Mobile Edge Computing for Task Offloading in Small-Cell Networks via Belief Propagation

Jun Li, Anping Wu, Shunfeng Chu, Tingting Liu, and Feng Shu
School of Electronic and Optical Engineering,
Nanjing University of Science and Technology, Nanjing, 210094, P. R. CHINA
Email: {jun.li,anping.wu,shunfeng.chu,shufeng}@njust.edu.cn, liutt@njit.edu.cn

***Abstract*—A large number of computation-hungry mobile applications have led to an ever-increasing computation demands. Mobile edge computing (MEC) has been considered as an emerging paradigm to alleviate the demand effectively by offloading the computationally intensive tasks from mobile devices (MD) to the adjacent MEC servers. It is expected that the quality of computation experience, e.g., the computing energy and the execution latency, can be greatly improved by the MEC. In this paper, we will investigate the computing task offloading problem via the MEC in the context of small-cell base-station (SBS) networks, where each SBS is equipped with an MEC server. Specifically, we first formulate the optimization problem to minimize the objective, i.e., the weighted sum of energy consumption and execution duration. The parameters to be optimized are the allocations of the MD's tasks to be offloaded to the MEC servers. Then we propose a novel belief propagation (BP) algorithm to optimize the task allocation in a distributed manner. Next, we develop the factor graph according to the network topology and decompose the object function into multiple local utilities to fit the factor graph. Finally, we transform local utilities into the estimations of marginal distributions and propose a distributed BP algorithm to solve the estimations. Simulations demonstrate that our BP algorithm can effectively approach the optimal solutions via exhaustive search.**

***Index Terms*—Mobile edge computing, task offloading, resource allocation, belief propagation**

I. Introduction

In recent years, computationally intensive mobile applications are gaining enormous attractions with the develop of smart mobile devices (MD). In turn, it becomes challenging for MDs to support these emerging applications due to its limited battery and computation resources [1]. Cloud computing, by offloading the computation-hungry applications, such as image recognition, from local MDs to the cloud servers, can dramatically save the power consumptions of the MDs from heavy computations, and thus has been widely adopted by many Internet services.

However, the long-distance data transmissions from user terminal to remote clouds may lead to a great latency in cloud computing systems. Mobile edge computing (MEC), which provides cloud computing capabilities within the radio access network (RAN), offers a new paradigm to liberate the MDs from heavy computation workloads [2]. By pushing the function of cloud towards the network edges in the vicinity of users, MEC has the potential to dramatically reduce latency and save buttery compared to conventional cloud computing architecture due to short-distance communications. When the offloaded task can be executed closer to the required mobile users, the users can directly fetch the offloading computing result from the server of SBSs, thereby achieving a better experience [3]. It is also shown in [4] that by offloading computational tasks to a physically proximate MEC server, the efficiency of task executions will be greatly improved.

A key issue needed to be solved in MEC systems is efficient resource allocation. To be specific, since task offloading introduces additional communication overhead between an MD and its associated MEC server, a technical challenge is how to balance the tradeoff between computational and communication costs to support applications with enhanced user experience, such as low latency and energy consumption. In [5], the authors focus on the design of an energy-efficient computation offloading mechanism for MEC system by energy-efficient computation offloading (EECO) algorithm. In [6], the authors exploit game theory to design efficient computation offloading mechanism for the MEC. Partial computation offloading optimization using dynamic voltage scaling is proposed in [1] to collaborate communication and computation resource. In [7], the authors consider energy-efficient resource allocation in a multiuser MEC system based on time-division multiple access (TDMA) and frequency-division multiple access (OFDMA). In [8], the authors design a Gini coefficient-based greedy heuristic (GCGH) to solve the the energy consumption minimization problem.

However, most of above works solve their optimization problems by centralized optimization algorithms. Generally, centralized solutions need a coordinator to collect the information from the entire network, which may limit the efficiency of the MEC framework. In contrast to the centralized algorithms, the distributed ones decompose the optimization problem into many small parts and allocate to multiple agencies for processing. Among many distributed solutions, the belief propagation (BP) algorithm has been widely adopted for distributively solving resource allocations in cellular networks [9].

In this paper, we propose a distributed method for the task offloading via MEC in the SBS systems. The network consists of multiple SBSs and MDs, where each SBS is equipped with an MEC server with finite computing capacity and each MD can only associate with one of the SBSs. We first formulate an optimization problem and establish the objective function, i.e., the minimum value of weighted sum of mobile energy and time consumption, subject to each SBS's computing capacity.

