A Socially-Aware Hybrid Computation Offloading Framework for Multi-Access Edge Computing

Shuai Yu, Boutheina Dab, Zeinab Movahedi, Rami Langar, and Li Wang, *Senior Member, IEEE*

Abstract—Computation offloading manages resource-intensive and mobile collaborative applications (MCA) on mobile devices where much processing is replicated with multiple users in the same environment. In this article, we propose a novel hybrid multicast-based task execution framework for multi-access edge computing (MEC), where a crowd of mobile devices at the network edge leverage network-assisted device-to-device (D2D) collaboration for wireless distributed computing (MDC) and outcome sharing. The framework is socially aware in order to build effective D2D links. A key objective of this framework is to achieve an energy-efficient task assignment policy for mobile users. Specifically, we first introduce the socially aware hybrid computation offloading (*SAHCO*) system model, which combines of MEC offloading and D2D offloading in detail. Then, we formulate the energy-efficient task assignment problem by taking into account the necessary constraints. We next propose a Monte Carlo Tree Search based algorithm, named, *TA-MCTS* for the task assignment problem. Simulation results show that compared to four alternative benchmark solutions in literature, our proposal can reduce energy consumption up to 45.37 percent.

Index Terms—Mobile collaborative application, wireless distributed computing, computation offloading, multicast communication, socially aware, monte carlo tree search

1 Introduction

Nowadays, the advances in hardware technology has leaded to more powerful mobile devices in terms of memory, processing speed and network connectivity. This development has pushed to the pervasive computing era, in which different mobile devices, ranging from smartphones, tablets and laptops to low-power sensors have widely penetrated to our everyday life. Accompanied by the emergence of near-to-eye display technologies, a variety of mobile collaborative applications (MCA) are developed to meet the user's requirements, such as augmented reality (AR) [2], collaborative gaming [3] and mobile crowd sensing applications [4]. These applications make use of complex algorithms for camera tracking, image processing and pattern recognition which are resource-intensive and hence, beyond the capabilities of current mobile devices. To cope with such challenges, a recent promising alternative consists in offloading the whole or some parts of these applications out of the mobile device, specially on powerful cloud servers.

- S. Yu is with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510275, China. E-mail: yushuai@mail.sysu.edu.cn.
- B. Dab is with LIP6, Sorbonne Université, Paris 75005, France. E-mail: Boutheina.Dab@lip6.fr.
- Z. Movahedi is with the Computer Engineering School, Iran University of Science and Technology (IUST), Tehran 16846-13114, Iran. E-mail: zmovahedi@iust.ac.ir.
- R. Langar is with LIGM CNRS-UMR 8049, University Paris Est Marne-la-Vallée (UPEM), Champs-sur-Marne 77420, France. E-mail: Rami.Langar@u-pem.fr.
- L. Wang is with the School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China, and also with the Key Laboratory of the Universal Wireless Communications, Ministry of Education, Beijing 100816, P.R. China. E-mail: liwang@bupt.edu.cn.

Manuscript received 12 June 2018; revised 29 Jan. 2019; accepted 24 Mar. 2019. Date of publication 29 Mar. 2019; date of current version 5 May 2020. (Corresponding author: Shuai Yu). Digital Object Identifier no. 10.1109/TMC.2019.2908154

Another alternative is to take advantage of available nearby mobile resources for executing resource-intensive applications of mobile devices, referred as device-to-device (D2D) computation offloading, Transient Clouds [5], or wireless distributed computing (WDC) [6], [7]. The main asset of computation offloading within a cluster of mobile devices consists in reducing the per-node and network power, energy, and processing resource requirements. Power analysis has shown that this approach is beneficial over local processing under certain conditions, particularly when the computational cost exceeds the communication overhead [7]. The latter conditions are usually achieved when resource intensive tasks are distributed in short-range networks [6]. Considering the short-range network services, multicast communication plays an important role in data distribution and energy reduction. As a combination of D2D and multicast, computation offloading through D2D multicast communication [8] enables multiple proximate users to share the content items of their common interests and computation results. This latter combination can greatly reduce power consumption (both communication and computation) and improve spectral efficiency in a local network.

Privacy issue is an important threat for the hybrid computation offloading system. On the one hand, mobile users risk exposing their sensitive data (e.g., personal images, personal clinical data and business financial records in real-life situations) by offloading it to untrusted mobile users through D2D computation offloading. On the other hand, MEC computation offloading will also face privacy leakage issues, when mobile users migrate computation to the edge servers for edge data analytic. Thus, it is a huge challenge to protect mobile users' sensitive information in the hybrid computation offloading system. A potential solution to address the challenges is to consider the social domain factors besides physical domain in the hybrid computation offloading. It is essential to find a model to describe the physical and social characteristics of the network, and to quantify the social relationships among mobile users.

To this end, we propose a fine-grained task assignment mechanism for MCA execution in MEC network with social trust consideration. The objective is to minimize the overall energy consumption at mobile terminal side. Based on the concepts of the call graph [9] and social trust [10], we propose in this article the framework of socially aware D2D computation offloading (*SAHCO*) and then compute the energy overhead for each application cluster. In order to enhance the performance, we solve the computation offloading problem for all the components of an application at the same time. In our previous work [1], while the sequential processing for components minimizes the computation energy, it may fail to ensure an efficient optimized offloading decision. To address the problem for the set of MCA components, we propose a new optimal task assignment approach based on Monte-Carlo search tree (MCTS) [11], named *TA-MCTS*. Our proposed solution, *TA-MCTS*, achieves an optimized computation offloading policy. We compare our approach with the related benchmark policies and with our previous sequential strategy [1] in literature. We discuss the associated gains in terms of mobile user density, social trust threshold and CPU structure, respectively.

The reminder of this paper is organized as follows. Section 2 introduces an overview of the related works. In Section 3, we present the system model of our proposed socially aware hybrid computation offloading (SAHCO) framework. Section 5 formulates the optimaztion problems of proposed SAHCO, followed by a description of our proposed algorithm in Section 6. Simulation results are presented in Section 7. Finally, conclusions are drawn in Section 8.

2 Related Work

2.1 Cloudlet-Based Computation Offloading

A cloudlet is a mobility-enhanced small-scale cloud datacenter that is located at the network edge. The main purpose of the cloudlet is to support mobile users execute their resource-intensive and interactive mobile applications with lower latency and energy consumtion. José et al in [12] propose ULOOF, an offloading framework that is equipped with a decision engine to minimizes cloudlet execution overhead, while not requiring any modification in the device's operating system. Cuervo et al in [13] propose a framework called MAUI, which focuses on energy saving. It uses a profiler which measures energy consumption and a solver which decides whether to offload or not a method based on the measurement provided from the profiler. Recently, the European Telecommunications Standards Institute (ETSI) provide a concept of multi-access edge computing (MEC) in their 5G standard [14]. In the MEC architecture, distributed MEC servers are located at the network edge to provide cloud-computing capabilities and IT services with low latency, high bandwidth, and real-time processing.

2.2 Wireless D2D Computation Offloading

The idea of offloading computations to nearby mobile devices has been studied in many works [5], [15]. Pu et al in [15] propose D2D Fogging, a mobile task offloading framework based on network-assisted D2D collaboration, where mobile users can dynamically and beneficially share the computation and communication resources among each other via the control assistance by the network operators. To minimize the time-average energy consumptions for task executions of all the users, they develop an online task offloading algorithm, which leverages Lyapunov optimization methods and utilizes the current system information only. Authors in [5] proposed a concept of *Transient Clouds* that allows mobile devices within range of each other to form an ad-hoc network and collaboratively execute tasks. However, the above works are based on D2D unicast communication. Thus only one user can benefit through a D2D unicast offloading group, which is inefficient. In addition, they ignore the fact that offloading computation to the nearby mobile devices that are also resource-poor cloudlet may cause performance degradation.

2.3 Socially Aware D2D Communication

Socially aware D2D communication has been studied extensively [8], [10], [16], [17], [18]. In [10], the authors discussed social interactions in cooperative D2D networks. In order to leverage social relationships, they first introduced a physical-social model and social relationship metrics. Then they proposed a social learning based cooperative network evaluation method and social relationship based peer selection method. Authors in [8] proposed a socially aware resource allocation for 5G D2D multicast communications. The objective of the paper is to maximize the throughput of the overall socially aware network and guarantee fairly allocation of the channel between different D2D multicast clusters. However, the above works focus on the communication aspect, this motivates us to propose a novel D2D computation offloading framework for the MEC network with social relationship consideration.

3 System Model

We consider a hybrid computation offloading system formed by a set $\mathcal{M} = \{1, 2, \ldots, M\}$ of mobile users and a MEC server. The term of hybrid computation offloading systems signifies that both the central MEC server and distributed mobile devices contributes to the system smooth operation. The MEC server has generally a global view of the whole network with limited computation and storage capacity and conducts both the cellular and D2D connections for mobile users. We also consider that both of the cellular and D2D connections are based on the orthogonal frequency division multiple-access (OFDMA) [19] in which M users within the MEC server are separated in the frequency domain. Note that, using such a transmission scheme for the uplink offloading implies that the users do not interfere with one another. For the sake of simplicity, we consider the basic fixed channel assignment (FCA) scheme [20], [21] for the mobile users. The bandwidth that is allocated to each user is B_m. Then, we will introduce the system model for our socially aware hybrid computation offloading framework.

3.1 Cellular Link Model

In our hybrid computation offloading system, each user can establish a cellular link with the MEC server. Moreover, we assume that the locations of users remain unchanged and that wireless channels are stable during one decision making procedure. Note, however, they may change across decision making procedure due to users' mobility. The maximum uplink rate in (bps), achievable for a user m $(m \in \mathcal{M})$ during each offloading decision procedure, over an additive white Gaussian noise (AWGN) channel, can be expressed as follows:

$$r_m^{ul} = B_m \log_2\left(1 + \frac{p_m^u |h_m^{ul}|^2}{\Gamma(g_{ul}) d_m^\beta N_0}\right), \tag{1}$$

where B_m is the bandwidth that is allocated to mobile user m, p_m^u denotes the transmit power for the mobile user m, d_m is the distance between the mobile user m and the MEC server, and N_0 denotes the noise power. In this paper, we consider the Rayleigh-fading environment for which we define h_m^{ul} as the channel fading coefficient for uplink, and β as the path loss exponent. Note that $\Gamma(BER) = -\frac{2log(5BER)}{3}$ represents the SNR margin introduced to meet the desired target bit error rate (BER) with a QAM constellation. g_{ul} is the target BER for uplink. Although the transmit power p_m^u and the corresponding user transmission rate r_m^{ul} are time-varying. We do not consider these parameters as control variables in our system similar to assumptions made in [22].

