The amount of newly generated green energy in each node at each time slot is different. For example, some nodes have sufficient green energy and some nodes are scarce. If the green energy amount on the selected node $i$ is not sufficient to process all tasks, certain energy may need to be transferred from the other nodes with residual green energy. Certain energy loss will be incurred due to the attenuation. It is necessary to determine the amount of energy transferring between different node pairs at the beginning of each time slot $t$, i.e., $\mu_{ji}(t), \forall j, i \in I$. With the consideration of energy transferring, the total green energy amount can obtained by node $i$can be calculated as:

$$g_i(t) = \sum_{j\in I}(1 - A_{ji})\cdot\mu_{ji}(t), \forall i \in I. \tag{7}$$

Since aggressively transferring green energy to node $i$ with VM might cause more green energy loss. Therefore, we constraint the green energy transmission by $s_i(t)$ as follows:

$$0 \le g_i(t) \le s_i(t), \forall i \in I. \tag{8}$$

That is, the maximum green energy obtained by the processing node cannot exceed its energy requirement for task processing as $s_i(t)$.

Based on the VM migration and energy transferring decisions, the green energy in each edge node might not be used up or fully transferred to the other nodes by the end of each time slot. In this case, the surplus green energy is stored and can be used together with the newly generated green energy in the next time slot. Let $r_j(t)$ be the surplus green energy in node $j$ at time slot $t$. We have

$$r_j(t) = r_j(t-1) + G_j(t) - \sum_{i\in I}\mu_{ji}(t), \forall j \in I. \tag{9}$$

Note that the green energy transferred from each node cannot exceed its total amount. Moreover, for energy efficiency concern, no green energy can be transferred to an edge node without the VM, i.e., $\mu_{ji}(t) = 0$ when $x_i(t) = 0$. Therefore $\mu_{ji}(t)$ is constrained by

$$0 \le \mu_{ji}(t) \le x_i(t)(r_j(t-1) + G_j(t)), \forall (i,j) \in I. \tag{10}$$

### 3.3. A joint MILP formulation

To achieve energy efficiency, we always use green energy to process tasks with a higher priority. When the green energy of all nodes run out, the brown energy consumption incurs. In this case, the brown energy consumption can be calculated as $s_i(t) - g_i(t)$. By summing up all above, the objective of the VM migration and energy scheduling problem is as follows:

$$\min : (\alpha\sum_{i\in I}(s_i(t) - g_i(t)) + \beta\sum_{i\in I}\sum_{j\in I}\mu_{ji}(t)), \tag{11}$$

where the coefficients $\alpha$ and $\beta$ are defined by users to balance the brown energy and green energy consumption. In this paper, our goal is to use green energy as much as possible to minimize the use of brown energy, hence $\alpha$ is set much larger than $\beta$. When there is sufficient green energy in the network, this problem is transformed into minimizing green energy consumption.

We consider a time period $T = \{1, 2, 3, \ldots, |T|\}$ divided into multiple discrete time slots and then make the time-slot decisions for VM migration and energy scheduling of each time slot $t \in T$. The VM location and energy amount at different time slots should be considered together and decisions at each time slot should be made based on the knowledge of previous and next time slots. For example, more green energy might be stored in current time slot and to be used later to avoid high green energy attenuation loss. By rewriting (11), the joint scheduling problem can be formulated as:

Cost-Min:

$$\min : \sum_{t\in T}(\alpha\sum_{i\in I}(s_i(t) - g_i(t)) + \beta\sum_{i\in I}\sum_{j\in I}\mu_{ji}(t)),$$

s.t. : (1), (4), (5), (6), (7), (8), (9) and (10).

### 3.4. Hardness proof

In this part, we prove the hardness of our problem by reducing it to the well-known general quadratic assignment problem (GQAP), which has been proved strongly NP-hard [17]. Given $K$ facilities and $M$ locations. A facility $k$ can only be assigned to one location $m$, but multiple facilities can be assigned to the same location. For each pair of locations $(m, n)$, a weight $H_{m,n}$ is specified as their distance. While for each pair of facilities $(k, p)$, a price $V_{k,p}$ is defined to denote the cost of workloads transported between them. The goal of GQAP is to assign all facilities to the locations with the minimum sum of the distances multiplied by the corresponding prices.

**Theorem 1.** *The brown energy cost minimization problem is NP-hard.*

**Proof.** In our study, we are interested to find the location to host the VM for task processing during the $|T|$ time slots, i.e., $x_i(1), x_i(2), \ldots, x_i(t), \ldots, x_i(|T|)$. As mentioned above, each pair of selected locations $i, j \in I$ is with a weight as $H_{ij}$. Moreover, once $x_i(t-1) \neq x_i(t)$, the VM image need to be transported to the next time slot location at a price of $V$ and corresponding task processing price of $\lambda_j(t)\cdot(W + E)$. Assume the energy can be transferred over links without attenuation, the original Cost-Min problem can be transformed to minimize the total required energy as the following Energy-Min problem:

Energy-Min:

$$\min : \sum_{t\in T}\sum_{i\in I}s_i(t),$$

s.t. : (1), (4), (5) and (6).

That is, we need to assign all $|T|$ VMs to $I$ edge nodes (VMs in different time slots can be located in the same nodes without migration), with the goal of minimizing the products of distance $H_{ij}$ multiplied by VM migration price $V$ and task processing price $\lambda_j(t) \cdot (W + E)$. This is a typical generalized quadratic assignment problem, which is NP-hard.