978-1-5386-3180-5/18/$31.00 ©2018 IEEE

The parameters to be optimized are the allocations of the MD's tasks to be offloaded to the MEC servers. Then we develop the factor graph according to the network topology and decompose the object function into multiple local utilities to fit the factor graph. We transform local utilities into the estimations of marginal distributions and propose a distributed BP algorithm to solve the estimations. Also note that the BP algorithm itself is computationally intensive. It is of low efficiency if the BP algorithm levies heavy computing tasks on the MDs. In our proposed BP, we enable the majority of the computationally intensive calculations to be executed on the MEC servers of the SBSs rather than on the MDs. Simulations show that the proposed BP algorithm can significantly reduce the weighted sum of energy and time consumption and approach the optimum solution with a negligible performance gap.

II. System Model

We now construct an MEC-based task offloading system drawn upon the small-cell network, which consists of I MDs and J SBSs. Denote by $\boldsymbol{\mathcal{U}} = \{\mathcal{U}_1, \mathcal{U}_2, \cdots, \mathcal{U}_I\}$ the set of MDs, and by $\boldsymbol{\mathcal{B}} = \{\mathcal{B}_1, \mathcal{B}_2, \cdots, \mathcal{B}_J\}$ the set of SBSs. Each MD has an intensively computational task to execute, while each SBS in the system is equipped with an MEC server. Part of the task in an MD can be offloaded to the MEC server for saving the energy of this MD. We assume that for $\mathcal{U}_i$, $\forall i \in \{1, 2, \cdots, I\}$, a task of Q_i-bit program needs to be executed, with the average CPU cycles for each bit in the task being denoted by C_i. Each MD can pick out part of its task and upload this part to the MEC server of it associated SBS for execution. An SBS will serve multiple MDs with its server, while each MD only associates with one SBS for task offloading.

We define an $I \times J$ offloading matrix $\boldsymbol{\Lambda}$, whose i-th row and j-th column entry $\lambda_{i,j} \in [0, 1]$, $\forall j \in \{1, 2, \cdots, J\}$, represents the ratio of offloaded computing amount of bits to the total bits of the task, i.e., a fraction of $\lambda_{i,j}$ of the entire task will be offloaded to the MEC server of SBS $\mathcal{B}_j$. Denote by $\boldsymbol{q}_i = \{q_{i,1}, \cdots, q_{i,n_i}, \cdots, q_{i,N_i}\}$ the set of the N_i options of $\lambda_{i,j}$, i.e., $\lambda_{i,j} \in \boldsymbol{q}_i$, where $q_{i,1} = 0$, $q_{i,N_i} = 1$, and $q_{i,n_i} < q_{i,n_i+1}$. We further use the row vector $\boldsymbol{\lambda}_i \triangleq [\lambda_{i,1}, \lambda_{i,2}, \cdots, \lambda_{i,J}]$ to represent the i-th row of the offloading matrix $\boldsymbol{\Lambda}$, and we can obtain $\boldsymbol{\Lambda} = [\boldsymbol{\lambda}_1^T, \cdots, \boldsymbol{\lambda}_i^T, \cdots, \boldsymbol{\lambda}_I^T]^T$. Note that an MD $\mathcal{U}_i$ can only associate with one SBS for offloading task. Therefore, there is at most one non-zero element in $\boldsymbol{\lambda}_i$. In the case that $\mathcal{U}_i$ chooses not to offload any part of its task to the MEC server, $\boldsymbol{\lambda}_i$ will be an all-zero vector.

Meanwhile, there is a restriction on the computation capability in each SBS server. In other words, for the SBS $\mathcal{B}_j$, the overall computation tasks executed by $\mathcal{B}_j$ cannot excess an upper-bound in a time slot. Let K_j denote the task upper-bound of $\mathcal{B}_j$'s MEC server within a time slot, and let f_j^{MEC} denote computing speed of the server, i.e., the cycles of this server's CPU per second.

A. Local Computing Model

Let f_i^{MD} be the computation capability of the MD $\mathcal{U}_i$, i.e., $\mathcal{U}_i$'s CPU cycles per second. We model the power consumption of $\mathcal{U}_i$'s CPU as $k(f_i^{\text{MD}})^3$ [10], where k is a coefficient of mobile chip. Thus, the energy consumption per cycle can be expressed as $k(f_i^{\text{MD}})^2$. The local computation execution time of $\mathcal{U}_i$ can then be given by

$$T_i^{\text{MD}} = \frac{\left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i}{f_i^{\text{MD}}}. \tag{1}$$

Furthermore, the energy consumption of $\mathcal{U}_i$, i.e., E_i^{MD}, caused by local computations is calculated as

$$E_i^{\text{MD}} = \left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i k \left(f_i^{\text{MD}}\right)^2. \tag{2}$$

