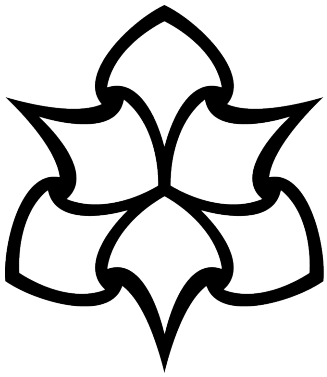

# Context-Aware Task Offloading for Multi-Access Edge Computing: Matching with Externalities

Bo Gu*, Zhenyu Zhou†, Shahid Mumtaz‡, Valerio Frascolla§ and Ali Kashif Bashir¶
*Kogakuin University, Tokyo 192-0015 Japan, Email: bo.gu@cc.kogakuin.ac.jp
†North China Electric Power University, Beijing 102206, China, Email: zhenyu_zhou@ncepu.edu.cn
‡Institute of Telecommunications, Lisbon, 1049-001, Portugal, Email: smumtaz@av.it.pt
§Intel Deutschland GmbH, Neubiberg, 85579, Germany, Email: valerio.frascolla@intel.com
¶University of the Faroe Islands, Faroe Islands, Denmark, Email: dr.alikashif.b@ieee.org

***Abstract*—Multi-Access Edge Computing (MEC) is an emerging technology that leverages computing, storage and network resources deployed at the proximity of users to offload terminal from computational- and delay-sensitive tasks. Various existing facilities including mobile devices with idle resources, vehicles, and MEC servers deployed at base stations or road side units, could act as edges in the network. Since offloading tasks incurs extra transmission energy consumption and transmission latency, two key questions to be addressed in MEC deployments are: (i) offload the workload to the edge or compute it in terminals? (ii) which edge, among the available ones, should the task be offloaded to? Hence, we propose a matching theory based task assignment mechanism which takes into account the devices' and MEC servers' computation capabilities, wireless channel conditions, and delay constraints. The main goal of our task assignment mechanism is to reduce overall energy consumption, while satisfying task owners' heterogeneous delay requirements and supporting good scalability. Simulations are conducted to evaluate the efficiency of our proposed mechanism.**

## I. Introduction

The explosive growth of information and communication technologies spurs an array of computation-intensive applications such as Augmented Reality (AR) or online gaming, and allows new functionalities, like self-driving vehicles, with the final aim of enriching our lives. However, despite the great advances that might be brought by these emerging applications, several challenging research issues are still unsolved and need to be addressed.

Computation-intensive applications generally require high computing capacity for data processing, which cannot be easily offered by mobile terminals (e.g., smart phones and wearable devices) due to their limited resources. Cloud computing [1] enables on-demand network access to a shared pool of configurable computing resources, which can largely augment on-device computing capacity. By 2021, it is predicted that 94% of total workloads and compute instances will be processed in the cloud, and that the annual IP traffic originated or terminated at data centers will reach 20.6 Zettabytes (ZB) [2]. However, the transmission of such huge volume of data to cloud data centers will not only pose a heavy burden on the capacity-constrained backhaul and backbone networks, but also results in unpredictable transmission latency and degraded Quality of Service (QoS) to end users. To this end, Multi-Access Edge Computing (MEC) [3] is proposed as a promising paradigm to address the latency issue by switching from the conventional centralized cloud computing architecture to a distributed one [4]. Various existing facilities such as mobile devices with idle resources, vehicles and MEC servers deployed at base stations or road side units, could act as distributed edges. Offloading computation tasks to such edges that are physically closer to the data sources could significantly reduce transmission latency and hence improve the QoS perceived by end users.

In this paper, we formulate the task assignment in MEC to a one-to-many matching problem by taking into account the devices' and MEC servers' computation capabilities, wireless channel conditions, and delay constraints. The main goal of our task assignment mechanism is to reduce overall energy consumption, while satisfying task owners' heterogeneous delay requirements and supporting good scalability. By dispatching tasks to edges carefully, our proposed mechanism can significant reduce the overall energy consumption, and provides a good compromise between computation complexity and energy consumption optimization.

The rest of this paper is organized as follows. In Section II, we give a brief literature review. In Section III, we present our problem formulation. In Section IV, we propose a matching-theory based solution and analyze its stability. In Section V, we conduct simulations to confirm the efficiency of our proposed mechanism. Finally, in Section VI, we conclude our paper.