## 4. Algorithm design

In this section, we design a low-complexity heuristic algorithm for our problem. When the network topology becomes complex or the number of time slots $|T|$ increases, the computational complexity of solving Cost-Min problem directly will significantly increase because of the binary variables. Therefore, we relax the binary variables $x_i(t)$ into continuous ones in the range of [0, 1]. Then, the original Cost-Min problem is transformed into a linear programming (LP) problem which can be solved in polynomial-time by most commercial solver, such as Gurobi.

Once the solutions of the LP problem is obtained as the relaxed $x_i(t)$, we then sort them increasingly in line 3. The maximal $x_i(t), \forall i \in I$ is set to 1 while the others are set to 0, for all time slots $t \in T$ in lines 2 to 12. Next, we bring this values of $x_i(t)$ to the original Cost-Min problem and determine the values of $\mu_{ij}(t), \forall i, j \in I$ by any LP solver to find a feasible solution using the minimum brown energy. The details of our relaxation-based algorithm is shown in Algorithm 1.

**Algorithm 1** Relaxation-based Algorithm

**Require:**
Network Graph $G_n = (I, E)$, Green Energy Graph $G_e = (I, E)$

**Ensure:**
VM migration and green energy transferring between the edge nodes

1: Relax the integer variable $x_i(t)$, solve the Cost-Min-LP problem

$$\min : \sum_{t\in T}(\alpha \sum_{i\in I}(s_i(t) - g_i(t)) + \beta \sum_{i\in I}\sum_{j\in I}\mu_{ji}(t)),$$

s.t. : (1), (2), (7), (8), (9) and (10)

$$0 \le x_i(t) \le 1.$$

2: **for all** time $t \in T$ **do**
3: Sort $x_i(t), \forall i \in I$ decreasingly
4: $counter = 0$
5: **for all** $i \in I$ **do**
6: **if** $counter < 1$ **then**
7: $counter++, x_i(t) = 1$
8: **else**
9: $x_i(t) = 0$
10: **end if**
11: **end for**
12: **end for**
13: Take $x_i(t)$ into the Cost-Min, and obtain the energy transmission $\mu_{ij}(t)$

It can be observed that Algorithm 1 can get the solution of Cost-Min with a low computational complexity. We first sort all $x_i^t$ after relaxation and assign the value 0 and 1 to the sorted $x_i^t$ in each time slot as shown in lines 3 to 12 with the complexity of $\emptyset(logN + N)$. Then we repeat the sort and assignment of $x_i^t$ at all time slots $t \in T$ with the complexity of $O(T)$. Hence, the complexity of Algorithm 1 is $O(TN)$.

### 4.1. Approximation ratio

Suppose in time $T$, there are totally $M$ tasks and $G$ green energy. Tasks and VM need to be migrated to the selected node in every time slot. $H_{max}$ and $H_{min}$ are the maximum and minimum network hops between 2 nodes. In the optimal case, the network hops of VM migration or tasks migration is larger than or equal to $H_{min}$. We denote $C^*$ as the optimal total energy cost and we have,

$$C^* \ge M \cdot E + V \cdot H_{min} + T \cdot H_{min} \cdot M. \tag{12}$$

In our solution, the total energy cost $C$ can be calculated as:

$$C \le M \cdot E + V \cdot H_{max} + T \cdot H_{max} \cdot M. \tag{13}$$

Combining (12) and (13), we have total energy cost approximation ratio as following:

$$\begin{aligned} \frac{C}{C^*} &\le \frac{M \cdot E + V \cdot H_{max} + T \cdot H_{max} \cdot M}{M \cdot E + V \cdot H_{min} + T \cdot H_{min} \cdot M} \\ &\le \frac{M \cdot E}{M \cdot E} + \frac{(V + T \cdot M) \cdot H_{max}}{(V + T \cdot M) \cdot H_{min}} \\ &= 1 + \frac{H_{max}}{H_{min}} \end{aligned} \tag{14}$$

That is,

$$C \le (1 + \frac{H_{max}}{H_{min}}) \cdot C^* \tag{15}$$

**Proof.** We set $a, b, c, d$ as 4 non-zero positive numbers, and obviously we have

$$\frac{c+d}{a+b} = \frac{c}{a+b} + \frac{d}{a+b} \le \frac{c}{a} + \frac{d}{b}.$$

Assume that the optimal solution need to transfer $G_o$, $G_o \le G$ green energy and our solution need to transfer $G_r$, $G_r \le G$ green energy during the entire period time $T$. $A_{max}$ and $A_{min}$ are the maximum and minimum attenuation ratios between the edge nodes. In the optimal case, the attenuation ratio of green energy transmission is larger than or equal to $A_{min}$. $G^*$ and $G$ units of energy is obtained for task processing for the optimal and our solution, respectively Then, we have:

$$G^* \le (1 - A_{min})G_o, \tag{16}$$

and

$$G \ge (1 - A_{max})G_r. \tag{17}$$

The approximate ratio of our solution then can be calculated in two different occasions: with and without sufficient green energy. When green energy can support the total energy demand and therefore no brown energy will be consumed. In this case, we usually have $G_o \ne G_r$, and the amount of green energy obtained on the selected node equals to the total energy amount to VM migration, task transmission and processing. According to (16) and (17), the approximation ratio in this case is

$$\begin{aligned} \frac{G_r}{G_o} &\le \frac{1 - A_{min}}{1 - A_{max}} \cdot \frac{G}{G^*} \\ &\le \frac{1 - A_{min}}{1 - A_{max}} \cdot \frac{C}{C^*} \\ &= \frac{1 - A_{min}}{1 - A_{max}} \cdot (1 + \frac{H_{max}}{H_{min}}). \end{aligned} \tag{18}$$