B. Task Offloading Model

Since the duration for uploading the task is generally shorter than that for execution, the probability that two mobile devices occupy the designated channel for task offloading at the same time is extremely minor. Thus, we assume that there is no interference caused among the MDs during uploading the tasks. Let $g_{i,j}$ denote the channel gain from the mobile device $\mathcal{U}_i$ to the SBS $\mathcal{B}_j$, which is assumed to be constant within the offloading duration, and let P_i denote the mobile device $\mathcal{U}_i$'s transmission power. The uplink rate $R_{i,j}$ from the mobile device $\mathcal{U}_i$ to the SBS $\mathcal{B}_j$ can be expressed as

$$R_{i,j} = W \log_2 \left(1 + \frac{P_i g_{i,j}}{N_0}\right), \tag{3}$$

where W and N_0 denote the uplink channel bandwidth and Gaussian noise power, respectively. Thus, the upload time of mobile device can be expressed by

$$T_i^{\text{UL}} = \sum_{j=1}^{J} \frac{\lambda_{i,j} Q_i}{R_{i,j}}. \tag{4}$$

Additionally, the time duration for executing the offloaded task of $\mathcal{U}_i$ at the MEC server of $\mathcal{B}_j$ can be obtained by

$$T_i^{\text{MEC}} = \sum_{j=1}^{J} \frac{\lambda_{i,j} Q_i C_i}{f_j^{\text{MEC}}}. \tag{5}$$

Due to the transmission of the offloaded task, the energy consumption of each MD needs be taken into account. We have the $\mathcal{U}_i$'s energy consumption for task transmission as

$$E_i^{\text{UL}} = \sum_{j=1}^{J} \frac{\lambda_{i,j} Q_i}{R_{i,j}} \cdot P_i. \tag{6}$$

C. Overall Delay and Energy Consumption

We assume that an MD's CPU keeps idle during the execution of its uploaded task at the MEC server. Thus, the latency to execute the whole task of $\mathcal{U}_i$ can be given by

$$\begin{aligned} T_i^{\text{TOL}} &= T_i^{\text{MD}} + T_i^{\text{UL}} + T_i^{\text{MEC}} \\ &= \frac{\left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i}{f_i^{\text{MD}}} + \sum_{j=1}^{J} \left(\frac{\lambda_{i,j} Q_i}{R_{i,j}} + \frac{\lambda_{i,j} Q_i C_i}{f_j^{\text{MEC}}}\right). \end{aligned} \tag{7}$$

Besides, the whole energy consumption of $\mathcal{U}_i$ will be

$$\begin{aligned} E_i^{\text{TOL}} &= E_i^{\text{MD}} + E_i^{\text{UL}} \\ &= \left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i k (f_i^{\text{MD}})^2 + \sum_{j=1}^{J} \frac{\lambda_{i,j} Q_i}{R_{i,j}} \cdot P_i. \end{aligned} \tag{8}$$

III. Problem Formulation

In this section, we focus on the resource allocation problem for computing offloading. Our objective is to minimize the weighted sum energy and latency of all the MDs, i.e.,

$$F = \sum_{i=1}^{I} \alpha E_i^{\text{TOL}} + \beta T_i^{\text{TOL}}, \tag{9}$$

where $\alpha, \beta \in [0, 1]$ denote the weighting parameters balancing the energy and time consumption, and we have $\alpha + \beta = 1$ [6]. Expanding the expression of F, we can further obtain

$$\begin{aligned} F = &\sum_{i=1}^{I} \alpha \left(\left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i k (f_i^{\text{MD}})^2 + \sum_{j=1}^{J} \frac{\lambda_{i,j} Q_i}{R_{i,j}} P_i \right) \\ &+ \beta \left(\frac{\left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i}{f_i^{\text{MD}}} + \sum_{j=1}^{J} \left(\frac{\lambda_{i,j} Q_i}{R_{i,j}} + \frac{\lambda_{i,j} Q_i C_i}{f_j^{\text{MEC}}} \right) \right). \end{aligned} \tag{10}$$

By optimizing the offloading matrix $\mathbf{\Lambda}$, the optimization problem can be obtained by

$$\begin{aligned} \min_{\mathbf{\Lambda}} \quad & F \\ \text{s.t.} \quad & C1: \sum_{i=1}^{I} \lambda_{i,j} Q_i C_i \le K_j, \\ & C2: \lambda_{i,j} \lambda_{i,l} = 0, \ j \ne l, \ l \in \{1, 2, \cdots, J\}. \end{aligned} \tag{11}$$