## II. Related Work

As one of the most promising technologies for improving computing experience of end users, MEC has attracted considerable attentions from both industry and academia since its inception. From the industry side, in 2013, IBM and Nokia Siemens Networks announced the very first MEC platform named Radio Applications Cloud Server (RACS), which could run applications directly within a mobile base station. But the real momentum was achieved in 2014, when an Industry Specification Group (ISG) focusing on MEC was started in ETSI to create consensus within the ecosystem on a first set of MEC specifications, and since then several companies have proposed semi-proprietary solutions. Arm's Mbed Edge, offers a transparent distributed application execution environment to enable processing of rules and data on gateways, thus enabling

978-1-5386-4727-1/18/$31.00 ©2018 IEEE

a wide range of IoT devices to be monitored and controlled remotely via gateways. Microsoft's Azure IoT Edge further integrates machine learning and artificial intelligence (AI) at gateways, which allows deploying advanced analytics, event processing, and image recognition services without accessing the cloud. Intel has been since the beginning of the ETSI MEC ISG work one of the key contributors and is very active in delivering cutting edge solutions for network edge virtualization, offering the first kit of its kind to provide a Network Function Virtualization (NFV) platform targeted for MEC application and services.

Many research efforts have focused on MEC as well. In [5], it is pointed out that providing virtualized resources and localized services to the edge of mobile networks can better cater to the versatility of requirements for end users. In [6], several MEC based use-cases are presented, including video analytics, smart home, smart city and collaborative edge. In, [7], the benefits of integrating MEC into the 5G ecosystem are demonstrated and the key technical challenges are discussed, e.g., in terms of resource management and mobility support. In [8], a MEC based solution is proposed to solve the fact that cloud resources can be utilized to process the medical sensor data while the unstable and high-latency links between the cloud and medical smart devices impedes the development of medical cyber-physical systems (MCPSs). In [9], MEC is deployed in vehicular ad hoc networks (VANETs), thus proposing an advanced framework for VANETs that integrates the flexibility, scalability, and programmability of Software Defined Networking (SDN) [10] and the low latency, mobility support, and location awareness enabled by MEC. In [11], MEC is merged with other 5G related concepts, like millimeter waves and NFV, so to stress their potentiality to create new business models for the whole communication ecosystem. In [12], authors focus on addressing the link breakage issues due to user mobility in MEC architectures. In [13], the concept of Fog of Everything (FoE) is introduced, which exploits the advantages of both the edge computing and Internet of Everything (IoE).

Another thread of research has been focusing on reducing energy consumption [14], [15], [16] in MEC architectures. In [17], authors propose an energy-efficient computation offloading mechanism, which jointly optimizes task offloading and radio resource allocation so as to achieve minimum energy consumption under delay constraints. In [18], the tradeoff between power consumption and transmission delay in the fog-cloud computing system is investigated. A workload allocation problem is formulated by taking into account the power consumption and service delay constraints, and an approximate approach based optimal workload allocation policy is proposed. In [19], authors present a Markov decision process approach for optimally scheduling stochastic tasks, so to achieve the objective of power-limited delay minimization. In [20], authors assume that task owners use orthogonal channels for input data transmission, and a canonical matching game is formulated to study the formation of a mutual beneficial relationship between tasks and edge nodes. Finally, in [21], authors formulate the task scheduling and task image placement in a fog computing supported software-defined embedded system (FC-SDES) as a mixed-integer nonlinear programming problem, and propose a computation-efficient algorithm to find the optimal solution.

Our work is different from the mentioned approaches, as we concentrate on the task assignment in a MEC based architecture and propose a distributed and context-aware task offloading mechanism to reduce the overall energy consumption, while satisfying the tasks' delay constraints. In particular, all of the considerable operations are made based on end users' local information.

## III. Problem Formulation

In this paper, mobile devices are classified into two categories: mobile devices with excessive computational capabilities, and those with limited computational capabilities. Note that all mobile devices may generate tasks but only those with excessive computational capabilities and MEC servers can accept offloaded tasks. For the sake of readability, mobile devices with excessive computational capabilities and MEC servers are collectively termed as edge nodes (ENs).