3.2 D2D Link Model

In our hybrid computation offloading system, each user can establish a D2D link with another user in proximity. Similar, we assume that, as in [23], two mobile users mobile user 1 and mobile user 2 (as shown in Fig. 3) can communicate with each other if and only if their separating distance is lower than the threshold R^d. In our proposed system model, as described previously, the MEC server plays a pivotal role in assisting D2D communications. Indeed, the MEC server can execute the device discovery process for a user to detect the set of its nearby users, and establish the LTE-direct between users. Note that the feasible D2D links among the users can be varying across different offloading decision making procedures. We denote by $r_{i,j}^{d2d}$ the D2D transmission rate from mobile user i to j as follows:

$$r_{i,j}^{d2d} = B_m \log_2\left(1 + \frac{p_i^{d2d} |h_{i,j}^{d2d}|^2}{\Gamma(g_{d2d}) d_{i,j}^\beta N_0}\right), (i, j \in \mathcal{M}), \tag{2}$$

p_i^{d2d} is the D2D transmission power of user i. $h_{i,j}^{d2d}$ and g_{d2d} denote the channel fading coefficient and target BER for D2D link, respectively. $d_{i,j}$ represents the distance between mobile user i and j.

In order to efficiently deliver computation results, we consider the multicast transmission scheme to transmit the common computation results. Note that we distinguish two kinds of Multicast: i) Multi-rate and ii) Single-Rate. In this paper, we consider Single-Rate Multicast transmission with Least Channel Gain (LCG) user rate [21]. Our choice is motivated by the fact that this scheme adaptively sets the group transmission rate to suit the user with the worst (minimum) channel quality. We consider a D2D multicast cluster consisting of a transmitter i and his set of receivers $\mathcal{Q}_i^{d2d}$ ($i \in \mathcal{M}$, $\mathcal{Q}_i^{d2d} \subset \mathcal{M}$).

Let $H_{i,j}^{d2d}$ denote the channel coefficient between the transmitter i and his receiver j ($j \in \mathcal{Q}_i^{d2d}$):

$$H_{i,j}^{d2d} = \frac{|h_{i,j}^{d2d}|^2}{\Gamma(g_{d2d}) d_{i,j}^\beta N_0}. \tag{3}$$

The multicast group transmission rate r_i^{d2d} is then given as:

$$r_i^{d2d} = B_m \log_2\left(1 + p_i^{d2d} \cdot h_{gr,i}^{d2d}\right), \tag{4}$$

where $h_{gr,i}^{d2d} = \min_{j \in \mathcal{Q}_i^{d2d}} H_{i,j}^{d2d}$ is the channel coefficient of minimum receiver j in the multicast cluster.

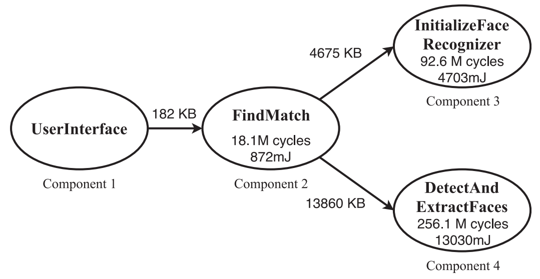


Fig. 1. Example of task graph for a face recognition application [13].

3.3 Application and Execution Model

3.3.1 Application Model

We assume that a mobile collaborative application can be split into multiple components [2] based on the granularity of either method [13] or thread (i.e., a fine-grained partitioning). We then exploit the concept of call graph [9] to model each mobile application. Typically, the call graph consists in modeling the relationship between components as a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the set of components, and $\mathcal{E}$ represents the data dependencies between components. Each edge $\mathcal{E}_{u,v} \in \mathcal{E}$ indicates the data communication (i.e., computation result) required by component v to be fed from component u. Let ϕ_v ($v \in \mathcal{V}$) denote the weight of component v, which specifies the total workload, in terms of CPU cycles, required by v. For a given input data size $\mathcal{E}_{u,v}$, ϕ_v can be derived from [24] as $\phi_v = \omega \cdot \mathcal{E}_{u,v}$, where ω expressed in cycles/byte (cpb) indicates the number of clock cycles a microprocessor will perform per byte of data processed in an algorithm. Note that this parameter depends on the nature of the component, e.g., the time complexity. The estimation of this value has been studied in [25] which is thus beyond the scope of our work.

Fig. 1 illustrates a call graph of a face recognition application [13]. Note that the dependency among different components cannot be ignored as it significantly affects the procedure of execution and computation offloading. Each vertex represents a component and its computational cost (e.g., $\phi_2 = 18.1$ M cycles for the "FindMatch"component) and energy cost. Each edge represents the size of the data dependencies between components (e.g., $\mathcal{E}_{1,2} = 182$ KB between components "UserInterface" and "FindMatch"). Note that due to either software or hardware constraints, some components can be offloaded to the server for remote execution, while other ones can only be evaluated locally. For example, component "UserInterface" in Fig. 1 must be executed on mobile device.

3.3.2 Execution Model

If the component v is executed on mobile device m locally, the completion time is $T_{m,v}^{local} = \phi_v / f_m^u$, and the corresponding energy consumption of the mobile device is $E_{m,v}^{local} = p_m^c \cdot \phi_v$, where f_m^u denotes the CPU rate of mobile user m (in Million Instructions Per Second, MIPS), p_m^c represents the energy consumption per cycle for local computing, ϕ_v denotes the number of instructions to be executed for component v.

In this paper, we divide the components into two categories, namely: i) offloadable and ii) non-offloadable, components. For the mobile user m, non-offloadable components consists of the components that must be executed locally (e.g., "UserInterface" in Fig. 1). The offlodable components denotes the rest components.

3.4 Social Relationship Model

In order to leverage the properties of social networks in the scenarios of our hybrid computation offloading, it is essential to find a model to describe the physical and social characteristics of the networks, and to quantify the social relationships among mobile devices. In this work, we use the social trust to measure the social relationship between mobile devices.

Social trust can be built up among humans such as kinship, friendship, colleague relationship, and altruistic behaviors as observed in many human activities. For example, when a mobile user is at home or work, typically family members, neighbors, colleagues, or friends are nearby within the same MEC coverage. Thus, a mobile user can then exploit the social trust from these neighboring users to offload his computational tasks, and the computation results can be shared among friends.

In this work, we measure the social relationship between mobile devices using three main metrics, namely social similarity, contact history and contribution history. These metrics are time varying and are defined as follows:

Social Similarity. Most smart devices are equipped with contact books or online social network applications from which the social relationships between two mobile users can be extracted. A simple way is to check whether a mobile user is in the contact book of another or is followed by him in online social networks. Let $x^s_{i,j}$ denote the social closeness index to measure the social similarity as follows:

$$x^s_{i,j} = \begin{cases} 1, & \text{if } i \text{ has social similarity with } j; \\ 0, & \text{if } i \text{ has no social similarity with } j. \end{cases} \tag{5}$$

$x^s_{i,j} = 1$ $(i, j \in \mathcal{M})$ signifies that either mobile user j is in the contact book or in the online social network followed by mobile user i. Otherwise, $x^s_{i,j}$ is equal to 0 denoting that mobile user j has no social similarity from user i's perspective. Note that we can not promise that $x^s_{i,j} = x^s_{j,i}$, while mobile user i has j in his contact book, mobile user j may not has i in his contact book.

Contact Closeness. The contact closeness measures the frequency of contact between two mobile users via both cellular and D2D links. Let $x^c_{i,j}$ denote the contact closeness index which is calculated as follows:

$$x^c_{i,j} = \begin{cases} 1, & I_{i,j} = 0; \\ \frac{1}{I_{i,j}} \cdot \left(\frac{t^c_{i,j}}{t^c_{i,j} + t^c_{j,i}} \right), & I_{i,j} \neq 0, t^c_{i,j} + t^c_{j,i} \neq 0; \\ 0, & t^c_{i,j} + t^c_{j,i} = 0, \end{cases} \tag{6}$$

where $I_{i,j}$ denotes the average time interval between two contiguous contacts between mobile users i and j during the past time unit (e.g., one week). Accordingly, $\frac{1}{I_{i,j}}$ denotes the average contact frequency and $t^c_{i,j}$ denotes the time of contact during which mobile user i contacts mobile user j within the past time unit.

Contribution History. This parameter rates the contribution of mobile devices in executing the applications of other devices. A mobile device which is more willing to help others should gain higher trust from others and get more chance to use reciprocally others' resources in the future. This latter encourages mobile users to cooperate and avoid selfish behaviour. Let $x^o_{i,j}$ denote the contribution closeness index to quantify contribution history as follows:

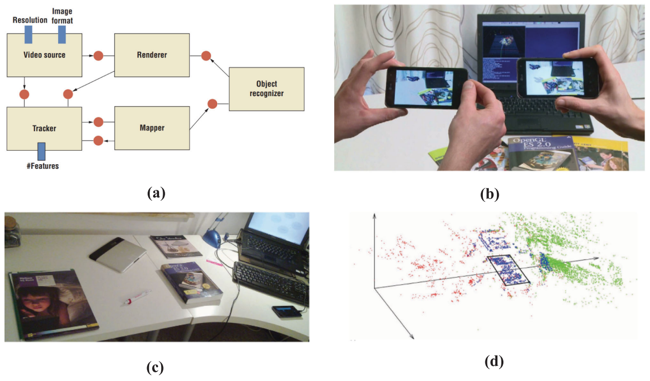


Fig. 2. Case study for an immersive application [2].

$$x^o_{i,j} = \begin{cases} \frac{\phi^c_{i,j}}{\phi^c_{i,j} + \phi^c_{j,i}}, & \phi^c_{i,j} + \phi^c_{j,i} \neq 0; \\ 0, & \phi^c_{i,j} + \phi^c_{j,i} = 0, \end{cases} \tag{7}$$

where $\phi^c_{i,j}$ denotes the amount of processed workload (CPU cycles) provided by mobile user j to help mobile user i in the contribution history. Based on the above metrics, we derive $x_{i,j}$ which represents the social relationship index between mobile users i and j in the view of mobile user i:

$$x_{i,j} = \zeta_1 x^s_{i,j} + \zeta_2 x^c_{i,j} + \zeta_3 x^o_{i,j}, \quad (i, j \in \mathcal{M}), \tag{8}$$

where ζ_1, ζ_2, and ζ_3 denote weight factors $(\zeta_1 + \zeta_2 + \zeta_3 = 1)$. Note that the larger the value of $x_{i,j}$, the closer the relationship, and vice versa. If the weight is equal to zero, then the two corresponding mobile users do not have any social relationship. The social relationship index between mobile devices is then fed to a social relationship weight matrix $X = [x_{i,j}]_{M \times M}$ $(i, j \in \mathcal{M})$ in which M is the number of mobile users. In this work, the matrix is centralized and dynamically maintained by the MEC server. Note that matrix X is not necessarily symmetric. Indeed in practice, while mobile user i believes it has a close relationship with mobile user j, mobile user j may not believe so.