On the other hand, we need to consume brown energy when green energy runs out, i.e., $G_o = G_r$. Considering (16) and (17), we have

$$\begin{aligned} \frac{G}{G^*} &\ge \frac{(1 - A_{max})G_r}{(1 - A_{min})G_o}, \\ &= \frac{1 - A_{max}}{1 - A_{min}}. \end{aligned} \tag{19}$$

That is,

$$G \geq \frac{1 - A_{max}}{1 - A_{min}} \cdot G^* \tag{20}$$

From (15) and (20), we can calculate the approximation ratio with brown energy consumption as

$$\begin{aligned} \frac{C - G}{C^* - G^*} &\leq \frac{(1 + \frac{H_{max}}{H_{min}}) \cdot C^* - \frac{1-A_{max}}{1-A_{min}} \cdot G^*}{C^* - G^*} \\ &= 1 + \frac{H_{max}}{H_{min}} + \frac{(1 + \frac{H_{max}}{H_{min}} - \frac{1-A_{max}}{1-A_{min}}) \cdot G^*}{C^* - G^*} \end{aligned} \tag{21}$$

Let $\alpha = \frac{1-A_{min}}{1-A_{max}}$, $\beta = \frac{H_{max}}{H_{min}}$ and $K = \frac{C^*}{G^*}$, and the approximation ratio $\Gamma$ can be represented as

$$\Gamma = \begin{cases} \dfrac{K \cdot (1 + \beta) - \alpha}{K - 1}, & \text{without sufficient green energy,} \\ \alpha \cdot (1 + \beta), & \text{with sufficient green energy.} \end{cases}$$

## 5. Performance evaluation

### 5.1. Simulation results

In this section, we compare our relaxation-based algorithm ("MTS") with four competitors, as "OPT", "OWN", "OTS" and "ATS". The OWN algorithm uses only local green energy without any energy transferring from other nodes. The OTS algorithm optimizes the energy consumption of each time slot $t$ separately and the ATS algorithm schedules the energy based on the average network states over period $T$.

In our experiments, we consider a network based on the topology of Abilene US Continental Network with 11 nodes connected by 14 links to mimic an edge computing environment. The number of hops between two nodes is randomly set within range [1, 8] and the corresponding attenuation ratio between any node pair is set within range [0.01, 0.5]. We divide the time period $T$ into 10 time slots, and the task arrival rate and green energy generate rate on each edge node at each time slot are randomly set within the ranges of [0, 20] and [0, 500], respectively. The energy consumption of task processing, transmission and VM migration are 10, 3 and 10, respectively.

First of all, we investigate the performance under different settings of task processing energy consumption $E$. As the value of $E$ increases from 3 to 30, we can see from Fig. 3 that the brown energy consumption of all five algorithms grows with $E$. It is because that with the same task arrival, the total energy consumption shall grow with the task processing energy consumption $E$. It can be observed that OWN algorithm always gives the worst energy efficiency because it only uses the local green energy. It is also worthy noticing that the energy consumption of both OTS and ATS increases much faster than our MTS algorithm. The reason is that MTS takes into account of multiple $t$ and therefore is able to make more fine-grained green energy scheduling more effectively. Fig. 4 shows the detailed energy consumption of our MTS algorithm by classifying it into green energy consumption, transferring loss due to attenuation, task transmission energy consumption (i.e., offloading) and VM migration energy consumption. It can be observed that when $E$ is larger than 15, the green energy is insufficient and more brown energy is needed. In this situation, the green energy consumption as well as the VM migration consumption will not change, as shown in Fig. 4. Similarly, by increasing the value of task transmission energy consumption $W$ from 2 to 20, we can see that the total energy consumption in Fig. 5 and task transmission energy in Fig. 6 also increase. This is simply because higher unit task transmission energy consumption definitely requires more

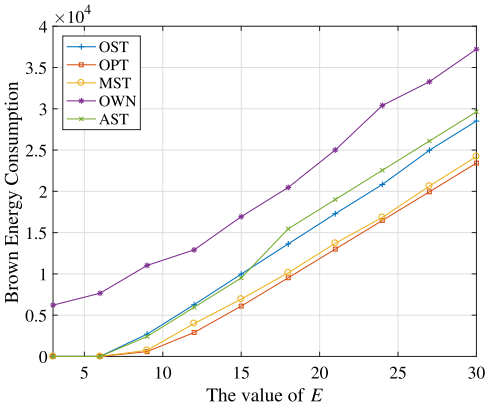



**Fig. 3.** The effect of $E$.

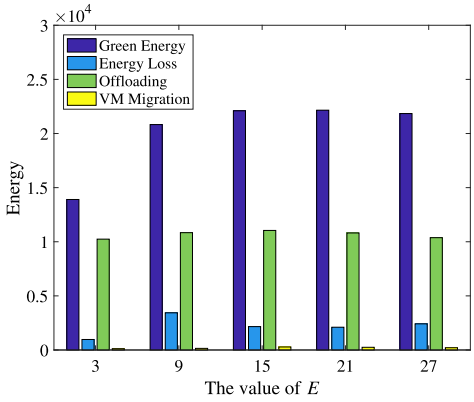



**Fig. 4.** Energy consumption of MTS under different $E$.

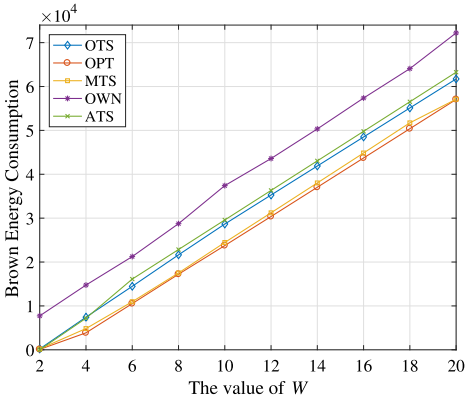



**Fig. 5.** The effect of $W$.

task transmission energy, and hence the total brown energy consumption. Nevertheless, with careful scheduling of the VM location and green energy transferring, our MTS algorithm can still reserve the best energy efficiency.