Time is divided into multiple time slots. We assume that most tasks can be completed within one slot, while large-size tasks are divided into several sub-tasks so that they can also be completed in one slot. The terminology "task" will be used to refer to both the task which is going to be computed as a whole in one slot and the divided sub-task in the rest of this paper. Furthermore, the number of tasks is $M$ and the set of tasks is denoted by $\boldsymbol{\tau} = \{\tau_i\}_{i=1}^{M}$. On the other hand, the number of ENs is $N$ and the set of ENs is represented by $\boldsymbol{\epsilon} = \{\epsilon_j\}_{j=1}^{N}$. ENs with high computational capability can divide its resource into several virtual resource units (VRUs) equally such that tasks can be computed in a parallel manner. The number of VRUs at EN $j$ is referred to as the quota of EN $j$, which is denoted by $q_j$. The computational capabilities of VRUs at different ENs are assumed to be heterogeneous. We use the CPU frequency $f_j$ (in Hz) to describe the computational capability of each VRU at EN $j$.

### A. One-to-Many Matching

A one-to-many matching between the task set and the EN set is considered. Namely, one task can be assigned to at most one EN, while one EN can accept multiple tasks. Formally, the one-to-many matching function $\Phi$ is defined as:

$$\begin{aligned} & \Phi : \{\boldsymbol{\tau}\} \times \{\boldsymbol{\epsilon}\} \leftrightarrow \{\boldsymbol{\tau}\} \times \{\boldsymbol{\epsilon}\} \\ \text{s.t.} \quad & 1)\ \Phi(\tau_i) \in \boldsymbol{\epsilon} \text{ and } |\Phi(\tau_i)| \in \{0,1\} \\ & 2)\ \Phi(\epsilon_j) \subset \boldsymbol{\tau} \text{ and } |\Phi(\epsilon_j)| \leq q_j \\ & 3)\ \Phi(\tau_i) = \epsilon_j \Leftrightarrow \Phi(\epsilon_j) = \tau_i \end{aligned} \tag{1}$$

Condition 1) implies that each task could be assigned to at most one EN for execution. Condition 2) implies that each EN could accept at most its quota of tasks. Condition 3) implies that if $\tau_i$ is matched to $\epsilon_j$ then $\epsilon_j$ is matched to $\tau_i$ as well.

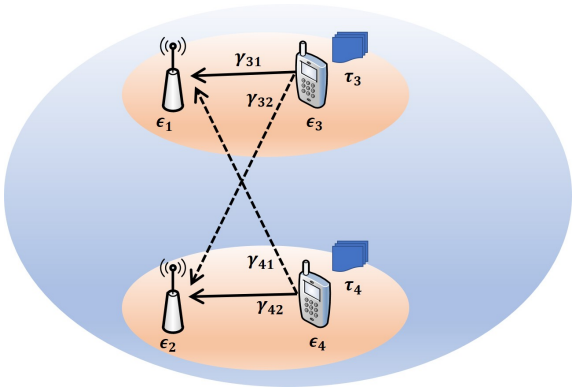


Figure 1. System model for the uplink transmission in MEC architectures

The matching index $x_{ij}$ is therefore defined as:

$$x_{ij} = \begin{cases} 1 & \text{if } \Phi(\tau_i) = \epsilon_j \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where $x_{ij} = 1$ implies that task $i$ is matched to EN $j$, while $x_{ij} = 0$ means that task $i$ is not matched to EN $j$.

### B. Delay Constraint

First, we consider the delay constraint which is one of major concerns that need to be addressed in the task assignment for MEC. In practise, different task owners have different time sensitivities. The metric of delay tolerance is a measure of task owner's time sensitivity, and it is defined as the time a computation request is made until the task should be completed. Specifically, the delay tolerance of each task $i$ is denoted by $L_i^{\text{tol}}$.

Since offloading tasks incurs extra transmission energy consumption and transmission latency, each task owner has to carefully decide wether to compute its task locally or offload it to neighbouring ENs. In case of task offloading, the overall latency generally consists of three components: (i) transmission delay; (ii) queuing delay; and (iii) computation delay. First, the transmission delay refers to the time of transferring the input data or application context to a neighbouring EN through a wireless connection, which highly depends on the wireless channel state and the input data size. In case of local computation, the transmission delay is 0. Second, the queuing delay is the time a task waits in a queue until it can be executed. In this article, we assume that each task exclusively utilize the whole resources of an EN or VRU for task execution. Therefore, queuing delay could be omitted. Finally, the computation delay is defined as the time needed to execute the task, which highly depends on both the EN's computation capability, and the size of task.