The MEC server uses the social relationship weight matrix to check whether a mobile user i should trust a mobile user j. Let x^{thrd} denote a threshold of social relationship index, a mobile user i may trust a mobile user j if $x_{i,j} \geqslant x^{thrd}$. In this work, our objective is to minimize the energy consumption at mobile terminal side. Thus, the overhead for maintaining the matrix can be ignored.

4 Case Study and Description of the Proposed Framework

In this section, we will first give a case-study for our socially aware hybrid computation offloading framework. Then, we will describe the SAHCO framework.

4.1 Case Study

For some type of mobile collaborative applications, multiple users in the same neighborhood typically look at the same scene, track the same environment, and need to recognize the same objects, so they can benefit from collaboration and data/resource sharing.

Fig. 2 illustrates an example of immersive application [2]. Note that such application can be split into several loosely coupled software components, as shown in Fig. 2a. Each component with its dependency, configuration parameters

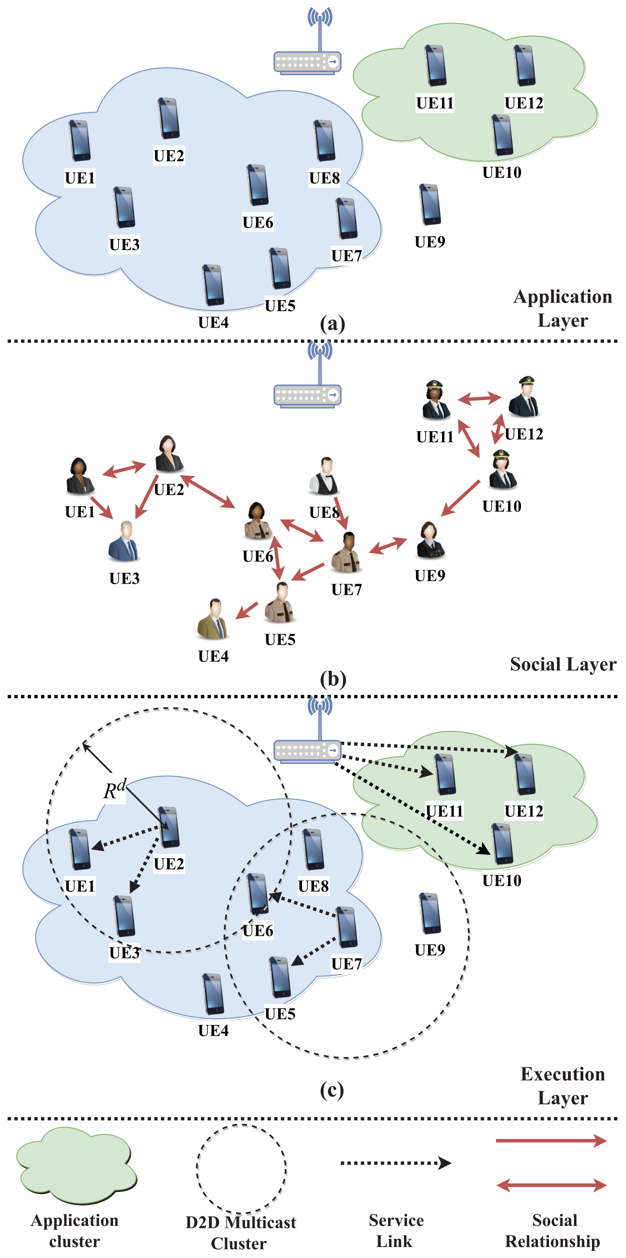


Fig. 3. Proposed socially aware hybrid computation offloading (SAHCO) system.

and constraints can be offloaded and shared among multiple users. Specifically, the Mapper component can be shared between multiple mobile devices in the same physical environment, as shown in Fig. 2b. All the mobile users receive the same world model (as shown in Fig. 2c.) to track the camera position and share the computation results (as shown in Fig. 2d). Because the model is updated and refined using camera images from multiple devices, the model will often be more accurate than the one created by just one device. On the one hand, through this computation/data sharing, the cloudlet agent allows users to not only save computational resources to avoid repeated calculation, but also to gain information from the input of others. On the other hand, for the sake of privacy, mobile users are unwilling to offload and share their sensitive data to other unfamiliar mobile user. Thus, it makes sense to design a novel energy-efficient computation offloading scheme with privacy consideration.

4.2 Description of the SAHCO Framework

At the beginning of each offloading decision making, the mobile devices upload the information required for offloading decision to the MEC server. This information relates to the offloading data, social relationship information, the mobile device and network characteristics.

Based on the received information, the MEC server first classifies the mobile users into multiple application clusters as shown in Fig. 3a. Each cluster is formed by users executing the same application and sharing their inputs to each other. The outputs (i.e., computation results) can be also shared among them.

In order to establish reliable D2D communications among the mobile users, the latter is allowed to share the resources and computation results to its trusted users. Thus, the MEC server builds a social trust graph (as showed in Fig. 3b) for the mobile users based on the received social relationship information. Then, it observes the current network state, computes the immediate costs (as will be explained in Section 5.2) of each component assignment strategy. Afterwards, based on these costs, it triggers the offloading decision process (as it will be explained in Section 5.1). The objective of the decision action is to select either: i) the MEC server or ii) the set of mobile users, able to compute the components. For example, Fig. 3c shows that UEs 2, 7 and the MEC server are selected. Next, based on the offloading decision, the remaining mobiles users offload their computation (i.e., input data) to their corresponding servers. Note that the latter can be either the MEC server (i.e., MEC offloading) or a nearby mobile user (D2D offloading). Accordingly, the selected mobile users are responsible for processing and sending computation results to the other users (i.e., the transmitter in each multicast cluster). Thus we refer to this kind of mobile users as *offloadee* in this paper, and the corresponding receivers as *offloader* (e.g., UEs 1 and 3 in the coverage of offloadee UE 2).

This process is repetitively executed until either reaching the energy budget threshold or exiting the component phase. In that case, the application has been executed in the application cluster. The output of our hybrid offloading framework is a sequence of fine-grained components assignment strategy for each component of an application. Therefore, our objective is to find the optimal components assignment strategy that can minimize the overall mobile users' energy consumption in an application cluster.

5 Problem Formulation

In this section, we will describe our socially aware hybrid computation offloading problem. We first introduce the state space and action space of our problem, followed by a description of immediate cost. Finally, we model the computation offloading as task assignment problem where the objective is to find the best sequence of component assignment policies for each application cluster.

5.1 State Space and Action Space

Assume that a set $\mathcal{M}^c$ ($\mathcal{M}^c \subset \mathcal{M}$) of mobile users in an application cluster runs application $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The state space of mobile users in an application cluster is defined as follows:

$$\xi = \{S = (V, Q, X) \in \mathcal{S} | V \subset \mathcal{V}, Q \in \mathcal{Q}, X \in \mathcal{X}\}, \tag{9}$$

where $V = \{v^o, v^n, (v^o, v^n \in \mathcal{V})\}$ denotes the set of component phases to be executed by the mobile users in the application

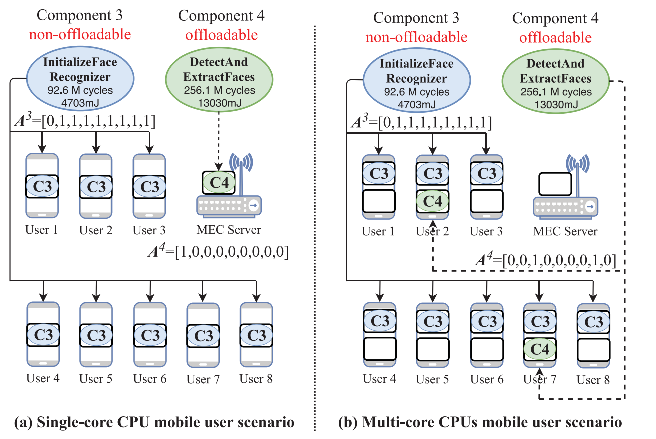


Fig. 4. Single/Multi-core CPU mobile user scenarios comparison for SAHCO system.

cluster. Note that v^o and v^n denote respectively the set of offloadable and non-offloadable components. $Q = [q_1, q_2, \ldots, q_m, \ldots, q_{|\mathcal{M}_c|}]_{1\times|\mathcal{M}_c|}$ $(m \in \mathcal{M}_c)$ represents the matrix of battery level for the mobile users, where q_m is the battery level of the mobile user m. $X = [x_{i,j}]_{|\mathcal{M}_c|\times|\mathcal{M}_c|}$ $(i, j \in \mathcal{M})$ is the social relationship weight matrix for the mobile users in the cluster.

The action space is $\phi = \{\mathcal{A} = \{A^v, v = 1, 2, \ldots, |V|\}\}$, where $\mathcal{A}$ is the component assignment (i.e., offloadee selection) strategy. Formally, $A^v = [a_0, a_1, a_2, \ldots, a_m, \ldots, a_{|\mathcal{M}_c|}]_{1\times|\mathcal{M}_c+1|}$ $(a_m \in \{0,1\}, v \in V, m \in \mathcal{M}_c)$, is the component assignment matrix for the v^{th} component. If $a_0 = 1$, then the component v should be allocated to the MEC server, and $a_0 = 0$ otherwise. $a_i = 1$ $(i \neq 0)$ indicates that the component v should be allocated to the i^{th} mobile user (i.e., selected as offloadee). Otherwise, $a_i = 0$ $(i \neq 0)$, i.e., component v is not allocated to mobile user i. For the non-offoadable components, we let $A^v = [0, 1, 1, \ldots, 1]$ denote the local execution strategy.