Note that constraint $C1$ ensures the constraint on computing capacity at each SBS, while the constraint $C2$ represents that an MD only associates with one SBS. With the aim to minimize F, we equivalently maximize the objective function $F' \triangleq -F$, where we have

$$\begin{aligned} &\max_{\mathbf{\Lambda}} F' = \\ &\max_{\mathbf{\Lambda}} \sum_{i=1}^{I} -\alpha \left(\left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i k (f_i^{\text{MD}})^2 + \sum_{j=1}^{J} \frac{\lambda_{i,j} Q_i}{R_{i,j}} P_i \right) \\ &- \beta \left(\frac{\left(1 - \sum_{j=1}^{J} \lambda_{i,j}\right) Q_i C_i}{f_i^{\text{MD}}} + \sum_{j=1}^{J} \left(\frac{\lambda_{i,j} Q_i}{R_{i,j}} + \frac{\lambda_{i,j} Q_i C_i}{f_j^{\text{MEC}}} \right) \right). \end{aligned} \tag{12}$$

We decompose the objective function into J local utility functions as $F_1', F_2', \cdots, F_J'$, where

$$\begin{aligned} F_j' = &\sum_{i=1}^{I} -\alpha \left(\frac{Q_i C_i k (f_i^{\text{MD}})^2}{J} - \lambda_{i,j} Q_i C_i k (f_i^{\text{MD}})^2 + \frac{\lambda_{i,j} Q_i P_i}{R_{i,j}} \right) \\ &- \beta \left(\frac{Q_i C_i}{f_i^{\text{MD}} J} - \frac{\lambda_{i,j} Q_i C_i}{f_i^{\text{MD}}} + \frac{\lambda_{i,j} Q_i}{R_{i,j}} + \frac{\lambda_{i,j} Q_i C_i}{f_j^{\text{MEC}}} \right), \end{aligned} \tag{13}$$

and we have $F' = \sum_{j=1}^{J} F_j'$.

IV. Factor Graph Model

In this section, we develop a factor graph which with variable nodes and factor nodes for the BP algorithm. Generally, variable nodes represent the variables (or parameters) to be optimized, and factor nodes represent the local utilities as the results of the decomposition of the objective function [11]. Furthermore, in our optimization problem, constraints $C1$ and $C2$ need to be handled, and they will be explained later.

A. Variable Nodes

The row vectors $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \cdots, \boldsymbol{\lambda}_I$ can be viewed as the I variable nodes to be optimized. Naturally, the variable node $\boldsymbol{\lambda}_i$ is related to the i-th MD $\mathcal{U}_i$. Meanwhile, two kinds of constraints $C1$ and $C2$ are associated with individual variable nodes. Particularly for $C2$, it represents intra-variable constraints associated with multiple variable nodes.

B. Factor Nodes

We have decomposed the objective function into J local utility functions as $F_1', F_2', \cdots, F_J'$. For each local utility function, the items

$$\sum_{i=1}^{I} -\alpha \frac{Q_i C_i k (f_i^{\text{MD}})^2}{J} \quad \text{and} \quad \sum_{i=1}^{I} -\beta \frac{Q_i C_i}{f_i^{\text{MD}} J}$$

are constant to the variables to be optimized. Hence, they can be neglected during the optimization. We then rewrite the local function as

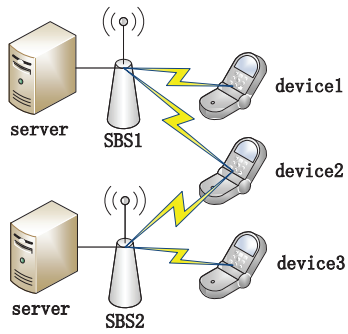
$$\begin{aligned} F_j' = &\sum_{i=1}^{I} -\alpha \left(-\lambda_{i,j} Q_i C_i k (f_i^{\text{MD}})^2 + \frac{\lambda_{i,j} Q_i P_i}{R_{i,j}} \right) \\ &- \beta \left(-\frac{\lambda_{i,j} Q_i C_i}{f_i^{\text{MD}}} + \frac{\lambda_{i,j} Q_i}{R_{i,j}} + \frac{\lambda_{i,j} Q_i C_i}{f_j^{\text{MEC}}} \right). \end{aligned} \tag{14}$$