We focus on the uplink transmission analysis. As shown in Fig.1, the channel gain from the owner of task $i$ to EN $j$ in the $t$-th time slot is $\gamma_{ij}(t)$, $p_i^{\text{tra}}$ is the maximum transmission power of the owner of task $i$, $B$ is the system bandwidth, $N_0$ is the noise power spectral density at the receiver. The received signal to interference plus noise ratio (SINR) at EN $j$ can therefore be expressed as:

$$\Gamma_{ij}(t) = \frac{p_i^{\text{tra}}\gamma_{ij}}{N_0 + \sum_{k\neq i} p_k^{\text{tra}}\gamma_{kj}} \tag{3}$$

Then the achievable data rate of from the owner of task $i$ to EN $j$ is given by:

$$R_{ij}(t) = B\log\left(1 + \frac{p_i^{\text{tra}}\gamma_{ij}}{N_0 + \sum_{k\neq i} p_k^{\text{tra}}\gamma_{kj}}\right) \tag{4}$$

Formally, when task $i$ is matched to EN $j$, the total energy consumption is denoted by $L_{ij}$ as:

$$L_{ij} = L_{ij}^{tra} + L_{ij}^{com} \tag{5}$$

where $L_{ij}^{com}$, and $L_{ij}^{tra}$ represent computational delay and transmission delay, respectively.

On the one hand, the transmission delay is given by:

$$L_{ij}^{tra} = \begin{cases} \frac{S_i}{B\log\left(1+\frac{p_i^{\text{tra}}\gamma_{ij}}{N_0+\sum_{k\neq i} p_k^{\text{tra}}\gamma_{kj}}\right)} & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $S_i$ denotes the input data size of task $i$.

On the other hand, the computational delay is given by:

$$L_{ij}^{com} = \frac{C_i}{f_j} \tag{7}$$

where $C_i$ represents the number of required CPU cycles for executing task $i$ successfully.

Intuitively, for computation-intensive tasks, computing delay is responsible for a large portion of the overall delay. Hence, computation-intensive tasks tends to find an EN with high computing capability to meet their latency constraints, while other kinds of task may prefer to be computed locally to save the extra transmission delay incurred by task offloading.

### C. Energy Consumption

Energy consumption is another major concern that need to be addressed in task assignment. The total energy consumption for executing task $i$ on EN $j$ is denoted by $P_{ij}$ as:

$$P_{ij} = P_{ij}^{\text{tra}} + P_{ij}^{\text{com}} \tag{8}$$

where $P_{ij}^{\text{tra}}$ and $P_{ij}^{\text{com}}$ represent the transmission energy consumption of task owner $i$ and the computation energy consumption of EN $j$, respectively.

On the one hand, the transmission energy consumption of task owner $i$ is given as:

$$P_{ij}^{\text{tra}} = \begin{cases} \frac{S_i p_i^{\text{tra}}}{B\log\left(1+\frac{p_i^{\text{tra}}\gamma_{ij}}{N_0+\sum_{k\neq i} p_k^{\text{tra}}\gamma_{kj}}\right)} & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

The transmission energy consumption is 0 in case of the local computation mode.

On the other hand, the computation energy consumption of EN $j$ is given as:

$$P_{ij}^{\text{com}} = \frac{C_i p_j^{\text{com}}}{f_j} \tag{10}$$

where $f_j$ (in Hz) is the CPU frequency of node $j$, and $p_j^{\text{com}}$ (in Joule per second) represents its energy consumption level.

### D. Utility Functions and Optimization Problem

In matching theory, utility is used to evaluate the gain of a agent achieved by a certain matching. We first define the utility of task $i$, if it is computed on EN $j$ as:

$$u_{ij} = \begin{cases} r_i - P_{ij}^{\text{tra}} - \lambda C_i & i \neq j \\ r_i - P_{ij}^{\text{com}} & \text{otherwise} \end{cases} \tag{11}$$

where $r_i$ is the owner's reservation price (i.e., satisfaction) if task $i$ could be finished within the designated delay tolerance; $\lambda$ is the unit price per CPU cycle that the owner of task $i$ pays for the computation service offered by other EN. Note that the overall payment $\lambda C_i$ is proportional to the size of the task.