Then, we study the energy consumption of all five algorithms with different values of $\lambda$ with upper bound of $\lambda_i(t)$ increasing from 12 to 48 to check how these algorithms adapt to different task workloads. The experiment results are reported in Fig. 7. We can see that the total brown energy consumption shows as an increasing function of the task arrival rate, for any algorithm. Higher task workloads imply more energy consumption. When the green energy is sufficient to support all the task processing, no brown energy is needed and the green energy consumption

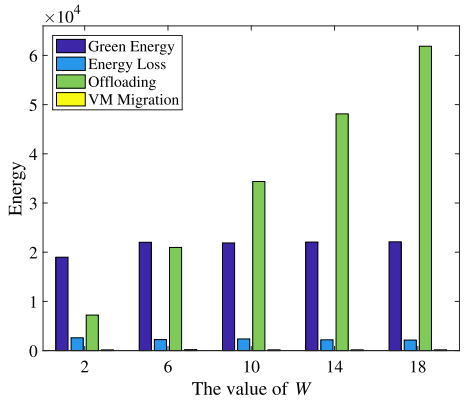



**Fig. 6.** Energy consumption of MTS under different $W$.

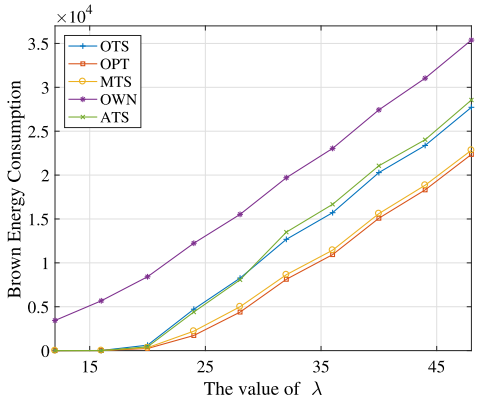



**Fig. 7.** The effect of λ.

increases, as shown in Fig. 8. However, when $\lambda_i(t)$ reaches 20, brown energy becomes needed, and the green energy consumption remains constant. This is because all the green energy is completely used and we still need to resort to brown energy such that all the tasks can be completely processed. Similar observation can be obtained from Fig. 9 when the upper bound of transmission distance $H$ increases from 3 to 25. One interesting phenomena is that, with the increase of $H$, the energy loss decreases in Fig. 10. That is because with the potential higher energy transferring cost, our MTS solution tries to reduce the energy transferring by carefully selecting the VM location.

Next, we vary the amount of available green energy generation rate $G_i(t)$ from 0 to 50 to check how our algorithm adapts to different green energy generation conditions. As shown in Fig. 11, we can see that the brown energy consumption decreases with the increase of green energy generation rates. This is attributed to the fact that, for either algorithm, we always use the green energy with higher priority in order to reduce the brown energy consumption. Thanks to the well scheduling of VM placement and task allocation, our algorithm always achieves the best energy efficiency. Meanwhile, from Fig. 12, we can see that the energy loss also increases with the green energy generation rates. This is because our algorithm always tries to transfer more green energy with residual energy to the server where the VM is placed. By such means, our algorithm can maximize the usage of green energy and minimize the brown energy consumption.

Finally, let us check how these algorithms adapt to different values of energy attenuation ratio by varying the upper bound of $A_{ij}, \forall i, j \in I$ from 0.1 to 1 (see Fig. 13). First, we can see that the brown energy always keeps the same value for "OWN" algorithm since it only use the local generated green energy. No energy transferring is required for "OWN" algorithm. However, for the other algorithms, we can see that the brown energy consumption increases with the value of green energy generation. Without doubt more brown energy shall be consumed if less green energy can be efficiently obtained by the server where the VM is placed. Once again, we can see that our MTS algorithm always achieves the best energy efficiency thanks to the maximal usage of green energy. Meanwhile, we can also see from Fig. 14 that the energy loss due to energy transferring attenuation also increases the attenuation ratio.

### 5.2. Discussion

In our MTS algorithm above, we averagely divide the time period $T$ into multiple time slots $t$. Here, we consider two interesting issues: (1) How many time slots should $T$ be divided? (2) How long should each time slot be?