Since the object function is related to $\sum_{j=1}^{J} F_j'$, we need to maximize each individual F_j' at the jth SBS $\mathcal{B}_j$ so as to maximize the object function F'. We define the local utility functions $F_j', j = 1, 2, \cdots, J$, as the J factor nodes in our factor graph, where the factor node F_j' is related to the SBSs $\mathcal{B}_j$. Meanwhile, due to the constraints of $C1$, we need to reconstruct factor nodes to coordinate the variable nodes for satisfying $C1$. Since the messages among the MDs need to be exchanged and the feasibility of the optimal values need to regulated, the SBSs can be selected as the coordinators. We redefine J local functions G_j, $\forall j$, as the factor nodes related to the J SBSs, which are given by

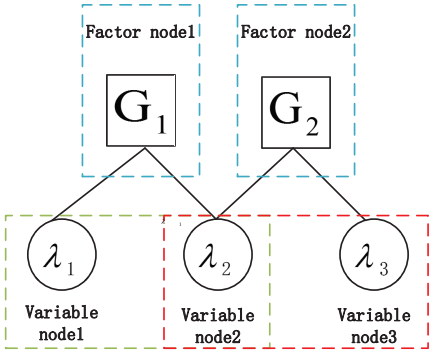
$$G_j = \begin{cases} F_j', & \text{if } \sum_{i=1}^{I} \lambda_{i,j} Q_i C_i \le K_j, \\ -\infty, & \text{otherwise.} \end{cases} \tag{15}$$

C. Edges

The edges in the factor graph connect factor nodes to the related variable nodes. As shown in Fig. 1(a), an example model based on computing offloading contains $J = 2$ SBSs and $I = 3$ MDs. From the figure, the MD 1 and MD 3 are covered by one SBS, and the MD 2 is covered by two



(a) Connectivity between SBSs and devices



(b) The factor graph model

Fig. 1: An example: (a) a system with 2 SBSs and 3 mobile devices, and (b) the corresponding factor graph model.

candidate SBSs for task offloading. Correspondingly, Fig. 1(b) shows the factor graph with three variable nodes and two factor nodes. The variable node $\boldsymbol{\lambda}_1$ and variable node $\boldsymbol{\lambda}_3$ are associated with their factor nodes G_1 and G_2, respectively, while the variable node $\boldsymbol{\lambda}_2$ is associated with the factor nodes G_1 and G_2. Accordingly, the edges in the factor graph emanated from the factor node G_j connect to the variable node $\boldsymbol{\lambda}_i$ if $\mathcal{U}_i$ is covered by $\mathcal{B}_j$.

D. Equivalent Problem

With regard to the BP factor graph, we introduce the equivalent problem of (11) as

$$\max_{\boldsymbol{\Lambda}} G(\boldsymbol{\Lambda}), \quad G(\boldsymbol{\Lambda}) = \sum_{j=1}^{J} G_j, \tag{16}$$

where G_j is defined by the local objective function (15).

V. Distributed Belief Propagation

In this section, we will describe the BP algorithm based on factor graph to perform the optimum computational offloading. In standard BP, the variables are optimized by estimating their marginal probability distributions [12]. We define a probability mass function (PMF) $p(\boldsymbol{\Lambda})$ of $\boldsymbol{\Lambda}$ based on its utility function:

$$p(\boldsymbol{\Lambda}) \triangleq \frac{1}{Z} \exp\left(\mu G(\boldsymbol{\Lambda})\right), \tag{17}$$

where μ represents a positive number and Z represents the normalization factor [9]. As noted in [12], we can obtain the conclusion: the probability distribution function for all possible $\boldsymbol{\Lambda}$ concentrate around the maxima of $G(\boldsymbol{\Lambda})$ as $\mu \to \infty$, i.e., $\lim_{\mu\to\infty} \mathbb{E}(\boldsymbol{\Lambda}) = \arg\max_{\boldsymbol{\Lambda}} G(\boldsymbol{\Lambda})$, in which $\mathbb{E}(\boldsymbol{\Lambda})$ is the expectation of $\boldsymbol{\Lambda}$. A good estimate for the solution of $G(\boldsymbol{\Lambda})$ can be provided by the marginal expectations of the probability distribution with a large enough μ.

As we known that by passing the belief messages between variable and factor nodes, we can compute the marginal distribution of the belief propagation algorithm. The belief message represents the influence that one node exerts on its parent or descendent nodes. Due to the message updated between the variable node $\boldsymbol{\lambda}_i$ and its neighboring factor node G_j, we can obtain the probability mass function of $\boldsymbol{\lambda}_i$ as

$$p(\lambda_{i,j}) = \{\Pr(\lambda_{i,j}^{[1]}), \cdots, \Pr(\lambda_{i,j}^{[n_i]}), \cdots, \Pr(\lambda_{i,j}^{[N_i]})\}, \tag{18}$$

where the n_i-th probability estimated of factor node G_j, i.e., $\lambda_{i,j}^{[n_i]}$, represents the n_i-th value of $\lambda_{i,j}$. The key components of the distributed BP are as follows and the specific process is presented in **Algorithm 1**.