For example, consider the blue cluster in Fig. 3. Assume that the mobile users in the cluster execute a face recognition modeled with a task graph as in Fig. 1. Suppose that for the current state, the components 3 (non-offloadable) and 4 (offloadable) have to be executed in parallel. The social relationship corresponding to the current state is depicted in Fig. 3b. Thus, both the social relationship weight and the battery level matrices, X and Q respectively, are obtained. Then, the current component phase is given by $V = \{\{3\}, \{4\}\}$. Accordingly, based on the observed composite state $S = (V, Q, X)$, the MEC server makes an action $\mathcal{A}$ of component assignment for the components.

Single-Core CPU Mobile User Scenario. When the mobile users in the cluster are equipped with single-core CPU while processing non-offloadable components (i.e., component 3), there will be no enough processing units to process the remaining components (i.e, components 4), as shown in Fig. 4a. Consequently the action for component 3 is $A^3 = [0, 1, 1, \ldots, 1]$. The remaining components (i.e., components 4) can, hence, be offloaded to the MEC server. The actions for components 4 are $A^4 = [1, 0, 0, \ldots, 0]$. Thus, the action associated to components assignment for the current state is $\mathcal{A} = \{A^3, A^4\}$. Note that each mobile user in the same cluster can process only one component at one time. Therefore, the action A should satisfy the following constraint:

$$\sum_{v\in v^o} A^v_{[1,m]} \begin{cases} \leqslant D_b, \text{if} \quad m = 1, \\ \leqslant 1, \text{if} \quad m = 2, 3, \ldots, |\mathcal{M}_c + 1|, \end{cases} \tag{10}$$

where D_b denotes the maximum number of components that can be simultaneously processed by the MEC server. In virtual machine (VM) based cloudlet system, D_b also denotes the number of VMs that a MEC server can provide.

Multi-Core CPUs Mobile User Scenario. Nowadays, in order to further increase the performance and extend battery life, mobile devices will transit to multi-core CPUs [26]. This kind of CPU architecture enables the mobile devices to execute multiple components at the same time. Consider the current state with component phase $V = \{\{3\}, \{4\}\}$ in multi-core CPUs mobile user scenario. Fig. 3c illustrates a component assignment matrix A^4 for component 4. If the offloading decision algorithm run by the MEC server selects mobile users 2 and 7 as offloadees to execute component 4, as shown in Fig. 4b, then $A^4 = [0, 0, 1, 0, 0, 0, 0, 1, 0]$. Accordingly, the action of component assignment for the current state is $\mathcal{A} = \{A^3, A^4\}$. After finishing the processing, both the MEC server and the nominated offloadees send the computation results to the trusted offloaders through D2D multicast links. Note that the latter should be in their coverage area. For example, an offloadee mobile user 2 broadcasts the results of component 4 to his trusted offloader mobile users 1 and 3 in his coverage, as shown in Fig. 3c. Similarly to the single-core scenario, the action $\mathcal{A}$ in multi-core CPU mobile user satisfies the following limitation:

$$\sum_{v\in v^o} A^v_{[1,m]} \begin{cases} \leqslant D_b, \text{ if} \quad m = 1, \\ \leqslant D_u, \text{ if} \quad m = 2, 3, \ldots, |\mathcal{M}_c + 1|, \end{cases} \tag{11}$$

where D_u denotes the maximum number of components that a mobile user can process (i.e., number of CPU cores) at the same time.

5.2 Immediate Cost

In order to evaluate the total energy consumption of the mobile users, we define an *immediate cost* $C(S, \mathcal{A})$ for the current state $S = (V, Q, X)$, under a given assignment action $\mathcal{A}$, as follows:

$$C(S, \mathcal{A}) = C_l(S, \mathcal{A}) + C_r(S, \mathcal{A}), \tag{12}$$

where $C_l(S, \mathcal{A}) = \sum_{v\in v^n, m\in\mathcal{M}_c} E^{local}_{m,v}$ is the immediate local execution cost of non-offloadable components by mobile users (equals to the sum of local energy consumption. $C_r(S, \mathcal{A})$ defined by Eq. (13), denotes the immediate offloading cost for the offloadable components.

Let $\mathcal{M}^b_c$ denotes the set of offloadees that are served by the MEC server in the application cluster. Thus, the first term in Eq. (13) denotes the total energy consumption for the offloaders to transmit their input data to the MEC server through cellular offloading. Considering the D2D offloading, let $\mathcal{M}^s_c$ denote the set of offloadees and $\mathcal{M}^d_c(j)$ the set of offloaders for the j^{th} offloadee in $\mathcal{M}^s_c$. Therefore, the second term in Eq. (13) refers to the total energy consumption required by the offloaders to transmit their input data to nearby offloadees through D2D offloading. The third term denotes the energy consumption for the selected offloadees to execute the component v. Next, the fourth term represents the energy required by the offloadees while broadcasting the computation results to their offloaders. Note that some kinds of mobile users can not be served by a offloadee, if either: i) they are not in the same communication range (e.g., mobile user 4 in Fig. 3c), or ii) the offloadees lack of social trust (e.g., mobile user 8 in

$$C_r(S,\mathcal{A}) = \sum_{v\in v^o,\mathcal{E}_{u,v},\mathcal{E}_{v,w}\in\mathcal{E}} \left(\sum_{m\in\mathcal{M}_c^b} p_m^u \cdot \frac{|\mathcal{E}_{u,v}|}{r_m^{ul}} + \sum_{j\in\mathcal{M}_c^s, i\in\mathcal{M}_c^d(j)} p_i^{d2d} \cdot \frac{|\mathcal{E}_{u,v}|}{r_{i,j}^{d2d}} + \sum_{m\in\mathcal{M}_c^s} p_m^c \cdot \frac{\phi_v}{f_m^u} + \sum_{m\in\mathcal{M}_c^s} p_m^{d2d} \cdot \frac{|\mathcal{E}_{v,w}|}{r_m^{d2d}} + \sum_{m\in\mathcal{M}_c^l} p_m^c \cdot \frac{\phi_v}{f_m^u} \right). \tag{13}$$

Fig. 3c). Thus, these kinds of mobile users must process component locally. Let $\mathcal{M}_c^l$ represents the set of these kind of mobile users. Accordingly, the last term in Eq. (13) denotes the corresponding energy consumption for local execution.

For some typical computation-intensive applications, e.g., virus scan and image retrieval, the size of results sent back in the downlink is much smaller than that in the uplink, and the ratio could be as low as 1/30 [27]. Besides, downlink transmission rate is generally larger than uplink transmission rate, e.g., 2 times in LTE standards. Hence, the downlink transmission delay is much smaller than uplink transmission delay. Moreover, according to [28], the power consumption of uplink transmission is about 5.5 times as much as that consumed in the downlink reception. Combining with uplink delay and downlink delay, we could obtain that energy consumption of downlink reception is only about 0.003 as much as that of uplink transmission. Thus, we focus, in this work, on the uplink energy consumption at mobile terminal side, and ignore the downlink and computation energy consumption at the MEC server side. In fact, the MEC server is usually located at fixed areas with stable power supply.

5.3 Optimal Problem Formulation

The main objective of task the assignment problem for a mobile application is to find the offloading decision for the set of components in the task graph. To do so, all the mobile tasks are processed together instead of deciding consecutively for each component. Therefore, our objective is to find the best sequence of component assignment policy for an application in each cluster. We first define a component assignment policy $\pi = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{|S|}\}$ $(\pi \in \pi^f \subset \phi)$ as *feasible* if the total induced cost is lower than a threshold energy value. In other words, a *feasible* assignment policy should meet the following constraint:

$$\sum_{\mathcal{A}\in\pi} C(S,\mathcal{A}) \leqslant E_{thrd}(\mathcal{G}), \tag{14}$$

where ϕ denotes the action space and π^f denotes the set of feasible actions. $E_{thrd}(\mathcal{G}) = \sum_{v\in\mathcal{V}, m\in\mathcal{M}_c} E_{m,v}^{local}$ is the overall energy consumption induced by the mobile users in the cluster to execute application $\mathcal{G}$ locally. In consequence, the optimal component assignment sequence (i.e., policy) denoted by $\pi^* = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_{|S|}\}$ $(\pi^* \in \pi^f \subset \phi)$ minimizing the overall system cost, is given by:

$$\begin{aligned} \pi^* &= \arg\min_{\pi\in\pi^f\subset\phi} \sum_{\mathcal{A}\in\pi} C(S,\mathcal{A}). \\ s.t. \quad & E_m(\mathcal{G}, pi^*) \leqslant \sum_{v\in\mathcal{V}} E_{m,v}^{local}, \\ & T_m(\mathcal{G}, pi^*) \leqslant \sum_{v\in\mathcal{V}} T_{m,v}^{local}, \end{aligned} \tag{15}$$

where $E_m(\mathcal{G}, pi^*)$ and $T_m(\mathcal{G}, pi^*)$ are the total energy consumption and delay for mobile user m to process application $\mathcal{G}$ under the task assignment policy pi^*. It indicates that the total energy consumption of mobile user m through task assignment policy pi^* must be less than the total energy consumption of local execution. At the same time, the corresponding execution delay must be less than the local execution delay.

It is worth noting that the system cost $\sum_{\mathcal{A}\in\pi} C(S,\mathcal{A})$ is a long-term cost, which is not provided immediately, but it is iteratively constituted to be obtained when all the components have been processed. This cost is formally named as *delayed cost*. Note that the optimization problem in Eq. (15) is integer programming problem which is NP-hard [29]. To get rid of such challenge, we aim to design a low-complexity solution able to achieve an optimized component assignment policy while reducing computation time. In order to compute the best sequence of task assignment policy along the search tree $\mathcal{T}$, we propose a new *task assignment* solution based on Monte Carlo Tree Search algorithm [11], named, *TA-MCTS*. We model the sequences of task assignment as a decision tree denoted by $\mathcal{T}$ defined as follows. Each node in $\mathcal{T}$ simulates the assignment decision of the current system state. We recall that the latter represents a possible system state including the components to be executed, the set of mobile users, the social relationships and the other network information as defined above. The link between one node S in the tree and its child node S' indicates the task assignment policy $\mathcal{A}$ for components included in the current system state (i.e., of node i). Our approach, detailed in the next section, computes an optimized task assignment sequence with a lower complexity.