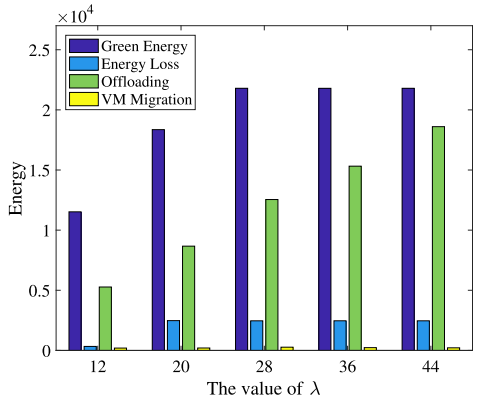



**Fig. 8.** Energy consumption of MTS under different λ.

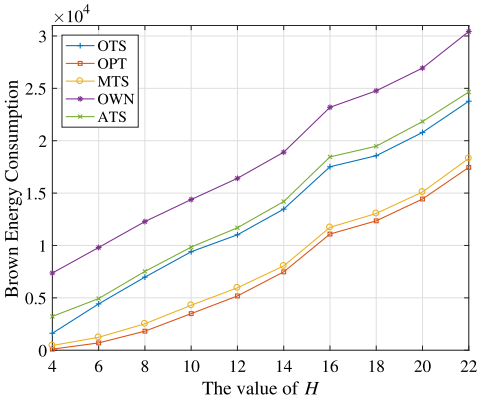



**Fig. 9.** The effect of $H$.

We first divide $T$ into 3 time slots, 6 time slots and 9 time slots, respectively, and vary the value of unit task processing energy consumption from 5 to 25. The results are reported in Fig. 15, from which we notice that the results of 3 time slots are always worse than the others. When the total time is divided into 9 time slots, it performs the best. From such phenomenon, it seems that we shall try to minimize the frequency of scheduling at runtime. However, frequent scheduling incurs lots of overhead. This reminds us another solution that, other than normally invoking the scheduling algorithm in a constant interval, we can adaptive invoke the scheduling algorithm according to realtime condition, e.g., task arrival rate. To this end, we implement another algorithm that adaptively invoke our MTS algorithm by checking the task arrival

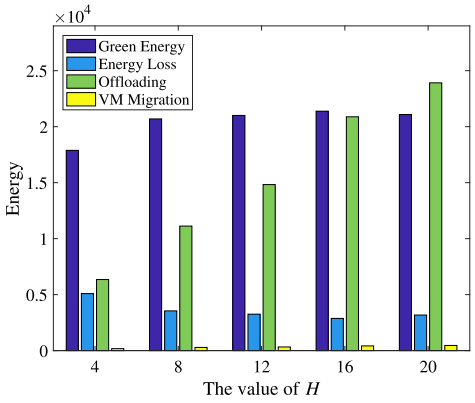



**Fig. 10.** Energy consumption of MTS under different $H$.

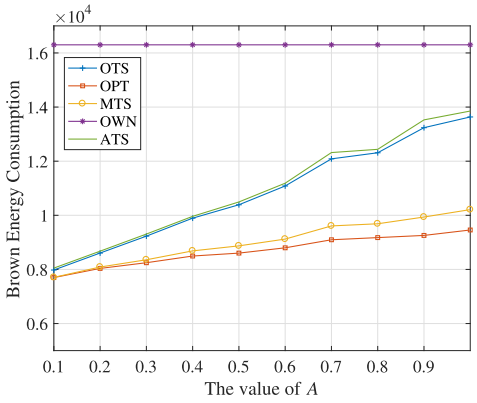



**Fig. 13.** The effect of $A$.

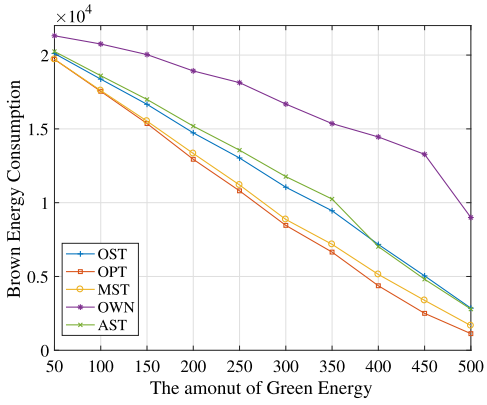



**Fig. 11.** The effect of $G_i(t)$.

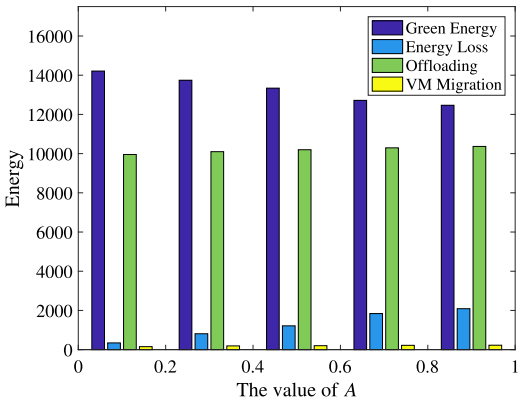



**Fig. 14.** Energy consumption of MTS under different $A$.

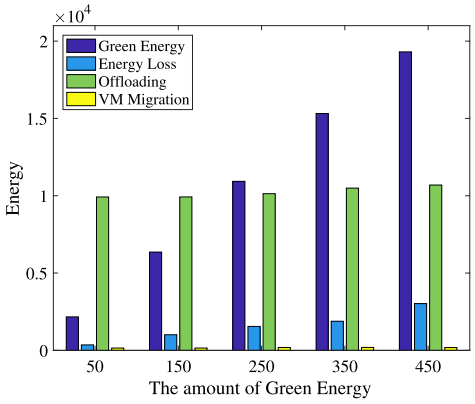



**Fig. 12.** The energy consumption of MTS under different $G_i(t)$.

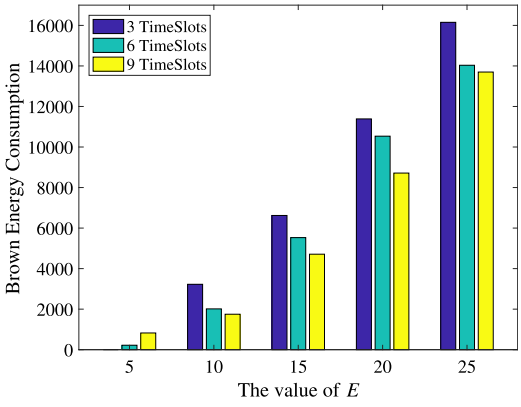



**Fig. 15.** The effect of the number of $t$.

difference. We call it as "Adaptive" in this section. In "Adaptive" algorithm, we set a threshold $\gamma$ and the MTS algorithm is invoked whenever the task arrival rate ratio, either increase or decrease, between two time slots is higher than $\gamma$. Therefore, different values of $\gamma$ shall exhibit different performance. We investigate its performance by varying its value from 0.1 to 0.5 and show the results in Fig. 16.