The utility of EN $j$ achieved by completing task $i$ is given as:

$$v_{ij} = \begin{cases} \lambda C_i - P_{ij}^{\text{com}} & i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

Resource-limited devices seek to offload their computations to nearby MEC servers while taking into account devices' and servers' computation capabilities, co-channel interference, and reliability constraints. The fundamental problem under consideration is cast as a network-wide utility maximization problem for task assignment, subject to delay requirements. By summing up the utility of all tasks and ENs over discreet $x_{ij}$, we have:

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\left(u_{ij}+v_{ij}\right)x_{ij} = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(r_i - P_{ij}^{\text{tra}} - P_{ij}^{\text{com}}\right)x_{ij} \tag{13}$$

The utility maximization problem is transformed to an energy consumption optimization problem, while satisfying task owners' heterogeneous delay requirements. Our delay-constrained energy consumption optimization problem is therefore defined as:

$$\begin{aligned} &\max_{\{x_{ij}\}} \sum_{i=1}^{M}\sum_{j=1}^{N}\left(r_i - P_{ij}^{\text{tra}} - P_{ij}^{\text{com}}\right)x_{ij} \\ \text{s.t.}\quad &1)\ \sum_{j=1}^{N} x_{ij} \leq 1, \forall i \in \{1,...,M\} \\ &2)\ \sum_{i=1}^{N} x_{ij} \leq q_j, \forall j \in \{1,...,N\} \\ &3)\ x_{ij}L_{ij} \leq L_i^{\text{tol}}, \forall i \in \{1,...,M\} \\ &4)\ u_{ij}x_{ij} \geq 0, \forall i \in \{1,...,M\} \\ &5)\ v_{ij}x_{ij} \geq 0, \forall i \in \{1,...,M\} \\ &6)\ \Gamma_{ij} \geq x_{ij}\Gamma_0, \forall i \in \{1,...,N\} \\ &7)\ x_{ij} \in \{0,1\}, \forall i \in \{1,...,M\} \text{ and } \forall j \in \{1,...,N\} \end{aligned} \tag{14}$$

Condition 1) guarantees that each task will be assigned to at most one EN; Condition 2) guarantees that each EN can accept at most its quota of tasks; Condition 3) guarantees that each task will be completed on time; Condition 4) an 5) imply that the utility of each task and EN should not be negative. Condition 6) implies that the SINR should be higher than a threshold value (i.e., $\Gamma_0$) to guarantee a successful transmission (reliable transmission constraint); Condition 7) implies that a task could be either matched or unmatched.

The optimization problem shown in Eq.(14) is in the standard form of binary linear programming problem (BLP) with a bundle of constraints, which is proven to be NP-complete [22]. Furthermore, given the utility functions shown in Eq.(11) and Eq.(12), when multiple task owners in the proximity use the same sub-channel for input data transmission, the inter-user interference increases and the achievable data rate decreases. As a key consequence, the order of preferences for a task may change. In other words, the utility of an agent (task or EN) depends on the other agents' choice, which are called externalities in matching theory. If the externalities are not well managed, an agent may have to keep changing its preference order responsive to the changes in other task-EN associations, and a stable result could never be expected.

## IV. Swap-Matching Based Algorithm

Since the optimization problem shown in Eq.(14) is NP-complete, and the utility of each agent is strongly affected by the externalities, we hence propose a heuristic swap matching [23] based algorithm to solve the optimization problem. The purpose of the proposed algorithm is to obtain a practical and stable solution to the optimization problem in a distributive fashion. More specifically, all decision makings could be made locally by each task and EN.

### A. Swap Matching

**Definition 1.** *Given a matching $\boldsymbol{\eta}$ and two task-EN pairs $(\tau_i, \epsilon_m)$, $(\tau_j, \epsilon_n) \in \boldsymbol{\eta}$, a swap matching is defined as $\boldsymbol{\eta}_{ij}^{mn} = \{\boldsymbol{\eta} \backslash (\tau_i, \epsilon_m), (\tau_j, \epsilon_n)\} \cup \{(\tau_j, \epsilon_m), (\tau_i, \epsilon_n)\}$, such that*

$$\begin{aligned} u_{in} \geq u_{im} \text{ and } u_{jm} \geq u_{jn} \\ v_{in} \geq v_{jn} \text{ and } v_{jm} \geq v_{im} \end{aligned} \tag{15}$$