6 Proposal: *TA-MCTS* Scheme

In this section, we will present our approach named *task assignment solution based on Monte Carlo Tree Search algorithm* (*TA-MCTS*), as shown in Fig. 5. Our optimization problem Eq. (15) is a mixed integer programming problem. It can be indeed relaxed based on some convex relaxation techniques proposed in literature such as Lagrangian relaxation, and continuous relaxation. However, such method is still an approximate solution to our original problem, as in our proposed *TA-MCTS* heuristic.

As we have explained above, the social aware hybrid computation offloading problem is modeled as a search tree problem. However, the main challenge when tackling such a problem is the scalability. In fact, although the solution space is initially of finite size, it may heavily increase following the number of components and the network size. Consequently, the problem becomes computational intractable for large-scale instances. In literature, several algorithms are investigated to solve the search tree problem, such as: *A** [30], *Best-First-Search* [31], etc. Unfortunately, these classical approaches struggle to resist the scalability constraint and are unable to converge to an optimized solution in a reasonable time. To efficiently handle the *SAHCO* problem while considering the dimension challenge, our proposed solution *TA-MCTS* makes use of Monte Carlo Tree Search optimization algorithm [11]. MCTS is a heuristic search method for making optimal decisions in decision making process. The main idea of our approach is to find the best ordered sequence of task offloading for the set of components in the mobile application. Basically, it combines the tree building with optimum search process. In doing so, the size of solution space is importantly alleviated since the search tree does not need to be fully built. Note that MCTS is an efficient tool since it has been widely applied in computer games as well as some combinatorial problem [29]. We recall that the objective of our approach is to

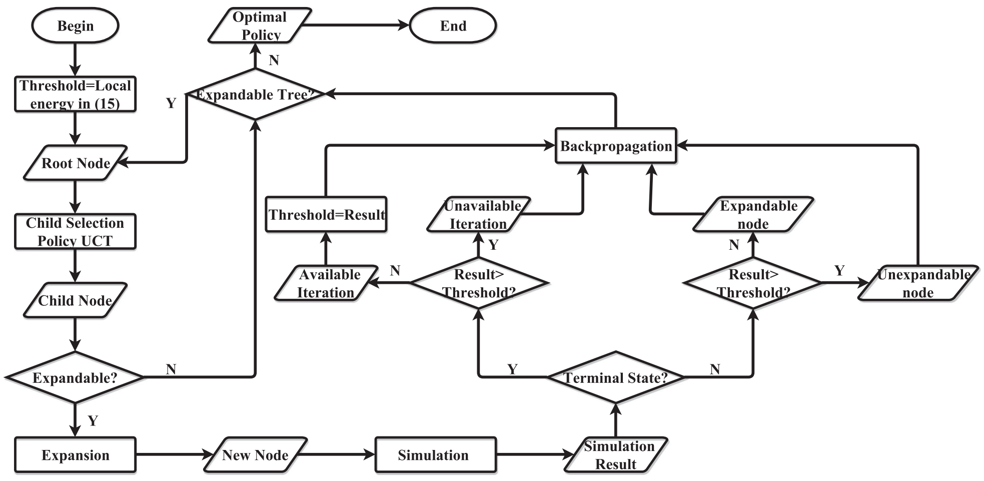


Fig. 5. Flowchart of proposed *TA-MCTS* approach.

determine the *best sequence* of task assignment for the different components of a mobile application in the network while minimizing the overall energy consumption. We can consider this problem as a *Single Player* game which can be efficiently solved using MCTS algorithm. Its objective is to minimize the energy consumption of task computation while satisfying the latency constraint. Note that each node in the tree corresponds to an action of a component in the application task graph. *TA-MCTS* iteratively builds the decision tree $\mathcal{T}$ and searches for the optimized task assignment sequence along the best branch in $\mathcal{T}$. To do so, our approach repetitively performs four stages, as shown in Fig. 6. First, the *Selection of the node to explore*. Second, the *expansion stage* to generate the sub-branch. Third, the *Simulation stage* by updating the node relevance function. Fourth, the *backpropagation stage* to propagate the cost value to all the ancestor nodes until reaching the root. These steps are iteratively repeated until either reaching the energy budget or all the components are assigned. Finally, the best sequence is hence obtained. It is worth noting that each iteration of our approach corresponds to a component assignment policy π. The numerator for the weight in each node of Fig. 6 denotes the number of *feasible* visit, whereas the denominator represents the number of total visit for the node in the historical iterations. Hereafter, we will describe the different steps of the *TA-MCTS* strategy.

6.1 Selection

First, an abstract empty node is considered as the root of the search tree. This root node means that no task of the mobile application is assigned. For the first iteration, the energy threshold value E_{thrd} is set to be $E_{thrd}(\mathcal{G})$, as shown in Eq. (14). The key insight of this stage is to select a node within the search tree in order to be further expanded. To do so, the idea is to choose the expandable node with the potential promoting branches while minimizing the number of explored branches. Note that a node is expandable if it represents a non-terminal state and has unvisited (i.e., unexpanded) children.

The main difficulty in selecting child candidate nodes is maintaining the balance between the exploitation of deep variants after moves with high average win rate, and the exploration of moves with few simulations. On the other hand, *TA-MCTS* should not explore only the promising nodes so as to avoid the local optimum. In order to efficiently guide the selection, we associate to each node in the search tree a relevance function, usually referred as UCT function (i.e., UCB 1 applied to Trees) [32]. Typically, the UCT value should incarnate the attractiveness of the node to be selected. Let: i) $\mathcal{I}$ denote the set of child nodes which are reachable from parent node p, ii) n_i represent the number of visits of node i (i.e., the number of sequences passing through node i) and iii) n_p the number of visits of the parent p (i.e., predecessor of i). Conventionally, a maximum UCT tree policy computes the UCT values for all of the node's children and then selects the child node with the maximum value. Note that if the maximum is shared by more than one child, then one of these children is selected arbitrarily. In this paper, our objective is to minimize the total energy consumption of task computation. Thus, *TA-MCTS* strategy selects the node by maximizing the UCT function [32] as:

$$\mathrm{UCT} = -y_i + C_p\sqrt{\frac{2\ln n_p}{n_i}}, \tag{16}$$

where $C_p > 0$ is a constant, y_i is the average immediate cost resulting from the set of sequences passing through child node i in previous iterations. It is straightforward to see that $n_i = 0$ yields a UCT value of infinity. Consequently, previously unvisited children nodes are assigned the largest possible value, and, hence, will be considered at least once before any child is expanded. In doing so, *TA-MCTS* avoids local optimum.

6.2 Expansion

During this step, *TA-MCTS* expands the selected node n_s. To do so, a new child of n_s is randomly generated and added to the tree $\mathcal{T}$. It is worth noting that randomness ensures the convergence of our solution to an upper bound in a polynomial time [11]. Note that only one child node is added during each iteration, and it represents the system state under specific actions. As shown in Fig. 6b, the algorithm is currently at the $2/2$ node, so there is a child node added onto that node indicated by the node with the $0/0$.

6.3 Simulation

In this step, the simulation of the new node is performed by choosing possible moving. This process is repeated until either: i) reaching a final state or ii) a predefined energy threshold E_{thrd} is reached. Note that a simulation is run from the new node according to a simulation policy to produce an outcome. In this work, the used simulation policy calculates

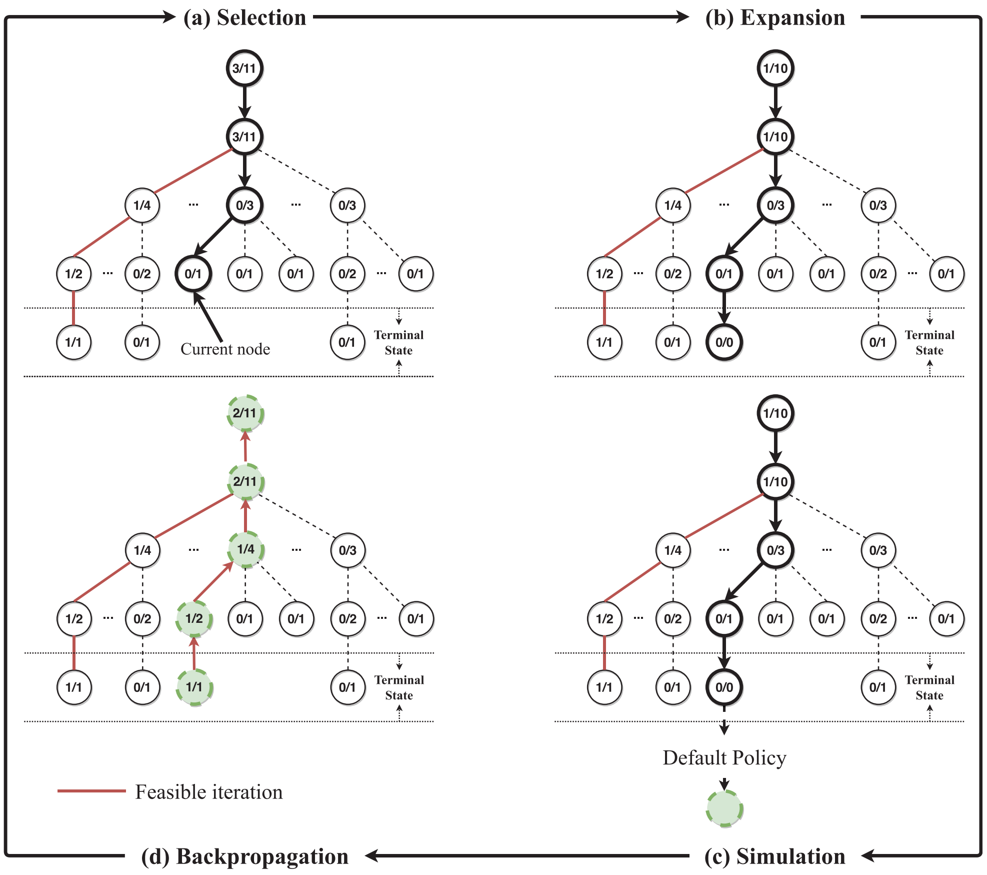


Fig. 6. One iteration of the proposed *TA-MCTS* approach.

the sum of the so far immediate cost for the current iteration and compares it to the predefined threshold. When the current node is terminal state, the sum of so far immediate cost is also the delay cost for current iteration. Then, based on the result of the simulation, the weight of the newly added node is established. Note that an iteration is feasible if it meets both of the following constraints: (i) the terminal state is reached and the delay cost is achieved, and (ii) the delay cost is less than the current energy threshold E_{thrd}.