We observe that the value of $\gamma$ indeed have deep influence on the energy efficiency. When $\gamma$ is small, MTS algorithm will be invoked frequently. The interesting thing is that, the brown energy consumption first shows as a decreasing function of $\gamma$ and then increases with $\gamma$. We obtain the best energy efficiency when $\gamma = 0.3$. When $\gamma$ is small, e.g., $\gamma = 0.1$, the MTS algorithm will be invoked frequently, inevitably resulting in VM migration and hence the energy consumption. While, if $\gamma$ is large, it fails to track the dynamics of both the task arrival and green energy generation. Hence, the brown energy begins to increase. As both VM migration and task offloading are energy consuming, we are also interested to know their impact on the energy efficiency. We vary the value of $V/W$ and set $\gamma$ as 0.2. The performance of

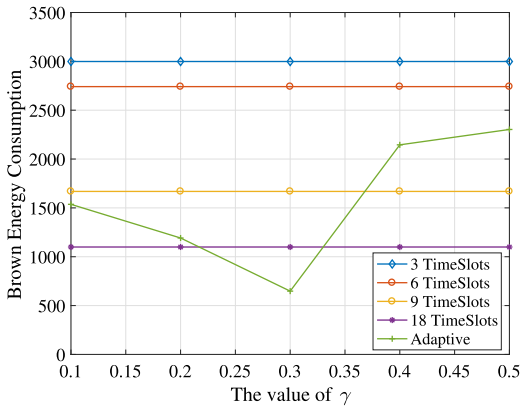



**Fig. 16.** The effect of $\gamma$.

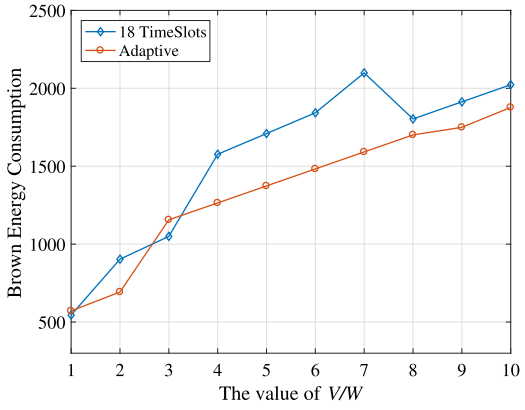



**Fig. 17.** The effect of the radio of $V$ and $W$.

"Adaptive" in comparison of the case when the total time is divided into 18 time slots is shown in Fig. 17. We notice that when $V/W$ is small, both algorithm performs similarly. However, when $V/W$ becomes large, "Adaptive" begins to outperform "18 TimeSlots". This is because when the VM migration cost is high, frequently migrating the VM may incurs unnecessary energy consumption. It would be better adaptively invoke the MTS algorithm according to the runtime condition.

## 6. Related work

### 6.1. Edge computing

As the demand of computing growing, the heavy workload and high energy consumption become an imperative problem [18]. The promising potentials of edge computing have attracted lots of interests from both academia and industry. In this section, we summarize some representative related work. Bonomi et al. [11] show the advantages of edge as the appropriate platform for the latency-sensitive services and applications, owing to its characteristics of energy efficiency, low latency, location awareness and geographical distribution. Jalali et al. [19] compare the energy consumption of edge node and data center under different conditions and show the advantages of edge computing in energy saving and latency reduction. To further study the workload offloading problem, Deng et al. [20] decompose the total energy consumption into edge, cloud and communication parts, and investigate how to allocate the workload for balanced delay and energy consumption. Moreover, edge nodes are geo-distributed and can make full use of local harvested green energy. For example, He et al. [21] investigate the possibility of introducing green energy to edge devices like access points to support user task computation with the goal of minimizing the grid power consumption. To provide an efficient energy management, Li et al. [22] propose a unified framework for sustainable edge computing with distributed green energy resources. Chen et al. [23] further study the multi-user and multi-task computation offloading problem for green energy supported edge computing, and adopt Lyapunov optimization to make energy harvesting decisions.

Present energy scheduling work in cloud or edge computing only focuses on local energy management, i.e., using its own energy in each node. These solutions neglect the large amount of available energy in neighbor nodes and result in a comparatively low global energy utility.

### 6.2. Energy internet

To address the energy efficiency issue, energy internet (EI) is proposed to solve energy crises by scheduling and transferring energy in a global and centralized way [12,24]. Wang et al. [12] introduce the advantages of EI and propose an efficient EI architecture for energy transfer in the power grid. The key device to compromise EI is an energy router (ER) with different scheduling strategies for energy scheduling and transfer. To improve the stability of energy scheduling, Gao et al. [25] propose a Markov decision process model based on an ER subsystem and the energy scheduling. Wang et al. [26] further propose a system-level stability evaluation model based on the energy function to explore small disturbance stability region. To maintain the reliability of the smart grid, Xie et al. [27] propose a contribution-based fairness energy scheduling, that is the high contributions which discharging during the load peak hours have high priorities to obtain charge energy. Zhong et al. [28] propose two auction mechanisms which are designed under the day-ahead and real-time markets for energy trading in a smart multi-energy district to maximize benefits of green energy. Kang et al. [29] propose a localized peer-to-peer (P2P) electricity trading model for locally buying and selling electricity in smart grids and use the blockchain technology to improve transaction security.