1) *Variable Nodes Update:* In the τth iteration, the variable node i updates the message $p_{i\to j}^{\tau}(\lambda_{i,j})$ for the factor node j based on the messages which were gleaned from i's neighboring factor nodes other than j, and we use $\bar{j}$ to represent these factor nodes. Then we have

$$p_{i\to j}^{\tau}(\lambda_{i,j}) = \frac{1}{Z_i^{\tau}} \prod_{\bar{j}\in\mathcal{H}(i)\setminus j} p_{\bar{j}\to i}^{\tau-1}(\lambda_{i,\bar{j}}), \tag{19}$$

where Z_i^{τ} is normalization constant, i.e., $\Sigma_{n_i} p_{i\to j}^{\tau}(\lambda_{i,j}^{[n_i]}) = 1$, and $\mathcal{H}(i)$ represents the neighboring factor nodes set of variable node i.

2) *Factor Nodes Update:* In the τth iteration, the factor node j updates the message $p_{j\to i}^{\tau}(\lambda_{i,j})$ for the variable node i based on the messages which were gleaned from j's neighboring variable nodes other than i, denoted by $\bar{i}$. The messages updated at the factor nodes are calculated according to the marginal PMF. We can then obtain the expression as

$$\begin{aligned} p_{j\to i}^{\tau}(\lambda_{i,j}) &= \sum_{\sim\boldsymbol{\lambda}_i} \left(\exp(\mu G_j(\lambda_{i,j})) \prod_{\bar{i}\in\mathcal{M}(j)\setminus i} p_{\bar{i}\to j}^{\tau} \right) \\ &= \mathbb{E}_{\sim\boldsymbol{\lambda}_i}(\exp(\mu G_j(\lambda_{i,j}))). \end{aligned} \tag{20}$$

Similarly, $\mathcal{M}(j)$ denotes the neighboring variable node set of factor node j.

3) *Final Decision:* Suppose that we have finished all $\tau^{\max}$ iterations in the distributed BP algorithm, the PMF in variable node i can be obtained by

$$\Pr(\lambda_{i,j} = \lambda_{i,j}^{[n_i]}) = \frac{1}{Z_i^{\tau^{\max}}} \prod_{j\in\mathcal{H}(i)} p_{j\to i}^{\tau^{\max}}(\lambda_{i,j} = \lambda_{i,j}^{[n_i]}). \tag{21}$$

Based on the above equation, the variable node $\boldsymbol{\lambda}_i$ will choose the specific offloading decision to factor node F_j that has the maximum posteriori probability, i.e., $\tilde{n}_i = \arg\max_{n_i} \Pr(\lambda_{i,j} = \lambda_{i,j}^{[n_i]})$.

Algorithm 1 Distributed BP Algorithm

Initialization: Set iteration index $\tau = 1$, and set $p_{i\to j}^{\tau=1}(\lambda_{i,j})$ of all variables uniformly.

1: **repeat**
2: *Variable Nodes Update*: Update $p_{i\to j}^{\tau}(\lambda_{i,j})$ by (19);
3: *Factor Nodes Update*: Update $p_{j\to i}^{\tau}(\lambda_{i,j})$ by (20);
4: $\tau = \tau + 1$;
5: **until** $(\tau > \tau^{\max})$
6: Compute (21) and obtain the $\tilde{n}_i$;

Output: $\tilde{\boldsymbol{\Lambda}} = \arg\max_{\boldsymbol{\Lambda}} \sum_{j=1}^{J} G_j$.

VI. Simulation Results And Discussions

In this section, simulation results of the proposed BP algorithm are presented in comparison with some baseline algorithms. We consider two network scenarios to be tested, one of which is a small-size network consisting of three SBSs and three MDs, and the other one is a medium-size network consisting of five SBSs and eight MDs. Unless otherwise specified, all the related parameters used in our simulations are summarized in the following table.