6.4 Backpropagation

Once the new sub-branch $\mathcal{B}$ is generated and added to the tree, the UCT function value should be updated for all the nodes in the sequence containing $\mathcal{B}$. As shown in Fig. 6d starting at the new node, the algorithm traverses back to the root node. During the traversal process, the number of simulations stored in each node is incremented. If the new simulated node is feasible, then the number of feasible visits is also incremented. This step ensures that the values of each node accurately reflect simulations performed in their corresponding sub-trees

If an iteration is feasible, the component assignment policy π and corresponding delayed cost are the so far local optimal solution. The energy threshold in the next iteration is set to be the delayed cost in this feasible iteration. The algorithm keeps iterating until the search tree is unexpandable. The global optimal component assignment policy π^* and corresponding minimized energy consumption are thus achieved.

6.5 Complexity Analysis

Indeed, the dimension of the solution space for our problem in Eq. (15) would heavily increase and the number of all possible component assignment policies is $2^{(1+|\mathcal{M}^c|+|\mathcal{V}|)}$ for an application cluster, where $|\mathcal{M}^c|$ denotes the number of mobile users in the cluster, $|\mathcal{V}|$ denotes the number of components for the application and the "1" refers to the MEC server. To reduce the complexity, we first classified the components into offloadable and non-offloadable, in order to: i) save energy, and ii) reduce the complexity. In doing so, the search space is reduced to $2^{(1+|\mathcal{M}^c|+|v^0|)}$ $(|v^0| < |\mathcal{V}|)$. Then, we consider a socially aware offloading framework, where the components are assigned within trusted mobile users. Note that in this way, we can establish reliable D2D links for the mobile users, and the search space will be further reduced to $2^{(1+|\mathcal{M}_c^s|+|\mathcal{M}_c^d|+\ |v^0|)}$ $(|\mathcal{M}_c^s|+|\mathcal{M}_c^d| < |\mathcal{M}^c|)$. As a result, the complexity of our *TA-MCTS* strategy is $\mathcal{O}(1)$ in the best case, and $2^{(1+|\mathcal{M}_c^s|+|\mathcal{M}_c^d|+\ |v^0|)}$ in the worst case. In practice, the complexity is decided by the iteration times. The runtime of the algorithm can be simply be computed as $\mathcal{O}(I_t K_p 2^{1+|\mathcal{M}_c^s|+}\ |\mathcal{M}_c^d|/C_o)$ [33], where $2^{1+|\mathcal{M}_c^s|+|\mathcal{M}_c^d|}$ is the number of random children to consider per search and K_p is the number of parallel searches, and I_t is the number of iterations and C_o is the number of cores available.

7 Performance Evaluation

In this section, we assess the performance of our proposed approach *TA-MCTS*, based on extensive simulations. First of all, we will describe our simulation environment. Then, we define the performance metrics used to gauge our proposal and the related strategies. Finally, we compare the performance of our approach with the most prominent related strategies while considering different scenarios.

TABLE 1
Network Parameters

Parameter	Value	Parameter	Value
B_m	1 M	$\zeta_1, \zeta_2, \zeta_3$	1/3
λ	5×10^{-3}	N_0	5×10^{-5}
g_{dl}, g_{ul}, g_{d2d}	10^{-3}	x^{thrd}	0.4
p^c_m	(0, 0.9] W	p^u_m	0.1 W
f^{max}_u	10^9 cycles/s	f_b	10^{10} cycles/s
R_d	10 m	C_p	$\sqrt{2}/2$

7.1 Simulation Environment

As proposed in [34], we assume that the geographical distribution for the mobile users is thus given by:

$$P(n, r) = \frac{[\lambda L(r)]^n}{n!} e^{-\lambda L(r)}, \quad (17)$$

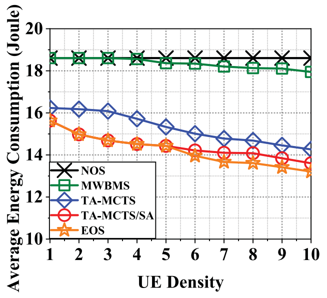
where $L(r) = r^2$ denotes the area of the network with a radius r. In our simulations, we set r to 100 meters. λ represents the mean density of mobile users.

Note that in our simulation, different devices in the network can have different capabilities. We assume that the maximum computation and battery capacity of all the mobile users are fixed to f^{max}_u and q^{max}_u, respectively. For mobile device m ($m \subset \mathcal{M}$), its computation capacity f^u_m is uniformly selected from the set $\{0.1, 0.2, \ldots, 1.0\} \times f^{max}_u$ cycles/s, similar to [24]. Its battery level q_m follows a uniform distribution in the range (0, 1]. We also assume that the local computing power p^c_m for mobile user m follows a uniform distribution in the range (0, 0.9] W.

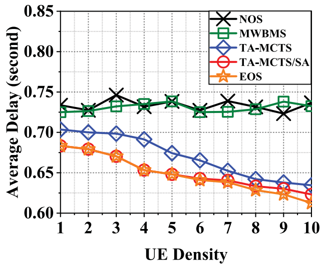
To assess the performance of our task assignment proposal for realistic MCA, we consider, in this work, an example of wearable mobile application. The task graph of this MCA is composed of 8 random tasks connected as in Fig. 1. Note that we generate the set of data dependency link $\mathcal{E}_{u,v}$ based on a uniform distribution in the range of [100, 500] KB. The values ω of required numbers of CPU cycles per bit (cpb) for mobiles components follow the uniform distribution in the range of [4000, 12000] cpb as in [24].

Afterwards, we implemented our approach *TA-MCTS* and compare it with respect to the most relevant benchmark related policies, namely:

- *Exhaustive-based Offloading Scheme (EOS):* Search the optimal solution through exhaustive approach.
- *No Offloading Scheme (NOS):* Local execution, which means that all the tasks of the MCA are locally executed on the mobile device. Typically, the assignment policies correspond to $\mathcal{A} = \{A^v = [0, 1, 1, \ldots, 1], v = 1, 2, \ldots, |V|\}$.
- *Random Component Assignment Scheme (RCAS):* Represents a random component assignment strategy, where each component assignment policy $A^v (v = 1, 2, \ldots, |V|)$ corresponds to randomly generated binary matrices, while still satisfying constraints in Eqs. (10) and (11).
- *TA-MCTS without social-award (TA-MCTS/SA):* Corresponds to our hybrid task assignment strategy based on MCTS, without considering social relationship (i.e., $x^{thrd} = 0$).
- *Minimum Weighted Bipartite Matching-based Scheme (MWBMS):* Represents to our previous offloading strategy [1], where task assignment is sequentially processed for each component in the graph instead of considering simultaneously all the components. Moreover, we considered only a multicast-based computation offloading scenario. Particularly, our proposed approach investigates the concepts of both minimum weighted bipartite matching and coalitional game to solve the optimal offloading problem.



(a) Average energy consumption vs. UE density λ (10^{-3})



(b) Average delay vs. UE density λ (10^{-3})

Fig. 7. *TA-MCTS* performance for the face recognition application.

7.2 Simulation Setup

To compare to our previous MWBMS, we first consider the MEC server equipped with 2-cores CPU (i.e., $D_b = 2$). Thus, the computation capacity of each core is $f^u_m/2$. Each mobile user equipped with only one CPU core (i.e., $D_u = 1$). We also set that social relationship threshold x^{thrd} to 0.4. It is worth noting that each performance value of the implemented strategies is the average of 1000 simulations.

In each realization, the iteration times for *TA-MCTS* is limited to be 2000. The parameters used in our simulations are reported in Table 1

7.3 Simulation Results

To assess the efficiency of our proposal, we consider four main scenarios. In the first one, *Hybrid-Offloading* scenario, we compare our hybrid task assignment approach *TA-MCTS* to the related strategies under different mobile user density. In the second scenario, *Offloading-comparison*, we compare the performance of our strategy for different type of computations: i) D2D, ii) MEC, iii) hybrid and iv) local. In the third scenario, *Socially aware*, in which we assess the performance of our proposal while considering different social relationship threshold values. Finally, in the fourth scenario, *CPU-structure*, we evaluate the performance of our proposal while considering different CPU-structure (i.e., different number of CPU cores for mobile users and the MEC server).

7.3.1 Face Recognition Application Scenario

Fig. 7a and 7b show the energy consumption and delay of face recognition application for different mobile user density, respectively. Assume that each mobile device equips with one CUP core. Several observations can be made. *TA-MCTS* reduces the energy consumption by 23.33, 20.52 percent compared to NOS and MWBMS, respectively, as shown in Fig. 7a, while meets the delay constraints (i.e., the local execution delay, as shown in Fig. 7b). The EOS always achieves the best performance, but with high complexity. Note that the input data of component "DetectandExtractFaces" is large. Thus mobile devices prone to process the component locally. The energy consumption for processing the component accounts for nearly 85 percent of the total energy consumption at the mobile terminal side. Due to each mobile device equips

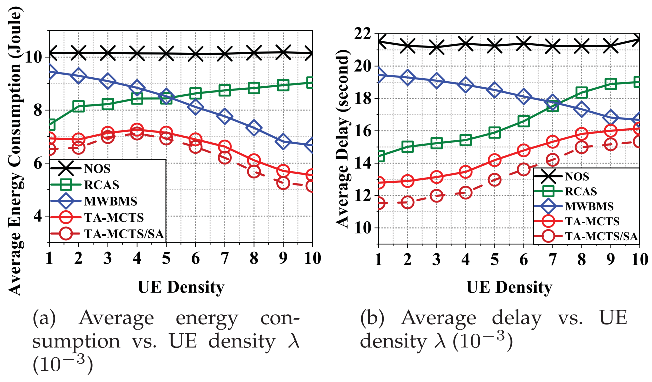


Fig. 8. *TA-MCTS* performance versus UE density λ (10^{-3}).