The great significance of EI has been validated by existing work above. To efficiently introduce EI into edge computing, we investigate the global energy scheduling problem in green energy supported edge computing.

## 7. Conclusion

In this paper, we investigate the VM migration, task allocation and the energy scheduling problem with the goal of brown energy consumption minimization. With the consideration of both green energy supply and user demand diversity, a discrete time-slotted scheduling optimization model is proposed. We prove the NP hardness of the problem and invent a relaxation-based heuristic algorithm to tackle the unpredictable green energy and user demand dynamics. Theoretical analysis shows that our proposal can achieve close-to-optimum solution and significantly reduce the brown energy consumption. Extensive simulation-based experiments are conducted to validate the correctness and energy efficiency of our algorithm. We also discuss the effect of the time slot duration to the energy efficiency. The results show that adaptively invoking the our scheduling algorithm according to the fluctuations of tasks and green energy shall have better performance in most cases.

## Acknowledgments

This research is supported by National Key Research and Development Program of China (Grant No. 2016YFB1000501), and the National Natural Science Foundation of China (Grant No. 61602199, 61772480, 61732010, 61602200).

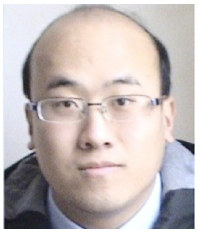
## References

[1] R. Lu, H. Zhu, X. Liu, J.K. Liu, J. Shao, Toward efficient and privacy-preserving computing in big data era, IEEE Netw. 28 (4) (2014) 46–50.
[2] N. Kumar, S. Misra, J.J.P.C. Rodrigues, M.S. Obaidat, Coalition games for spatio-temporal big data in internet of vehicles environment: A comparative analysis, IEEE Internet Things J. 2 (4) (2015) 310–320.
[3] C. Lai, R. Lu, D. Zheng, H. Li, X. Shen, Toward secure large-scale machine-to-machine comm unications in 3GPP networks: challenges and solutions, IEEE Commun. Mag. 53 (12) (2015) 12–19.
[4] Z. Li, Y. Liu, A. Liu, S. Wang, H. Liu, Minimizing convergecast time and energy consumption in green internet of things, IEEE Trans. Emerg. Top. Comput. (2018) http://dx.doi.org/10.1109/TETC.2018.2844282, 1–1.
[5] M. Huang, A. Liu, N.N. Xiong, T. Wang, A.V. Vasilakos, A low-latency communication scheme for mobile wireless sensor control systems, IEEE Trans. Syst. Man Cybern.: Syst. (2018) 1–16.
[6] H.T. Dinh, C. Lee, D. Niyato, P. Wang, A survey of mobile cloud computing: architecture, applications, and approaches, Wirel. Commun. Mob. Comput. 13 (18) (2013) 1587–1611.
[7] X. Tao, K. Ota, M. Dong, H. Qi, K. Li, Performance guaranteed computation offloading for mobile-edge cloud computing, IEEE Wirel. Commun. Lett. 6 (6) (2017) 774–777.
[8] X. Wang, L.T. Yang, X. Xie, J. Jin, M.J. Deen, A cloud-edge computing framework for cyber-physical-social services, IEEE Commun. Mag. 55 (11) (2017) 80–85.
[9] S. Akoush, R. Sohan, A. Rice, A.W. Moore, A. Hopper, Free lunch: Exploiting renewable energy for computing, in: Proceedings of ACM USENIX Conference on Hot Topics in Operating Systems, 2011, pp. 17–21.
[10] Y. Li, A.-C. Orgerie, I. Rodero, M. Parashar, J.–M. Menaud, Leveraging renewable energy in edge clouds for data stream analysis in iot, in: Proceedings of IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2017, pp. 186–195.
[11] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proceedings of ACM MCC Workshop on Mobile Cloud Computing, 2012, pp. 13–16.
[12] K. Wang, J. Yu, Y. Yu, Y. Qian, D. Zeng, S. Guo, Y. Xiang, J. Wu, A survey on energy internet: Architecture, approach, and emerging technologies, IEEE Syst. J. pp (99) (2017) 1–14.
[13] D. Xu, Q. Li, Price-based time and energy allocation in cognitive radio multiple access networks with energy harvesting, Sci. China Inf. Sci. 60 (10) (2017) 1–3.
[14] S. Maharjan, Q. Zhu, Y. Zhang, S. Gjessing, T. Basar, Dependable demand response management in the smart grid: A Stackelberg game approach, IEEE Trans. Smart Grid 4 (1) (2013) 120–132.
[15] W. Zhong, K. Xie, Y. Liu, C. Yang, S. Xie, Topology-aware vehicle-to-grid energy trading for active distribution systems, IEEE Trans. Smart Grid pp (99) (2018) 1–11.
[16] H. Jiang, K. Wang, Y. Wang, M. Gao, Y. Zhang, Energy big data: A survey, IEEE Access 4 (2016) 3844–3861.
[17] C.-G. Lee, Z. Ma, The generalized quadratic assignment problem, 2004.
[18] X. Ge, J. Chen, C.-X. Wang, J. Thompson, J. Zhang, 5g green cellular networks considering power allocation schemes, Sci. China Inf. Sci. 59 (2) (2016) 1–14.
[19] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, R.S. Tucker, Fog computing may help to save energy in cloud computing, IEEE J. Sel. Areas Commun. 34 (5) (2016) 1728–1739.
[20] R. Deng, R. Lu, C. Lai, T.H. Luan, H. Liang, Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption, IEEE Internet Things J. 3 (6) (2016) 1171–1181.
[21] X. He, Y. Chen, K.K. Chai, Delay-aware energy efficient computation offloading for energy harvesting enabled fog radio access networks, in: 2018 IEEE 87th Vehicular Technology Conference (VTC Spring), 2018, pp. 1–6.
[22] W. Li, T. Yang, F.C. Delicato, P.F. Pires, Z. Tari, S.U. Khan, A.Y. Zomaya, On enabling sustainable edge computing with renewable energy resources, IEEE Commun. Mag. 56 (5) (2018) 94–101.
[23] W. Chen, D. Wang, K. Li, Multi-user multi-task computation offloading in green mobile edge cloud computing, IEEE Trans. Serv. Comput. (2018) 1–1.
[24] K. Wang, Y. Wang, X. Hu, Y. Sun, D.J. Deng, A. Vinel, Y. Zhang, Wireless big data computing in smart grid, IEEE Wirel. Commun. 24 (2) (2017) 58–64.
[25] M. Gao, K. Wang, L. He, Probabilistic model checking and scheduling implementation of energy router system in energy internet for green cities, IEEE Trans. Ind. Inf. 14 (4) (2018) 1501–1510.
[26] K. Wang, H. Li, Y. Feng, G. Tian, Big data analytics for system stability evaluation strategy in the energy internet, IEEE Trans. Ind. Inf. 13 (4) (2017) 1969–1978.
[27] S. Xie, W. Zhong, K. Xie, R. Yu, Y. Zhang, Fair energy scheduling for vehicle-to-grid networks using adaptive dynamic programming, IEEE Trans. Neural Netw. Learn. Syst. 27 (8) (2016) 1697–1707.
[28] W. Zhong, K. Xie, Y. Liu, C. Yang, S. Xie, Auction mechanisms for energy trading in multi-energy systems, IEEE Trans. Ind. Inf. 14 (4) (2018) 1511–1521.
[29] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, E. Hossain, Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchain, IEEE Trans. Ind. Inf. 13 (6) (2017) 3154–3164.