TABLE I: Simulation Parameters

Parameter	Value
Uplink channel bandwidth, W	10 MHz
Power of Gaussian noise, N_o	10^{-9} W
Frequency of $\mathcal{U}_i$, f_i^{MD}	{0.1,0.2,$\cdots$,0.5}GHz
Frequency of $\mathcal{B}_j$'s MEC server, f_j^{MEC}	1.2GHz
Coefficient of MD's chips, k	10^{-26}
Data size for MD, Q_i	[100,500] KB
Computation complexity per bit, C_i	[500,1500] cycles/bit
Mobile device $\mathcal{U}_i$'s transmission power, P_i	[1,1.2]W
Channel gain from $\mathcal{U}_i$ to SBSs $\mathcal{B}_j$, $g_{i,j}$	$g_{i,j}\sim$E(1)
Max iterations, $\tau^{\max}$	10
Weighting parameter of energy consumption, α	0.6
Weighting parameter of computing time, β	0.4
Task upper-bound of $\mathcal{B}_j$'s MEC server, K_j	4.5×10^9 cycles/slot

In our simulations, f_i^{MD}, Q_i, C_i and P_i are all uniformly distributed within their respective limits and $g_{i,j}$ obeys exponential distribution [7]. We define $K = K_1 = K_2 = \cdots = K_j$ and $f^{\text{MEC}} = f_1^{\text{MEC}} = \cdots = f_j^{\text{MEC}}$, which means all the task upper-bounds and frequencies of MEC servers are equal. Four benchmarks, i.e., greedy algorithm, optimal result (via exhaustive search), random result and local computing result are compared with the proposed distributed BP algorithm. Here, local computing represents the case where there is no task offloading to the MEC servers. Fig. 2 shows the performance of these five algorithms in the small-size network. It can be seen that the performance of the BP algorithm is quite close to the optimal result. As the computing capability's upper-bound of MEC server K increases, the weighted sum of energy and time consumption decreases correspondingly.

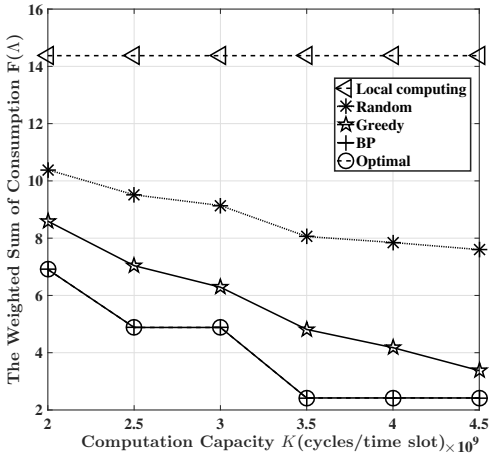


Fig. 2: The weighted sum of consumption versus K in the small-size network.

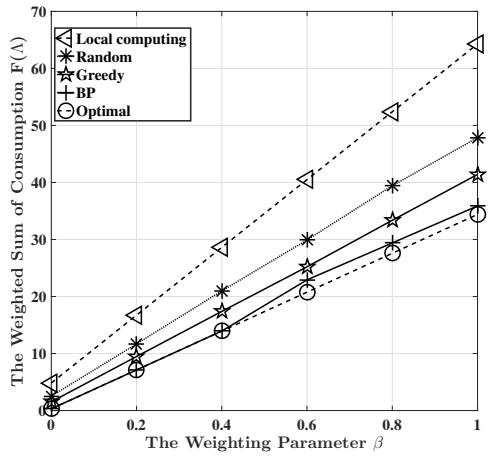


Fig. 3: The weighted sum of consumption versus weighting parameter β in the small-size network.

Weighting parameters α and β are two other factors to consider, as they reflect the importance of computation time or energy cost in the total utility and affect the total utility deeply. Since there is $\alpha + \beta = 1$, we only take the parameter β into consideration. Fig. 3 shows the parameter β's influence on the weighted sum of energy and time consumption in the designed small-size network. From the simulation results, we can see that there is a positively correlated linear relationship between the weighted sum and β, and the corresponding result is also close to the optimal result. We can also conclude that if we want to reduce more total consumption, time consumption should be considered more. The simulation results of the medium-size network are presented in Fig. 4 and Fig. 5 and they are similar to the results of small-size network. Fig. 6 shows the simulation result when considering different MEC server's frequencies f^{MEC} in the medium-size network. We can see that a lager f^{MEC} will help decrease the total consumption, and the BP algorithm can always achieve an excellent performance for different f^{MEC}.

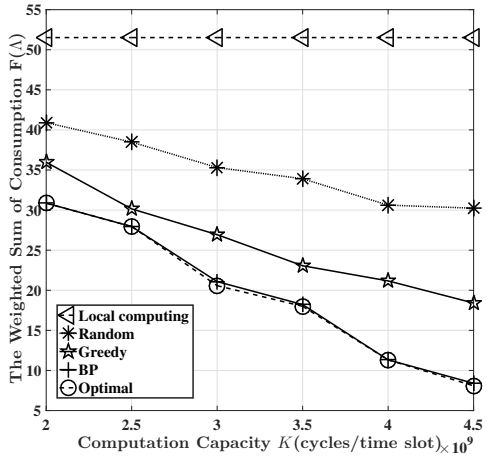


Fig. 4: The weighted sum of consumption versus K in medium-size network.