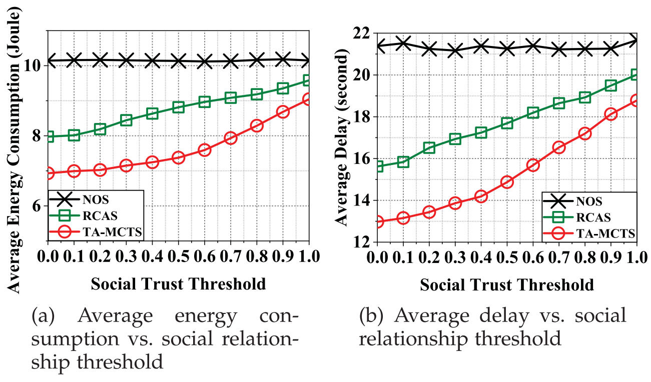


Fig. 10. *TA-MCTS* performance versus social relationship threshold.

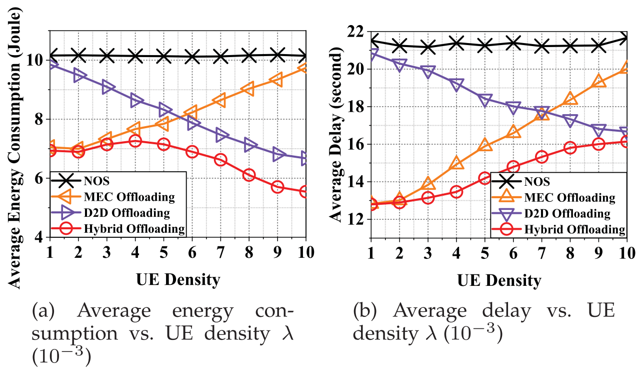


Fig. 9. *TA-MCTS* performance for different offloading strategies.

with one CUP core. Thus mobile devices prone to offload component "InitializeFaceRecognizer" to join our *TA-MCTS*, when they execute the "DetectandExtractFaces" locally.

7.3.2 Hybrid-Offloading Scenario

Fig. 8a and 8b show the average energy consumption and average delay for different mobile user density, respectively. Several observations can be made. First of all, we notice that our proposal *TA-MCTS* outperforms the related benchmark policies in term of energy saving. Note that this is observed for both cases: with or without social consideration. Consequently, our approach guarantees the delay constraints. In fact, *TA-MCTS* reduces the energy consumption by 45.37, 36.21 and 26.62 percent compared to NOS, RCAS and MWBMS, respectively. Note that the energy consumption increases slightly when the density grows from 0 to 0.4, and decreases when the density grows from 0.4 to 1. The reason is that, when the number of mobile users in an application cluster is small, the overhead for the users to create a D2D offloading group is high. As such, the mobile users are prone to perform MEC offloading. As the user number grows, the overhead decreases because more offloadees can save execution energy through *TA-MCTS* scheme. Thus, the mobile users perform hybrid offloading. Second, *TA-MCTS* without social-award (*TA-MCTS*/SA) saves 2.01-6.96 percent energy than *TA-MCTS* in energy saving, and 4.98-10.20 percent in delay reduction. This suggests that we sacrifice only 4.32 percent energy and 7.69 percent delay in average, to guarantee D2D links within effective and reliable mobile users. At last, the delay for *TA-MCTS* increases when the density grows. This is due to the fact that in each state, we consider the worst delay case as the state delay. It means that the system can not go to the next state until all the mobile users receive their computation result.

7.3.3 Offloading-Comparison Scenario

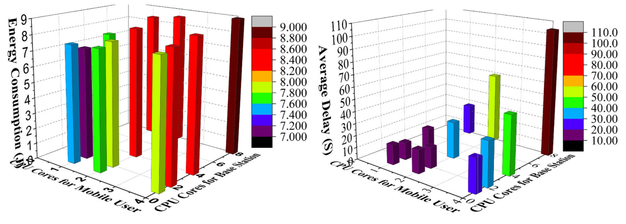
Then, we illustrate the performance of *TA-MCTS* approach for the *Offloading-comparison* scenario. Fig. 9 reports the *TA-MCTS* performance under different offloading strategies: i) hybrid, ii) MEC and iii) D2D offloading. MEC offloading means that mobile users offload their offloadable components to the MEC server. Whereas D2D offloading refers to the case where mobile users offload their offloadable components to the other mobile users. From this figure, we can see clearly that our hybrid offloading strategy performs the best. MEC offloading strategy performs worse as mobile users density grows. This is due to the limited computation capacity of the MEC server. In fact, when the number of mobile user increases, more mobile users have to perform local execution. On the other hand, we notice that the D2D offloading strategy performs better as mobile user density grows. The reason is that more offloadees can save execution energy through *TA-MCTS* as number of mobile user increases. It is straightforward to see that when the user density is small, our hybrid offloading strategy performs similarly as MEC offloading. When the user density is large, our hybrid offloading strategy performs similarly as D2D offloading.

7.3.4 Socially Aware Scenario

In this scenario, we illustrate the performance of *TA-MCTS* under different social relationship threshold values as depicted in Fig. 10. The mobile users density is set to be $\lambda = 5 \times 10^{-3}$, which means that there are around 12 mobile users in an application cluster. Note that $x^{thrd} = 0$ means that the mobile users in the same application cluster trust each other (i.e., *TA-MCTS/SA*). On the contrary, $x^{thrd} = 1$ represents the case where mobiles users do not trust each other. In this case, only MEC offloading is performed. As illustrated in Fig. 10, we can clearly see that *TA-MCTS* performs worse as x^{thrd} grows. The reason is that, creating D2D offloading group becomes hard when the mobile users do not trust each other. Consequently, both the energy consumption and delay increase. Note that *TA-MCTS* performs more stable than the RCAS while x^{thrd} ranging from 0.2 to 0.6. Although the choices of server decrease as x^{thrd} increase, *TA-MCTS* always select the optimal nearby device as server for each single mobile user.

7.3.5 CPU-Structure Scenario

At last, the performance of our *TA-MCTS* approach under different CPU structures is given in Fig. 11. Note that our algorithm achieves the best performance when $D_u = 1$ and $D_b = 2$



(a) Average energy consumption vs. D_u and D_b

(b) Average delay vs. D_u and D_b

Fig. 11. *TA-MCTS* performance versus CPU structure (i.e., D_u and D_b).

TABLE 2
Jain's Fairness Index for Different Schemes

Scheme	worst	TA-MCTS/SA	TA-MCTS	MWBMS	Best
Index value	0.028	0.64	0.67	0.85	1

(i.e., single-core CPU mobile user and double-core CPU MEC server). This is because we consider that the mobile wearable application has at most 3 parallel processing components. Such CPU divide (i.e., $D_u = 1$ and $D_b = 2$) is suitable for the application execution through our hybrid offloading strategy. In addition, it can be observed from Fig. 11 that while different CPU structures have limited impact on energy consumption, they strongly impact the delay. In fact, as D_u and D_b increase, the computation capacity for each single core (i.e., $\frac{f_m^u}{D_u}$ and $\frac{f_b}{D_b}$) becomes weaker. In this paper, we consider that each component is assigned to one single CPU core. Thus, the execution delay grows significantly.

7.3.6 Fairness Scenario

At last, we use the Jain's fairness index [35] to evaluate how fairly the offloadee selection (energy loss) among the m users. It is expressed as follows: $\mathcal{J}(m) = (\sum_{i=1}^{m} x_i)^2/(m \cdot \sum_{i=1}^{m} x_i^2)$, where x_i represents the remaining battery level of UE i after processing his component using our algorithm. Note that the index ranges from $1/m$ (worst case) to 1 (best case). Assume that the density of mobile users is 5×10^{-3} (i.e., around 35 mobile users in the network). Table. 2 illustrates the Jain's fairness index of different offloading schemes. Note that the *TA-MCTS* proposed in this work achieves an index of 0.67.

8 Conclusions

In this paper, we have studied the component assignment problem with social relationship consideration for multi-access edge computing. We proposed an energy efficient task assignment approach based on Monte-Carlo search tree, named *TA-MCTS*. First, we introduced our system model as a socially aware D2D computation offloading (SAHCO) system in detail. Then, in order to minimize the overall energy consumption at mobile terminal side, we formulated the optimal problem. Our objective is to let the MEC server makes an optimal sequence of component assignment policy for an application cluster. At last, we proposed a new optimal task assignment approach *TA-MCTS* based on Monte-Carlo search tree to solve the optimization problem. Compared to alternative solutions in literature, our proposed approach achieves the best performance. Specifically, *TA-MCTS* approach can reduce at most 45.37, 36.21 and 26.62 percent energy consumption than NOS (No Offloading Scheme), RCAS (Random Component Assignment Scheme) and MWBMS (Minimum Weighted Bipartite Matching-based Scheme), respectively. At the same time, we sacrifice only 4.32 percent energy and 7.69 percent delay in average, to guarantee that the D2D links among mobile users are reliable and effective.

Acknowledgments

This work was supported by the National Science Foundation of China (No. U1711265), the Fundamental Research Funds for the Central Universities (No.17lgjc40), the Program for Guangdong Introducing Innovative and Enterpreneurial Teams (No.2017ZT07X355), and the FUI PODIUM project (Grant no. 10703). A preliminary version of this paper appeared in the *Proceedings of the 2016 Global Communication Conference* (*GLOBECOM 2016*) [1].

References

[1] S. Yu, R. Langar, and X. Wang, "A d2d-multicast based computation offloading framework for mobile edge computing," presented at the *Global Commun. Conf.*, Washington, DC, USA, Dec. 2016.

[2] T. Verbelen, P. Simoens, F. D. Turck, and B. Dhoedt,"Leveraging cloudlets for immersive collaborative applications," *IEEE Pervasive Comput.*, vol. 12, no. 4, pp. 30–38, Oct. 2013.

[3] F. Chi, X. Wang, W. Cai, and V. Leung, "Ad-hoc cloudlet based cooperative cloud gaming," *IEEE Trans. Cloud Comput.*, vol. 6, no. 3, pp. 625–639, Nov. 2015.

[4] X. Zhang, Z. Yang, W. Sun, Y. Liu, S. Tang, K. Xing, and X. Mao, "Incentives for mobile crowd sensing: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 54–67, Mar. 2015.

[5] T. Penner, A. Johnson, B. V. Slyke, M. Guirguis, and Q. Gu, "Demo: Transient clouds," presented at the *IEEE 6th Int. Conf. Mobile Comput. Appl. Serv.*, Austin, TX, USA, Nov. 2014.