**Lin Gu** received her M.S. and Ph.D. degrees in computer science from University of Aizu, Fukushima, Japan in 2011 and 2015. She is currently a lecturer in School of Computer Science and Technology, Huazhong University of Science and Technology, China. She is a member of IEEE. Her current research interests include cloud computing, vehicular cloud computing, Big Data and Software-defined Networking.

**Jingjing Cai** currently is a master student in School of Computer Science and Technology, Huazhong University of Science and Technology, China. She graduated from Northeastern University and obtained the B.S. degree in 2016. Her current interests mainly focus on the resource management in fog computing.

**Deze Zeng** is currently a full professor in School of Computer Science, China University of Geosciences, Wuhan, China. He received his Ph.D. and M.S. degrees in computer science from University of Aizu, Aizu-Wakamatsu, Japan, in 2013 and 2009, respectively. He received his B.S. degree from School of Computer Science and Technology, Huazhong University of Science and Technology, China in 2007. His current research interests include: network function virtualization, software-defined networking, cloud computing and edge computing. He has authored 1 book and over 80 papers in refereed journals and conferences in these areas. He also received 3 best paper awards from IEEE/ACM conferences and the IEEE Systems Journal Annual Best Paper Award of 2017. He serves in editorial boards of Journal of Network and Computer Applications and guest editors of many prestigious journals. He has been the in organization or program committees of many international conferences including ICPADS, ICA3PP, CollaberateCom, MObiQuitous, ICC, Globecom. He is a member of IEEE.



**Yu Zhang** is now an assistant professor in computer science and technology of Huazhong University of Science and Technology (HUST), Wuhan, China. His research interests include big data processing, cloud computing and distributed systems. His current topic mainly focuses on application-driven big data processing and optimizations.

**Hai Jin** is a Cheung Kung Scholars Chair Professor of computer science and engineering at Huazhong University of Science and Technology (HUST) in China. He is now Dean of the School of Computer Science and Technology at HUST. Jin received his PhD in computer engineering from HUST in 1994. In 1996, he was awarded a German Academic Exchange Service fellowship to visit the Technical University of Chemnitz in Germany. Jin worked at The University of Hong Kong between 1998 and 2000, and as a visiting scholar at the University of Southern California between 1999 and 2000. He was awarded Excellent Youth Award from the National Science Foundation of China in 2001. Jin is the

chief scientist of ChinaGrid, the largest grid computing project in China, and the chief scientists of National 973 Basic Research Program Project of Virtualization Technology of Computing System, and Cloud Security. Jin is a senior member of the IEEE and a member of the ACM. He has co-authored 15 books and published over 500 research papers. His research interests include computer architecture, virtualization technology, cluster computing and cloud computing, peer-to-peer computing, network storage, and network security.

**Weiqi Dai** received the Ph.D. degrees in computer science from the Huazhong University of Science and Technology (HUST), China, in 2014, respectively. He is currently an Assistant Professor with HUST. His research interests are mainly about blockchain, cloud security, trusted computing, system security and virtualization.