*Formally, $\boldsymbol{\eta}_{ij}^{mn} \succ \boldsymbol{\eta}$ implies that $\boldsymbol{\eta}_{ij}^{mn}$ is a swap matching of $\boldsymbol{\eta}$.*

**Definition 2.** *A matching function $\boldsymbol{\eta}$ is called two-side exchange stable if there is no agent that has incentive to swap from its current association.*

Given the definitions, a matching $\boldsymbol{\eta}$ with a pair $(\tau_i, \epsilon_m) \in \boldsymbol{\eta}$ is considered to be two-side exchange stable if there does not exist any pair $(\tau_j, \epsilon_n) \in \boldsymbol{\eta}$ , for which task $i$ prefers EN $n$ over EN $m$, or any EN $m$ which prefers task $j$ over $i$. Such two-side exchange stability is achieved by guaranteeing that swaps occur if and only if the swaps are beneficial for all of the agents involved. In particular, a swap matching $\boldsymbol{\eta}_{ij}^{m}$ can only occur, if this causes no decline of the utility for any of the agents in $\{\tau_i, \tau_j, \epsilon_m, \epsilon_n\}$, without decreasing the utilities of the other agents (both tasks and ENs) that are not involved in the swap. When the stability is achieved, swap matching will no longer occur in the result.

### B. Algorithm

A detailed specification of the proposed concept is shown in **Algorithm** 1, which consists of an initialization stage, and an iterative swap matching stage.

In the initialization step, each agent firstly exchanges information (e.g., input data size, required CPU cycles, and computing capacity) with the agents in its proximity. In particular, the set of indexes of ENs to which the input data of task $i$ could be successfully transmitted is denoted by:

$$\boldsymbol{\phi}_i^{\boldsymbol{\epsilon}} = \{j' | \frac{p_i^{\text{tra}} \gamma_{ij'}(t)}{N_0} \geq x_{ij'} \Gamma\} \tag{16}$$

Secondly, each task is initially matched to a randomly selected EN as long as all conditions in Eq.(14) can be satisfied. Finally, the owner of each task $i$ ranks EN $j \in \boldsymbol{\phi}_i^{\boldsymbol{\epsilon}}$ in order of preference. The preference is set up based on the utility function that qualifies the QoS achieved by a certain task-EN matching shown in Eq.(11).

In each iteration, the owner of each task $i$ firstly proposes to its most preferred EN in $\boldsymbol{\phi}_i^{\boldsymbol{\epsilon}}$ except for its current partner: $\epsilon_m$. If $\boldsymbol{\eta}_{ij}^{mn} \succ \boldsymbol{\eta}$ and all conditions in Eq.(14) are satisfied, then the swap operation will be approved. Otherwise the swap operation will be rejected. The algorithm proceeds iteratively and terminates when there is no task that has incentive to swap from its current association.

## V. Numerical Evaluation

Simulations are conducted to evaluate the performance of the proposed scheme. We simulate a 50 m×50 m square area with 10 ENs randomly deployed on the area. The input data size and the number of required CPU cycles of each task are randomly selected within the range of [400, 800] kB, and [1, 5]$\times 10^9$, respectively. The delay tolerance of each task is set to 5 sec. The transmission power ($p_i^{\text{tra}}$) and computing energy consumption level ($p_i^{\text{com}}$), are set to 36 dBm and 40 W, respectively. Moreover, the CPU frequency of each EN is randomly selected within the range of [2, 4] GHz. Other simulation settings can be found in TABLE I. We use a random matching scheme, and a sequential optimal matching (SOM) scheme for comparison.

- Random Matching: tasks and ENs are randomly paired together as long as all conditions in Eq.(14) are satisfied.
- SOM: tasks are sequentially matched to their most preferred ENs as long as all conditions in Eq.(14) are satisfied.

Figure 2 and 3 depict the overall energy consumption and the overall delay with varying number of tasks. Compared to the random matching and the SOM schemes, the overall energy consumption of the proposed scheme is reduced by 16.04% and 15.21%; the overall delay of the proposed scheme is reduced by 18.77% and 14.34%, respectively. This is not surprising, because in our proposed scheme, each agent (task or EN) ranks the agents in the opposite set based on a utility function that captures the energy consumption, and our matching algorithm in nature guarantees the overall energy efficiency. Although the energy consumption and the delay constraint are also taken into account when pairing up ENs and tasks in the SOM scheme, earlier paired task-ENs are likely to starve the task-ENs with higher energy efficiency. The gap between the proposed scheme and the SOM scheme exactly shows the efficiency achieve by performing swap operations.