[6] D. Datla, X. Chen, T. Tsou, S. Raghunandan, S. S. Hasan, J. H. Reed, C. B. Dietrich, T. Bose, B. Fette, and J.-H. Kim, "Wireless distributed computing: a survey of research challenges," *IEEE Commun. Mag.*, vol. 50, no. 1, pp. 144–152, Jan. 2012.

[7] D. Datla, X. Chen, T. R. Newman, J. H. Reed, and T. Bose, "Power efficiency in wireless network distributed computing," presented at the *IEEE 70th Veh. Technol. Conf. Fall*, Anchorage, AK, USA, Sep. 2009.

[8] P. Zhao, L. Feng, P. Yu, W. Li, and X. Qiu, "A social-aware resource allocation for 5g device-to-device multicast communication," *IEEE Access*, vol. 5, pp. 15 717–15 730, Jul. 2017. https://ieeexplore.ieee.org/document/7990484

[9] B. G. Ryder, "Constructing the call graph of a program," *IEEE Trans. Softw. Eng.*, vol. SE-5, no. 3, pp. 216–226, May 1979.

[10] Y. Meng, C. Jiang, H.-H. Chen, and Y. Ren, "Cooperative device-to-device communications: Social networking perspectives," *IEEE Netw.*, vol. 31, no. 3, pp. 38–44, May 2017.

[11] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Trans. Comput. Intell. AI Games*, vol. 4, no. 1, pp. 1–43, Feb. 2012.

[12] J. L. D. Neto, S. Yu, D. F. Macedo, J. M. S. Nogueira, R. Langar, and S. Secci, "Uloof: A user level online offloading framework for mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2660–2674, Nov. 2018.

[13] E. Cuervo, A. Balasubramanian, D. ki Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making smartphones last longer with code offload," in *Proc. 8th Int. Conf. Mobile Syst. Appl. Serv.*, Jun. 2010, pp. 49–62.

[14] ETSI. "Multi-access edge computing (mec)," 2018. [Online]. Available: https://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing

[15] L. Pu, X. Chen, J. Xu, and X. Fu, "D2d fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted d2d collaboration," *IEEE J. Select. Areas Commun.*, vol. 34, no. 12, pp. 3887–3901, Nov. 2016.

[16] X. Chen, B. Proulx, X. Gong, and J. Zhang, "Exploiting social ties for cooperative d2d communications: A mobile social networking case," *IEEE/ACM Trans. Netw.*, vol. 23, no. 5, pp. 1471–1484, Oct. 2015.

[17] X. Chen, B. Proulx, X. Gong, and J. Zhang, "Social trust and social reciprocity based cooperative d2d communications," in *Proc. 14th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2013, pp. 187–196. [Online]. Available: http://doi.acm.org/10.1145/2491288.2491302

[18] F. Boustanifar and Z. Movahedi, "A trust-based offloading for mobile m2m communications," in *Proc. Int. IEEE Conf. Ubiquitous Intell. Comput./Adv. Trusted Comput./Scalable Comput. Commun./Cloud Big Data Comput./Internet People/Smart World Congr.*, Jul. 2016, pp. 1139–1143.

[19] D. Lopez-Perez, A. Valcarce, G. de la Roche, and J. Zhang, "Ofdma femtocells: A roadmap on interference avoidance," *IEEE Commun. Mag.*, vol. 47, no. 9, pp. 41–48, Sep. 2009.

[20] G. Ku and J. M. Walsh, "Resource allocation and link adaptation in lte and lte advanced: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1605–1633, 2015.

[21] R. O. Afolabi, A. Dadlani, and K. Kim, "Multicast scheduling and resource allocation algorithms for ofdma-based systems: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 240–254, Feb. 2012.

[22] J. Kwak, Y. Kim, J. Lee, and S. Chong, "Dream: Dynamic resource and task allocation for energy minimization in mobile cloud systems," *IEEE J. Select. Areas Commun.*, vol. 33, no. 12, pp. 2510–2523, Dec. 2015.

[23] J. Jiang, S. Zhang, B. Li, and B. Li, "Maximized cellular traffic offloading via device-to-device content sharing," *IEEE J. Select. Areas Commun.*, vol. 34, no. 1, pp. 82–91, Jan. 2016.

[24] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Dec. 2016.

[25] W. H. Yuan and K. Nahrstedt, "Energy-efficient soft real-time cpu scheduling for mobile multimedia systems," in *Proc. ACM 19th ACM Symp. Operating Syst. Principles*, Oct. 2003, pp. 149–163.

[26] Z. Jiang and S. Mao, "Energy delay tradeoff in cloud offloading for multi-core mobile devices," *IEEE Access*, vol. 3, pp. 2306–2316, Nov. 2015. https://ieeexplore.ieee.org/document/7323792

[27] M. Chen, M. Dong, and B. Liang, "Resource sharing of a computing access point for multi-user mobile cloud offloading with delay constraints," *IEEE Trans. Mobile Comput.*, vol. 17, no. 12, pp. 2868–2881, Dec. 2018.

[28] TROPIC. "Tropic deliverable report d21: Scenario and requirements," Feb. 2013. https://www.ict-tropic.upc.edu/index.php?option=com_spcom&Itemid=88

[29] D. S. Hochbaum, Ed., *Approximation Algorithms for NP-hard Problems.* Boston, MA, USA: PWS Publishing Co., 1997.

[30] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, 1968.

[31] M. P. D. Schadd, M. H. M. Winands, H. J. van den Herik, G. M. J. B. Chaslot, and J. W. H. M. Uiterwijk, "Single-player monte-carlo tree search," in *Computers and Games*, H. J. van den Herik, X. Xu, Z. Ma, and M. H. M. Winands, Eds. Berlin, Germany: Springer, 2008, pp. 1–12.

[32] L. Kocsis and C. Szepesv Ãari, "Bandit based monte-carlo planning," in *Proc. 17th Eur. Conf. Mach. Learn.*, 2006, pp. 282–293.

[33] S. B. Y. Jin, "Cme 323, report." Feb. 2016. [Online]. Available: http://stanford.edu/rezab/classes/cme323/S15/projects/montecarlo_search_ tree_report.pdf

[34] S. Lee, R. Zhang, and K. Huang, "Opportunistic wireless energy harvesting in cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4788–4799, Sep. 2013.

[35] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *CoRR*, vol. cs.NI/9809099, 1998. [Online]. Available: http://arxiv.org/abs/cs.NI/9809099

Shuai Yu received the BS degree from the Nanjing University of Post and Telecommunications (NJUPT), Nanjing, China, in 2009, the MS degree from the Beijing University of Post and Telecommunications (BUPT), Beijing, China, in 2014, and the PhD degree from University Pierre and Marie Curie (now Sorbonne Universite), Paris, France, in 2018. He is now a post-doctoral research fellow with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His research interests include wireless communications, mobile computing, and machine learning.

Boutheina Dab received the MSc degree in networks and the engineering degree in computer science from the National School of Computer Science (Tunisia), in 2013 and 2012, respectively and the PhD degree in computer science from the University of Paris-Est, France, in 2017. She has been a postdoctoral fellow with Orange Labs, France, since November 2018. She was a postdoctoral research fellow with UPMC Sorbonne University from 2017 to 2018. Her main research interests include cloud and data center networking, wireless communications, mobile edge computing, network orchestration, and optimization.

Zeinab Movahedi received the MSc and PhD degrees in computer networks and telecommunications from the University of Pierre and Marie Curie (Paris 6), Laboratoire d'Informatique de Paris 6 (LIP6), Paris, France, in 2007 and 2011, respectively. She is currently an assistant professor with IUST. Her research interests include green communication, mobile cloud computing, software-defined networks, autonomic networking, wireless networks, network security, performance evaluation, and quality-of-service support.

Rami Langar received the MSc degree in network and computer science from UPMC, in 2002 and the PhD degree in network and computer science from Telecom ParisTech, Paris, France, in 2006. He is currently a full professor with the University of Paris Est Marne-la-Vallée (UPEM), France. Before joining UPEM, he was an associate professor with LIP6, University Pierre and Marie Curie (now Sorbonne Université) between 2008 and 2016, and a post-doctoral research fellow with the School of Computer Science, University of Waterloo, Waterloo, ON, Canada, between 2006 and 2008. His research interests include resource management in future wireless systems, cloud-RAN, network slicing in 5G, software-defined wireless networks, and mobile cloud offloading. He is involved in many European and National French research projects, such as MobileCloud (FP7), GOLDFISH (FP7), ANR ABCD, FUI PODIUM, FUI ELASTIC, and FUI SCORPION. He is chair of the IEEE ComSoc Technical Committee on Information Infrastructure and Networking (TCIIN), and co-recipient of the IEEE/IFIP International Conference on Network and Service Management 2014 (IEEE/IFIP CNSM 2014) best paper award.

Li Wang (S'08-M'14-SM'16) received the PhD degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2009. She is a full professor with the School of Electronic Engineering, BUPT, where she heads the High Performance Computing and Networking Laboratory. She is also with the Key Laboratory of the Universal Wireless Communications, Ministry of Education, China. She also held visiting professor positions with the School of Electrical and Computer Engineering, Georgia Tech, Atlanta, Georgia, from December 2013 to January 2015, and with the Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden, from August to November, 2015. Her current research interests include wireless communications, secure communications, cooperative networking, and distributed networking and storage. She has authored/coauthored two books: *Device-to-Device Communications* (Springer, 2016) and *Physical Layer Security* (Springer, 2017). She has been an editor for the *IEEE Transactions on Vehicular Technology* and an associate editor for *IEEE Access*, since June 2016. She is the symposium chair of IEEE ICC 2019 on Cognitive Radio and Networks Symposium, and chairs the special interest group (SIG) on Social Behavior Driven Cognitive Radio Networks for the IEEE Technical Committee on Cognitive Networks. She also served technical program committees of multiple IEEE conferences, including IEEE GLOBECOM, ICC, WCNC, and VTC over the years. She was the recipient of the 2013 Beijing Young Elite Faculty for Higher Education Award, the Best Paper Award at ICCTA 2011, Best Paper runner up fromWASA 2015, and the Best Paper Award from IEEE ICCC 2017. She has also been selected by the Beijing Nova Program (2018). She is a senior member of the IEEE.