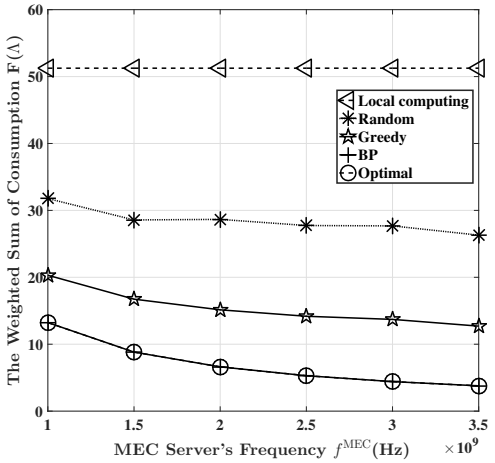


Fig. 6: The weighted sum of consumption versus MEC server's frequency f^{MEC} in medium-size network.

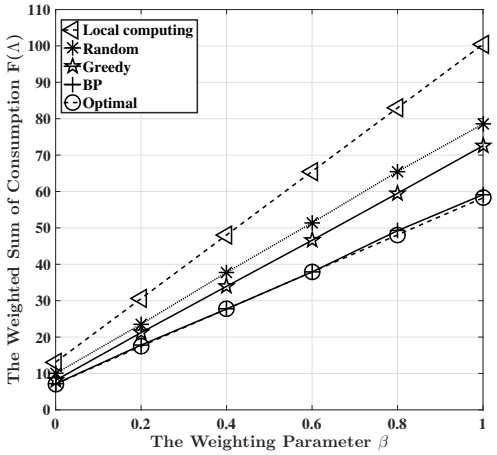


Fig. 5: The weighted sum of consumption versus weighting parameter β in medium-size network.

VII. Conclusion

In this paper, we proposed a distributed scheme which uses BP algorithm to deal with the offloading problem in the MEC-based cellular network. We first took into consideration the weighted sum of computing time and energy cost as our objective function and decomposed the function in a manner that we can build a factor graph model and solve it distributively. A novel BP algorithm was developed to deal with the transformed distributed local utility functions. Simulation results demonstrate that the proposed BP algorithm can achieve a better performance than other basic algorithms, and its performance is very close to the optimum results in both the small and medium scale networks.

Acknowledgement

This work is supported in part by the National Natural Science Foundation of China under Grant 61702258, in part by the China Postdoctoral Science Foundation under grant 2016M591852, in part by Postdoctoral research funding program of Jiangsu Province under grant 1601257C, in part by the China Scholarship Council Grant 201708320001.

References

[1] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

[2] ETSI, Sophia Antipolis, France, "Mobile-edge computing-introductory technical white paper," http://portal.etsi.org/portals/0/tbpages/mec/docs/mobile-edge_computing_introductory_technical_white_paper_v1\%2018-09-14.pdf, [Online].

[3] D. Calin, H. Claussen, and H. Uzunalioglu, "On femto deployment architectures and macrocell offloading benefits in joint macro-femto deployments," *IEEE Comm. Mag.*, vol. 48, no. 1, pp. 26–32, Jan. 2010.

[4] M. Sstyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct.-Dec. 2009.

[5] 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, Aug. 2016.

[6] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[7] C. You, K. Huang, H. Chae, and B. Kim, "Energy-efficient resource allocation for mobile-edge computing offloading," *IEEE Transactions on Wireless Communications.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2013.

[8] P. Zhao, H. Tian, C. Qin, and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access.*, vol. 5, pp. 11 255–11 268, Jun. 2017.

[9] J. Li, Y. Chen, Z. Lin, W. Chen, B. Vucetic, and L. Hanzo, "Distributed caching for data dissemination in the downlink of heterogeneous networks," *IEEE Transactions on Communications*, vol. 63, no. 10, pp. 3553–3568, Jul. 2015.

[10] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[11] Y. Chen, J. Li, W. Chen, Z. Lin, and B. Vucetic, "Joint user association and resource allocation in the downlink of heterogeneous networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 7, pp. 5701–5706, Jul. 2016.

[12] S. Rangan and R. Madan, "Belief propagation methods for intercell interference coordination in femtocell networks," *IEEE J. Select. Areas Commun.*, vol. 30, no. 3, pp. 631–640, Apr. 2012.