**Algorithm 1** The proposed algorithm

**Step 1: Initialization stage**
The owner of each task $i$ broadcasts its input data size $S_i$, the number of required CPU cycles $C_i$ to its neighbouring ENs. ENs that received the broadcast message replies with their CPU frequency $f_j$.
Each task is initially matched to a randomly selected EN, as long as all conditions in Eq.(14) are satisfied.
Each task then calculates its utility, and constructs its preference list according to Eq.(11).
**Step 2: Swap matching stage**
**while** $\boldsymbol{\phi}_i^{\boldsymbol{\epsilon}} = \{j' | \frac{p_i^{\text{tra}} \gamma_{ij'}(t)}{N_0} \geq x_{ij'} \Gamma_0\}$ is not empty **do**
    Given the current matching $(\tau_i, \epsilon_m)$, each $\tau_i$ makes a proposal to a neighbouring node (except for its current partner $\epsilon_m$) that ranks the first in its preference list: $\epsilon_n$.
    **for** each task $j$ that is currently matched to $\epsilon_n$ **do**
        **if** $\boldsymbol{\eta}_{ij}^{mn} \succ \boldsymbol{\eta}$ and all conditions in Eq.(14) are satisfied **then**
            The swap operation is approved: $\boldsymbol{\eta}_{ij}^{mn} \rightarrow \boldsymbol{\eta}$.
        **else**
            The swap operation is denied.
        **end if**
    **end for**
**end while**
**Step 3: End**
**while** there exist a task that can find its swap matching **do**
    Go to Step 2.
**end while**

Table I
Simulation settings

| Simulation parameter | Value |
|---|---|
| Number of ENs ($N$) | 10 |
| Number of Tasks ($M$) | (0,50] |
| Input data size ($S_i$) | [400, 800] kB |
| Number of required CPU cycles ($C_i$) | [1, 5]$\times 10^9$ |
| Delay tolerance ($L_i^{\text{tol}}$) | 5 sec |
| Maximum transmission power ($p_i^{\text{tra}}$) | 36 dBm |
| Energy consumption level ($p_j^{\text{com}}$) | 40 W |
| Quota ($q_j$) | 1, 2, or 3 |
| System bandwidth ($B$) | 20 MHz |
| Channel power gain ($\gamma_{ij}$) | $-40d^{-4}$ (dB) |
| CPU frequency ($f_j$) | [2, 4] GHz |
| Noise power spectral density ($N_0$) | -110 dB |
| Unit price per CPU cycle ($\lambda$) | 1.5 |
| SINR threshold value ($\Gamma_0$) | - 3 dB |

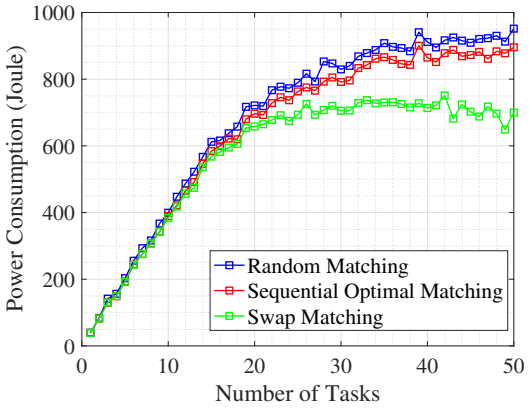


Figure 2. Power consumption of the proposed scheme versus the random matching and the SOM schemes.

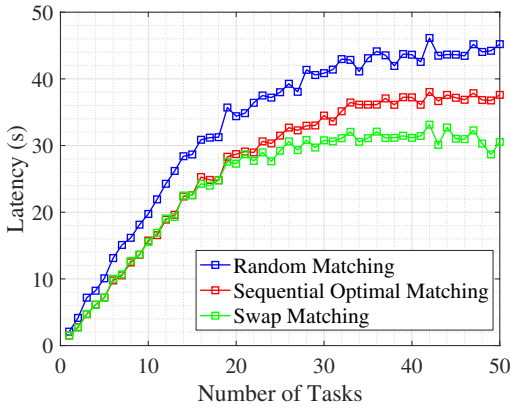


Figure 3. Overall latency of the proposed scheme versus the random matching and the SOM schemes.

## VI. Conclusions

In this paper, we propose a context-aware task assignment scheme for MEC systems aiming at optimizing overall energy consumption, while satisfying task owners' heterogeneous delay requirements. The task assignment problem is formulated as a matching game with externalities, where tasks and ENs set up preferences over one another so as to be matched to their preferred partners. Then we propose a heuristic swap matching based algorithm that solves the energy consumption minimization problem in focus. Simulation results confirm that our proposed scheme outperforms the random matching and the SOM schemes in terms of reducing the overall energy consumption and latency.

## References


[1] P. M. Mell and T. Grance, "Sp 800-145. the nist definition of cloud computing," 2011.

[2] Cisco, "Cisco global cloud index: Forecast and methodology, 2016-2021 white paper," Feb. 2018.

[3] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *2016 10th International Conference on Intelligent Systems and Control (ISCO)*, Jan 2016, pp. 1–8.

[4] ETSI, "Mbile-edge computing-introductory technical white paper," Sept. 2014.

[5] T. H. Luan, L. Gao, Z. Li, Y. Xiang, and L. Sun, "Fog computing: Focusing on mobile users at the edge," *CoRR*, vol. abs/1502.01815, 2015. [Online]. Available: http://arxiv.org/abs/1502.01815

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct 2016.

[7] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5g networks: New paradigms, scenarios, and challenges," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, April 2017.

[8] L. Gu, D. Zeng, S. Guo, A. Barnawi, and Y. Xiang, "Cost efficient resource management in fog computing supported medical cyber-physical system," *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 1, pp. 108–119, Jan 2017.

[9] N. B. Truong, G. M. Lee, and Y. Ghamri-Doudane, "Software defined networking-based vehicular adhoc network with fog computing," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, May 2015, pp. 1202–1207.

[10] Z. Zhou, L. Tan, B. Gu, Y. Zhang, and J. Wu, "Bandwidth slicing in software-defined 5g: A stackelberg game approach," *IEEE Vehicular Technology Magazine*, vol. 13, no. 2, pp. 102–109, June 2018.

[11] V. Frascolla, J. Englisch, and L. Chiaraviglio, "Millimeter-waves, mec, and network softwarization as enablers of new 5g business opportunities," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, Barcelona, Spain, Apr. 2018, pp. 16–19.

[12] E. Ahmed, A. Naveed, A. Gani, S. H. A. Hamid, M. Imran, and M. Guizani, "Process state synchronization for mobility support in mobile cloud computing," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6.

[13] E. Baccarelli, P. G. V. Naranjo, M. Scarpiniti, M. Shojafar, and J. H. Abawajy, "Fog of everything: Energy-efficient networked computing architectures, research challenges, and a case study," *IEEE Access*, vol. 5, pp. 9882–9910, 2017.

[14] Z. Zhou, K. Ota, M. Dong, and C. Xu, "Energy-efficient matching for resource allocation in d2d enabled cellular networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 5256–5268, June 2017.

[15] Z. Zhou, J. Gong, Y. He, and Y. Zhang, "Software defined machine-to-machine communication for smart energy management," *IEEE Communications Magazine*, vol. 55, no. 10, pp. 52–60, October 2017.

[16] Z. Zhou, J. Feng, B. Gu, B. Ai, S. Mumtaz, J. Rodriguez, and M. Guizani, "When mobile crowd sensing meets uav: Energy-efficient task assignment and route planning," *IEEE Transactions on Communications*, pp. 1–1, 2018.

[17] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.

[18] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, Dec 2016.

[19] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE International Symposium on Information Theory (ISIT)*, July 2016, pp. 1451–1455.

[20] B. Gu, Y. Chen, H. Liao, Z. Zhou, and D. Zhang, "A distributed and context-aware task assignment mechanism for collaborative mobile edge computing," *Sensors*, vol. 18, no. 8, 2018.

[21] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, "Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system," *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3702–3712, Dec 2016.

[22] R. M. Karp, *Complexity of Computer Computations*. Springer, Boston, MA, 1972, ch. Reducibility among Combinatorial Problems, pp. 85–103.

[23] F. Pantisano, M. Bennis, W. Saad, S. Valentin, and M. Debbah, "Matching with externalities for context-aware user-cell association in small cell networks," in *2013 IEEE Global Communications Conference (GLOBECOM)*, Dec 2013, pp. 4483–4488.